CRANFIELD UNIVERSITY

DENIZ TURAN

Defence College of Management and Technology

ON RECOGNITION OF GESTURES ARISING IN FLIGHT DECK OFFICER (FDO) TRAINING

PhD THESIS

CRANFIELD UNIVERSITY

COLLEGE OF DEFENCE TECHNOLOGY

ENGINERRING SYSTEM DEPARTMENT

PhD THESIS

Academic Year 2006-2007

Deniz Turan

On Recognition of Gestures Arising in Flight Deck Officer (FDO) Training.

Supervisor : Dr Venkat V. S. S. Sastry

May 2007

Contents

List of Tables

List of Figures

Abstract

This thesis presents an on-line recognition machine *RM* for the continuous and isolated recognition of dynamic and static gestures that arise in Flight Deck Officer (FDO) training.

This thesis considers 18 distinct and commonly used dynamic and static gestures of FDO. Tracker and computer vision based systems are used to acquire the gestures.

The recognition machine is based on the generic pattern recognition framework. The gestures are represented as templates using summary statistics. The proposed recognition algorithm exploits temporal and spatial characteristics of the gestures via dynamic programming and Markovian process. The algorithm predicts the corresponding index of incremental input data in the templates in an on-line mode. Accumulated consistency in the sequence of prediction provides a similarity measurement (Score) between input data and the templates. Having estimated *Score*, some heuristics are employed to control the declaration in the final stages.

The recognition machine addresses general gesture recognition issues: to recognize real time and dynamic gesture, no starting/end point and inter-intra personal temporal and spatial variance. The first two issues and temporal variance are addressed by the proposed algorithm. The spatial invariance is addressed by introducing independent units to construct gesture models. An important aspect of the algorithm is that it provides an intuitive mechanism for automatic detection of start/end frames of continuous gestures. The algorithm has the additional advantage of providing timely feedback for training purposes.

In this thesis, we consider isolated and continuous gestures. The performance of *RM* is evaluated using six datasets - artificial (W_TTest), hand motion (Yang, Perrotta), Gesture Panel and FDO (tracker, vision). The Hidden Markov Model (HMM) and Dynamic Time Warping (DTW) are used to compare RM's results.

Various data analyses techniques are deployed to reveal the complexity and inter similarity of the datasets before experiments are conducted. In the isolated recognition experiments, the recognition machine obtains comparable results with HMM and

outperforms DTW. In the continuous experiments, *RM* surpasses HMM in terms of sentence and word recognition. In addition to these experiments, a multilayer perceptron neural network (MLPNN) is introduced for the prediction process of *RM* to validate modularity of *RM*.

The overall conclusion of the thesis is that, *RM* achieves comparable results which are in agreement with HMM and DTW. Furthermore, the recognition machine provides more reliable and accurate recognition in the case of missing and noisy data. The recognition machine addresses some common limitations of these algorithms and general temporal pattern recognition in the context of FDO training. The recognition algorithm is thus suited for on-line recognition.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institution of learning.

Acknowledgements

It was almost four and half years ago when, I started my postgraduate studies as a Master of Research student. The following year, I transferred to the PhD programme. I must admit that I did not imagine studing for the PhD would be so challenging at these times. But with guidance, help, support and encouragement from several people, that study has now been completed. Hence, here, I would like to express my sincere gratitude to those people, and also to several organizations.

Foremost, I am deeply grateful to my supervisor Dr. Venkat V. S. S. Sastry for his everlasting support and guidance throughout my PhD study and private life. Being his student has always been a privilege to me. His patience, motivation and enthusiasm were great value for me throughout my study. His analytical approach, detailed and constructive comments and guidance made this study possible. I must also acknowledge Dr. Lakshmi Sastry, my line manager at CCLRC, Rutherford Appleton Laboratory, for her help and understanding especially during the writing up of my thesis.

I owe special thanks to my internal co-supervisors Dr. Trevor J Ringrose, Dr Dr. Evan J. Hughes and Mike R. Bathe for their valuable feedback, discussions and guidance. They provided a great amount of discussion, feedback and resources during my study. Especially, I am grateful to Trevor for reading my internal reports and this thesis constructively.

This study has been fully funded by the Engineering System Department (ESD), which made the research and study possible. My sincere and warms thanks are due to Prof. John Hetherington, Head of Engineering Systems for this funding and his support and understanding during my difficult moments. He always had time for me even in his busiest days.

And of course, I must acknowledge the endless support of the administrative staff, Jo Nash, Ann O'Hea, Claire Lankester, Ros Gibson and Paula Bentley at ESD and Applied Mathematics and Operational Research Group (AMORG). From the very first day to the last day of my study, their warm, sincere and sympathetic support made this study bearable even at its toughest.

I am also thankful to other members of the academic staff and students in AMORG for their discussion and help in various fields. In particular, I enjoyed the technical and non-technical discussion with Dr. Emmanuel Tadjouddine as a colleague and friend. D. Ian Brown has helped with the proofreading of my thesis.

I would also like to thank Ed P. Oates from RNAS Culdrose, UK for providing the FDO images for using in this thesis.

I wish to thank the official referee committee for their constructive feedback and reviews.

I gratefully acknowledge Prof. Yang-Hee Nam and his team for providing the hand gesture dataset (Yang) and correspondence. I also wish to acknowledge Lt. Alex Perrotta for providing hand gesture datasets and users Raj Machavarem, Tracy C. Enderwick, Udeha Ketipearachchi, Lt. Mahir Erken, Zeki Aslan, who contributed to the collection of vision based FDO dataset (FDO_CV).

I also would like to thank my first degree advisor Tatyana Yakhno (Dokuz Eylul University, Turkey), friends from Turkey, and the UK in Swindon and Oxford, UK for their support and encouragement during the study. I know, sometimes I have neglected you due to work, but you were always in my thoughts.

And a very special thank you goes to Marzena Komorowska, for her understanding and patience especially during the writing up of this thesis. She has been a great motivator and inspirational source for me during the writing up.

And last but not least, my family. Words are not enough to express my deepest gratitude to my family for their endless love and support during my last 20 years studentship. Thanks to my beloved mum, Beser, dad Ahmet, sister Devrim and brothers Ismail, Bulent, and Hakan, new and extended members of my family.

If I have forgotten someone to acknowledge, I beg your pardon and wish to thank you too, if you cannot find your name above.

Chapter 1

Introduction

"Let the poses of the people and the parts of their bodies be so disposed
that they display the intent of their minds."
Leonardo da Vinci

Interacting with a computer by natural methods has been attracting more attention as computers play a more important role in our daily lives. The natural ways in which people interact with each other in daily life are suitable for human-computer interaction because of their ease and metaphors used. Gestures are one of the interaction methods used along with other methods such as speech and writing.

Gesture is the second most used interaction type in daily life after speech. In some parts of daily life, especially where the speech interaction is restricted because of various reasons such as noise, and deafness, gestures are a preferred means to interact with the environment. An environment, in which a Flight Deck Officer (FDO) and pilot work, is a typical example. FDOs and pilots interact with each other by means of gestures.

Gestures of FDOs are the main subject of this project. FDO is the officer in charge of ensuring craft (atmospheric and space) maintain operational status and readiness. Safe conduct of flight deck operations for helicopter launches and recoveries on board are some of their responsibilities.

FDOs are trained at the School of Flight Deck Operations (RNSFDO), RNAS Culdrose. RNSFDO comprises several sections, which carry out the training in individual specializations. In a trainee program, students are trained to carry out certain responsibilities such as give clear directions to an approaching helicopter enabling it to land on the flight deck.

The primary aim of this study is to develop a recognition system that recognizes

Figure 1.1: A training session of the Flight Deck Officer (FDO) at the School of Flight Deck Operations (RNSFDO), RNAS Culdrose, UK. The gestures of trainees are interpreted by an instructor who, then, runs further simulator operations (for example taking off a helicopter) according to the performed gesture. The aim of this study is to develop a recognition algorithm to automatically recognize FDO gestures by computer and hence to eliminate the role of the instructor in this virtual training environment (Figure 1.1 is taken from the Royal Navy's Website-www.royal-navy.mod.uk).

FDOs gestures as part of a virtual training system. The current FDO training simulator at RNAS Culdrose requires another person, typically the instructor, to fly the helicopter, in response to the signals given by the trainee. Figure 1.1 illustrates a training session in this virtual environment. This project aims to remove this need, by linking the hand movements of the trainee, via the input devices, to the flight dynamics model of the helicopter. Thus as the trainee waves the helicopter in, for example, the helicopter will move accordingly. This will require effective management of two-handed inputs and/or interactions in the virtual environment. The recognition operation should be on-line in other words, in real-time, as data incrementally becomes available. In addition to that, the user should get some feedback in the case of wrong performance of gestures.

In this paper, a gesture recognition algorithm (Recognition Machine, RM) is proposed to recognize dynamic and static FDO gesture streams in an on-line manner (real time). In gesture streams, the start and end points of gestures are unknown in advance by RM. The proposed system is based on a special matching technique and scoring. The matching technique relies on predicting the index of input points in the class models, which are equal or close to the input point rather than a substring operation. The predicted indices play an important role in the algorithm. The distance between

consecutive predicted indices is used as scoring to determine the similarity of gesture models and input data. The predicted indices are also utilised to detect intuitively the start/end of gestures in the continuous recognition. In addition to that, some predicted indices based on heuristics are used to prevent premature and wrong recognition in case of missing, noisy and unreliable data.

The proposed scheme is expected to solve general gesture recognition issues: to recognize real time and dynamic gesture, without specifying the starting/end points and inter-intra personal temporal and spatial variance. The first two issues and temporal variance are addressed by the proposed algorithm. The spatial invariance is addressed by introducing independent units to construct gesture models.

1.1 Thesis Layout

This thesis starts with a formal discussion of the problem definition in the second chapter, but first, from a wider perspective, a general definition and taxonomy of *gesture* is elaborated. After emphasising the FDO's gestures in "gesture continuum", a formal definition of the problem, notation and terms are represented in the domain of temporal pattern recognition. FDO gestures are elaborated upon with the introduced notation and terminology and the chapter then continues with the surrounding problem definition topics such as challenging issues, other related problems, complexity of the problem and how to measure the performance of any proposed recognition algorithm for the problem.

Gesture recognition is a kind of temporal pattern recognition problem such as speech and handwriting recognition. Around the world, several researchers have been studying temporal pattern recognition problems as well as gesture recognition problem. In the third chapter, a comprehensive literature survey on gesture and temporal pattern recognition systems is presented. The thesis is structured along the lines of a generic gesture (pattern) recognition system which is outlined in this chapter. Components of a generic gesture recognition system such as sensor processing, gesture modelling & analysis and gesture recognition are investigated in detail. In the literature, two types of data acquisition techniques are mainly considered: Computer Vision and Tracker-based. This chapter addresses the advantages and disadvantages of these two sensor techniques both of which are deployed in various experiments in the thesis. The literature review chapter contains detailed discussion on the recognition algorithms such as the Hidden Markov Model (HMM), Dynamic Time Warping (DTW), Neural Networks,

the hybrid system and other related algorithms. Since the proposed recognition algorithm in the thesis is similar to HMM and DTW with some degree, the main emphasis is given to these two recognition algorithms.

Although these algorithms and systems are proposed for recognizing gestures, all of them have similar issues such as source dependent vs. independent, discrete vs. continuous and vocabulary size. All of these issues are the problem of artificial intelligence, which is called complete-AI. Solving all of these issues for a problem means addressing many problems in the various areas of AI. Therefore, proposing a solution to the gesture recognition problem also helps to solve other recognition problems such as speech and handwriting recognition.

Chapter four focuses on gesture modelling and similarity/complexity analysis of datasets. The class models construction scheme is elaborated upon. This scheme is an off-line process and consists of data acquisition, data pre-processing, feature selection/extraction, and construction of class models. Classes are modelled by utilising summary statistics of training cycles with a template based approach. Template based modelling represents the trajectory of classes containing features with summary, compact statistical parameters. Features represent directly or indirectly spatial or temporal properties of classes.

In this thesis, FDO gestures are investigated using two datasets (FDO_PT and FDO_CV). The only difference between these datasets is the method of data acquisition. The FDO_PT dataset is based on tracker input device, whereas FDO_CV is computer vision based. These two datasets have common properties apart from the number of samples, data acquisition methods, and the number of users used to perform gestures. While the FDO_PT deployed only one user (the author himself), the FDO_CT deployed four users.

Channel construction is based on estimating the parameters which represent the best underlying statistical distribution of training data. In this study it is assumed that, features are independent of each other and training data obeys normal statistical distribution. Therefore, summary template based channel construction procedure is based on estimating the parameters of statistical mean μ and standard deviation σ.

Having constructed the class models, one is lead to the issue of intra/inter similarities between classes and their samples. Therefore, in the second half of the fourth chapter, a comprehensive complexity and similarity analysis of class models are addressed. For this purpose, several well established techniques are considered for inter/intra class temporal and spatial similarity and complexity analysis. Discussions of these techniques are supported and exemplified over an artificial dataset W_Test, before a detailed analysis on FDO datasets is carried out.

The next chapter deals with the proposed recognition algorithm (Recognition Machine, RM). The recognition machine (RM) is conceptually an on-line template matching technique. The main idea behind the recognition algorithm is to exploit sequential consistency of the input frames according to class models by using a dynamic programming paradigm and the Markovian process. Sequential consistency or so-called *Score* (S) addresses the similarity between the incremental input data and the class models. *Scores* are aggregated sequentially for each class for the given input by deploying some similarity factors and prediction process in an on-line manner. The similarity factors are mainly based on membership degree of input data to the class model of interest and the distance between consecutive predicted frames index. The prediction process is a probabilistic estimation of the index of frames for each class which are spatially closest to the input frame, given the most recently predicted frame index.

Recognition machine is implemented according to the classical pattern recognition framework [87]. The recognition machine (RM) consists of various interacting components, each of which is responsible for a task in generic pattern recognition such as sensor processing, class modelling and feature extraction and recognition tasks.

The proposed scheme is expected to solve general gesture recognition issues: to recognize real time and dynamic gesture, without specifying starting/end points and inter-intra personal temporal and spatial variance. The first two issues and temporal variance are expected to be tackled by the proposed algorithm. For overcoming the spatial invariance, class models are constructed using user independent features.

The chapter also covers the discussion relating to time and space complexity of the algorithm and analogy of RM with other popular temporal recognition algorithms. The analogy focuses especially on the Hidden Markov Model and Dynamic Time Warping. In addition to that, a hybrid RM, which utilises a multilayer perceptron neuron network for index prediction, is proposed in order to discuss modularity of RM.

The sixth chapter focuses on isolated and continuous gesture recognition experiments and discussions surrounding the proposed recognition algorithm (Recognition Machine, RM). It also covers the comparison of RM with other well established recognition algorithms on various datasets. In addition to FDO datasets (FDO_PT and FDO_CV), this thesis conducts several experiments over one parametric artificial (W_Test [57]) and three other supplementary real world datasets (Gesture Panel [146], Yang [64, 63], Perrotta [89]). The artificial dataset W_Test accommodates several control parameters which are utilised to analyse the performance of RM and other recognition algorithms under various combinations of control parameters. Supplementary real world datasets are included to analyse the performance of RM in various real world scenarios in terms of characteristics of data, data acquisition techniques (for example

computer vision, tracker based) and a number of deployed users for data collection. Various data analyses techniques, which are elaborated in the Gesture Modelling and Analysis chapters, are deployed to reveal the complexity and inter similarity of datasets before the recognition experiments are carried out.

Two types of gesture recognition experiments are conducted in this study: isolated and continuous gestures. As the recognition algorithms, the Dynamic Time Warping (DTW), the Hidden Markov Model (HMM), the Elman Recurrent Neural Network (ERNN) and the Recognition Machine (RM) are considered for isolated recognition over all the datasets. In the case of continuous recognition, due to their achievement in isolated recognition experiment, only HMM and RM are considered on FDO_PT and FDO_CV sentences with various length and control parameters. In addition to these, isolated recognition experiments are conducted on the FDO_PT dataset by the hybrid version of RM.

Finally, the thesis concludes with a summary, important achievements and future work.

1.2 Contribution

This study contributes in the temporal pattern recognition domain, and specifically in the FDOs gesture recognition domain with the following outcomes and achievements:

- Recognition Machine (RM): The study proposes a recognition machine (RM) which addresses the following issues of generic temporal pattern and gesture recognition:
  - Recognition of dynamic and static gestures.
  - Isolated and continuous recognition.
  - On-line recognition.
  - Automatic Segmentation: RM is able to detect start/end points of gestures in continuous recognition.
  - It exploits dynamic programming and the Markovian process; in doing so, it addresses several limitation of existing recognition algorithms such as Dynamic Time Warping (DTW) and the Hidden Markov Model (HMM). Conclusions related to these algorithmic issues are as follows:
    * RM intuitively deals with traditional issues of HMM such as evaluation, decoding and topology. For topology, RM employs an ergodic architecture, which is biased from right to left with a larger number of states.

For training and modelling the class, RM accumulates summary statistics with a template based representation. RM intuitively addresses the decoding process by employing some dynamic programming scheme.

* Weak criteria to announce the classification: The study addresses the weak criteria (maximum likelihood) of HMM to announce the classification with some novel additional control heuristics in RM. The probability of models $P(O|\lambda)$ (where O is the observation vector and λ is the HMM model), is the only criteria (maximum likelihood $P(O|\lambda)$) to announce a recognition in HMM. This criterion is weak in the case of on-line recognition in which start and end points of patterns are not known in advance. Whereas, RM employs some heuristics during on-line recognition to prevent/reject premature, unreliable recognitions. This feature of RM is novel.
* HMM obtains best results in the case of a smaller number of states which do not provide meaningful feedback (observation sequence in decoding) for training. Whereas, RM employs a larger number of states (as it is concluded in [102]) to represent characteristic of gesture; therefore, the decoding process provides more meaningful feedback (observation sequence).
* HMM assumes that observations are managed with some underlying "*hidden*" states. Whereas, in case of gesture recognition, states are more observable; therefore, RM represents gestures with a large number of *observable* states which is useful for meaningful decoding sequences.
* HMM makes a questionable recognition, in the case of missing or incorrect data, which arises due to the only deployed criteria, maximum likelihood. Whereas, RM approaches the data more carefully, and it rejects any recognition in the case of incorrect and missing data by some built-in heuristics. This is an important feature for training purposes, because RM can detect these mistakes in the performance of gestures in training sessions and as a result provides better training.
* HMM tends to model undefined, transition movements. Since these approaches have to model a large number of movements, their success is limited. Whereas, RM deploys a heuristic based rejection scheme to spot undefined, transition data between gestures.
* HMM and RM are based on the first order Markovian process. But RM can be easily extended to the higher order Markovian Process by

utilising some historical data.

  - RM is a modular (component) based architecture, so different techniques can be deployed for components to obtain more efficient overall results. This thesis also introduces a hybrid system out of neural networks, HMM and DTW.
  - RM is able to provide timely feedback for training purposes.

- Template Based Approach: For the representation of classes, in the last few decades, the interest in using the template based approach has been declining. But this study, deploys template based modelling, instead of the more popular feature-based representation. A comprehensive discussion is presented for template based modelling. But unlike the traditional template based approach, in which the distance or similarity between input signal and templates are estimated in terms of Euclidean (or area, volume) distance, in this thesis, templates are just used for representation. In other words, the similarity between templates and input signals in this thesis is estimated in terms of distance between predicted consecutive indices. This serves to estimate the similarity between input signal and templates without knowing the start/end points of gestures, which is needed for euclidean distance estimation, in the case of continuous recognition [147].

- Temporal datasets: In order to validate RM, several temporal datasets, (artificial and real world) are used besides FDO gestures. Two types of FDO datasets have been created by the author, the first of which is tracker based (FDO_PT) and the second one is based on computer vision (FDO_CV). These datasets consist of sufficient complex representatives of real life applications. FDO dataset is available on-line at the following web address with a description article and MATLAB scripts to extract data from samples file.

  http://personal.rmcs.cranfield.ac.uk/˜turand

- Complexity and Similarity Analysis: This study presents a variety of complexity and similarity (temporal and spatial) analysis techniques for datasets. While some of these techniques are novel and proposed by the author, some of them are modified for temporal classes from classical static pattern classification.

- Comprehensive Literature Review: This study, both in general temporal pattern recognition and in specific FDO's gesture recognition presents, a comprehensive and structured literature review. Latest trends and techniques in data acquisition

and pre-processing, class modelling/analysing and recognition algorithms in these domains are discussed with their advantages and disadvantages.

- Recognition Experiments: This study conducts several isolated and continuous experiments on aforementioned temporal datasets with various well established techniques such as the Hidden Markov Model (HMM), Dynamic Time Warping (DTW) and Elman Neural Networks (ENN) besides the Recognition Machine.

- Regarding other recognition algorithms such as DTW and recurrent neural networks (RNN), based on experiments and literature reviews, the study notes that these algorithm have their limitation for gesture recognition.

- Published papers: The following papers have been published out of this study:

  - Deniz T. Sodiri [1] and Venkat V. S. S. Sastry; *On the Interpretation of Gestures arising in Flight Deck Officers Training*; SISO, 13th Conference on Behavior Representation in Modeling and Simulation; Virginia, USA, May, 2004 [119].

  - Deniz T. Sodiri and Venkat V. S. S. Sastry; *Recognition Machine (RM) for On-line and Isolated Flight Deck Officer (FDO) Gestures*; IJIT, International Journal of Intelligent Technology, Volume 1, pages 138-145, 2006 [121].

  - Deniz T. Sodiri and Venkat V. S. S. Sastry; *On-line Recognition of Isolated Gestures of Flight Deck Officers (FDO)*; Transactions on Engineering, Computing and Technology, Volume 13, pages 119-126, Budapest, 2006 [120].

[1] The Author of thesis (Deniz Turan) uses Deniz T. Sodiri as pen name in his publications.

Chapter 2

Problem Definition

"The eyes have one language everywhere"
George Herbert

2.1 Definition of Gesture

The Concise Oxford Dictionary defines gesture as: "A movement of part of the body, especially a hand or the head, to express an idea or meaning." However, its common usage in several areas of daily life by humans makes it difficult to come up with a unified gesture definition. Thus, each discipline constructs and limits its boundaries and definition by itself. For example, from a biological perspective [85]:

> "The notion of gesture is to embrace all kinds of instances where an individual engages in movements whose communicative intent is paramount, manifest, and openly acknowledged"

On the other hand, in social sciences you can just see gesture and non-gesture movements. Social scientists prefer to distinguish gesture from other movement rather than define it and systematize them depends on their requirements and content [59].

The following are typical examples of gestures: okay (thumbs up), begging (flat hand), counting (fingers or/and hand), waving and saluting, praying, raising an eyebrow, marshalling signals, giving finger, making a British Sign Language (BSL) sign, expressing anger (raising a fist), accusation (index pointing), rejection (index up moving left & right), dance, traffic control signals, dinner table action, moving, touching and interacting with objects, conducting an orchestra, etc.

In Human-Computer Interaction (HCI) or computer science literature, gestures are generally based on hand and arm movements. For clarifying the concept the following two definitions are widely used in AI literature:

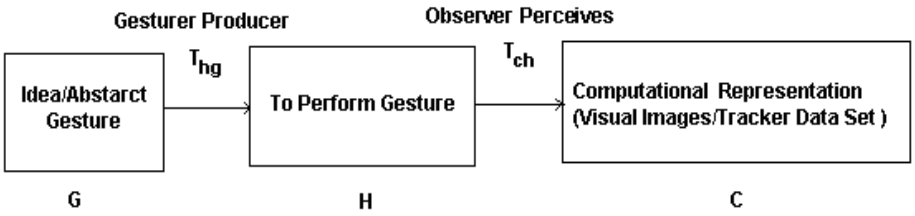


Figure 2.1: Production and perception of gestures [87]

" whole hand input as full and direct use of the hand's capabilities for the control of computer-mediated tasks." [125]

"Gestures are expressive, meaningful, body motion -i.e., physical movement of the fingers, hands, arms, head, face or body with the intent to convey information or interact with the environment." [133]

In this sense, "blowing a kiss" is a gesture but pressing a key on a keyboard is not a gesture because the motion of a finger is neither significant nor observed. All that matters here that pressing a key does a job rather than conveys a message or interacts with the environment [67].

In order to make a mathematical definition of gesture, we can use a speech recognition paradigm [87]. Speech recognition and gesture recognition resemble each other in respect of their production and perception. Therefore, we can apply techniques to describe speech to gesture. Figure 2.1 depicts a modified variation of this technique. According to this technique, abstract gestures, which correspond to the concept of gestures in mind, are transformed to activity (H) by model T_{hg}. Then, performed gesture (H), is transformed to computational representation (C) by T_{ch} model. Computational representation can be in different forms according to the type of sensor used. For example, if a vision-based sensor is used, it can be a visual image; if a tracker or glove based sensor is used, it will be a type of sensor device's output. Formally this technique can be expressed as:

$$\begin{aligned} H &= T_{hg}G \\ C &= T_{ch}H \\ C &= T_{ch}(T_{hg}G) = T_{cg}G \end{aligned}$$

T_{hg} is a model of performed gestures given gesture (G), T_{ch} is a model of computational representation given performed gesture (H), and T_{cg} indicates how computational representation (C) is constructed given some gesture (G). All models are parametric and belong to model parameter space M_T. In the light of these abstract notations, a gesture (G), can be defined as following [87]:

Definition 1 *A gesture (G) is a trajectory of model parameters in modelling parameter space M_T over a defined time interval I.*

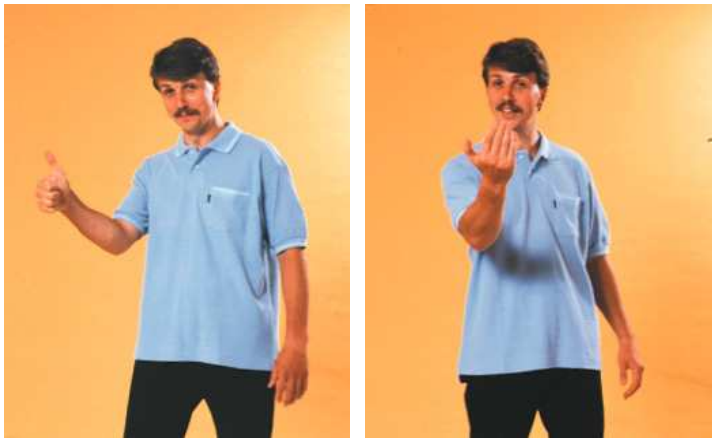


Figure 2.2: Examples of static (Okay, Thumbs up) and dynamic gestures (come) commonly used in daily life [150].

Gestures can be either static or dynamic. Static gestures are those having certain poses or configuration. In other words, the trajectory of gesture does not change with time. For example, okay (Thumbs up) is an example of a static gesture. On the other hand, a dynamic gesture consists of motions of either the same or different static gestures/posture, such as come here. Figure 2.2 illustrates these static (Okay) and dynamic (come here) gestures.

2.1.1 Gesture Taxonomy

There is considerable ambiguity over the definition of gesture, which leads to certain vagueness in the taxonomy of gestures. Several alternative taxonomies have been proposed by psychologists and linguists in the literature. Human gestures are categorized into three groups according to their functional roles by Cadoz [13]:

- Semiotic : Communicative purposed gestures for conveying information to the environment. These gestures change from culture to culture. Saluting, waving goodbye, British Sign Language, and marshalling signals are typical examples.

- Ergotic: Manipulative purposed gestures for modifying and transforming the environment usually associated with the intuition of work. Examples of these gesture include Wiping dusting and peeling an orange, combing hair.

- Epistemic : Discover purposed gesture for getting information from the environment through haptic experience - touching or moving hand around an object to try to detect an object. For example, in a dark room, trying to find light switch on the wall by touching.

Kendon [59], divides gesture into two main groups "autonomous gestures", which is independent from speech, and "gesture continuum", which occurs in association with speech with regards to language. Furthermore, "gesture continuum" is subdivided into five subgroups which can be considered as subgroups of semiotic gestures which are described below:

- Gesticulation: Spontaneous movements of hands and arms during speech.

- Language-like: Gestures like gesticulation that are integrated into spoken utterance, replacing a particular spoken word or phrases.

- Pantomimes: Gestures that depict objects or actions, with or without accompanying speech.

- Emblems: Gestures like "V for victory", "thumbs up", insulting and praising

- Sign Languages: Well-defined set of gestures and postures for a fully fledged linguistic communication system such as British Sign Language.

Dependency to speech in Kendon's continuum decreases from Gesticulation to Sign Language and 90 % of human gestures fall into the Gesticulation subgroups. Even on the phone people gesture, and blind people gesture during speech.

Cognitive scientist McNeill [76, 77] constructs a similar taxonomy:

- Iconic: gestures depicting a concrete object or event and bearing a close formal relationship to the semantic content of speech

- Metaphoric: as iconic but rather than concrete object, abstract idea is gestured

- Beat: small formless gestures, emphasizing a word or phrase

- Deictic: gestures pointing people, objects, event either concrete or abstract in space and time

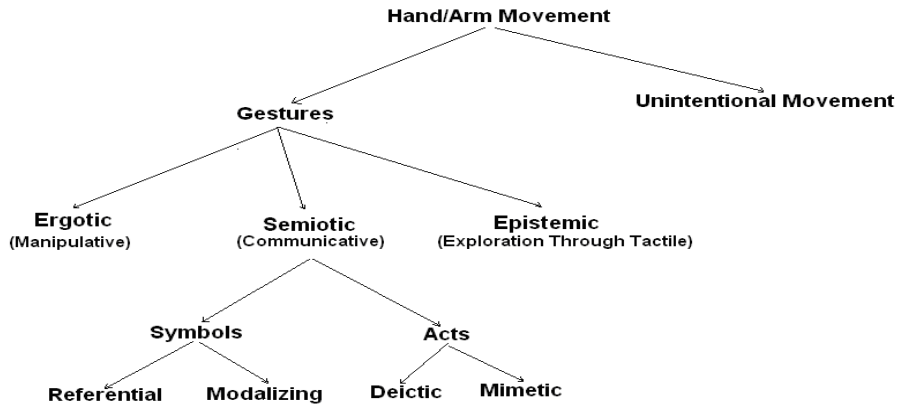


Figure 2.3: Gesture Taxonomy

Probably the most appropriate taxonomy in the context of HCI is the one proposed by Quek [93, 92] and modified by Pavlovic, Sharma and Thomas [87]. This classification is virtually a combination of the above taxonomies. Figure 2.3 depicts this taxonomy. Here, firstly, human hand/arm movement is divided into two groups:

- Unintentional Movements
- Gestures

Then gestures are subdivided into three groups as in Cadoz's taxonomy. In HCI, communication purposed gestures (Semiotic) are used. Further classification is then made under the Semiotic branch resulting in Symbolic and Acts. Symbolic gestures are spontaneous gestures that accompany speech and have linguistic roles. Symbolic gestures are either used for referring (referential gestures, for example, the circular motion of the index finger for referring to a wheel) or modelling (modalizing gestures, for example, disclosing "Look at that wing" and then modelling the gesture by specifying the wing is vibrating by hand and arm movement). Referential and modalizing gestures are the most commonly used gestures in HCI. Acts gestures can be either deictic, which is the action of pointing at something or somebody, or can be mimetic, which is the action of imitating something.

2.2 Formal Definition of The Problem

In order to comprehend the problem in all aspects and to represent a common notation in the remainder of this thesis, we propose a formal definition of real time, dynamic

gesture recognition problem. The problem (Ξ) can be specified as five-tuple *(C, L,H, F, B)* where:

- C is the set of class model with cardinality of ϖ. Classes are modelled based on the template approach.
  $C = (C_1, C_2, C_3, \cdots C_{\varpi},)$

- L is the set of period/length of class templates.
  $L = (l_1, l_2, l_3 \cdots l_{\varpi}) \qquad 0 < l_{1 \cdots \varpi} << T;$
  T is the length of an experiment, and period of each class is small compared to the experiments (T) in on-line recognition. Period of classes may be different from each others.

- C_i is the i^{th} class model template and consists of channel (H) with cardinality of ϑ. Each channel accompanies a time series of a particular feature.
  $C_i = \{H_{i,1}, H_{i,2}, H_{i,3} \cdots H_{i,\vartheta}\}, \quad 0 < i \leq \varpi$

- $H_{i,j}$ is the j^{th} channel of i^{th} class model and consists of a time sequence of the j^{th} feature(f_j). Note that, in the remainder of the report, attribute, channel and component are used interchangeably.
  $H_{i,j} = \{f_{j,1}, f_{j,2}, f_{j,3} \cdots f_{j,l_i}\}, \quad 0 < i \leq \varpi \; and \; 0 < j \leq \vartheta$

- F is set of features or alphabet with cardinality of ϑ. The domain or set of alphabet for each feature can be different. Features may indicate discrete values or intervals as much as continuous values and intervals. Streams of a feature determine a channel.
  $F = \{f_1, f_2, f_3 \cdots f_{\vartheta}\}$

- B is an ϑ dimensional input channel which consist of the historical set of incremental test data (b). The content of B is a combinatorial variant of defined (C) and non-defined classes (C_{NoN}). Since B is provided incrementally at time t, the recognition algorithm can only use the present and previous data on the channels.
  $B = \{b_t\} \quad b_t = \{f_1, f_2, \cdots f_{\vartheta}\} \quad 0 < t \leq T$ or in a higher representation
  $B = Z(\widetilde{C_1}, \widetilde{C_2}, \cdots, \widetilde{C_{\varpi}}, C_{NoN});$
  where $\widetilde{C_i}$ indicates of variant of class C_i and $Z(.)$ is a grammar function, which regulates syntax of given problem domain.

The above notation is illustrated in figure 2.4 and further elaborated in section 2.3. In the light of these notations, similar to Pavlovic's [87], a temporal class or gesture can be defined as follows:

Definition 2 *A temporal class C_i, is a trajectory of points expressed in channels ($H_{i,1\cdots\vartheta}$) in a ϑ dimensional feature space, F, over a defined time interval l_i.*

Classes can be either static or dynamic. Static classes are those that have certain poses or configurations called trajectories that remain approximately constant for the period of the class. In other words, the trajectory of a class does not change with time. If C_i is a static class then data in all channels remains nearly constant.

$$f_{j,1} \cong f_{j,2} \cong f_{j,3} \cong \cdots \cong f_{j,l_i} \text{ where } f_{j,t} \in H_{i,j} \; ; \; 0 < j \leq \vartheta \; ; \; 0 < i <\leq \varpi 0 < t \leq l_i \quad (2.1)$$

Note that approximate equality accommodates slight tremors while holding the pose. On the other hand, dynamic classes are the motion of same or different static classes. Another important phenomenon that needs to be addressed is sub-events. Sub-events are short phenomena such as peak points, global or/and local maximum/minimum points, in channels which are distinctive compared to the rest of the channel. Sub-events are formed due to the sudden change of speed, direction and acceleration. For example, figure 2.4 shows three sub-events in an angular velocity channel. Sub-events are important phenomena for class model construction and recognition algorithm.

Using this notation, the problem can be expressed as:

Problem 1 *Given four-tuple (C, L, H, F) and incremental data in B, develop an algorithm or a recognition machine (RM) to recognize the classes to which each part of B belongs.*

$$\Xi \;=\; RM(B|C,L,H,F) \quad (2.2)$$

Note that, even the tuples L and H are sub-components of the class tuple C, it is shown explicitly in the problem definition to illustrate the underlying structure of the problem.

2.3 Notation and Terms

In this section, notation and terminology used in this thesis are illustrated with an example.

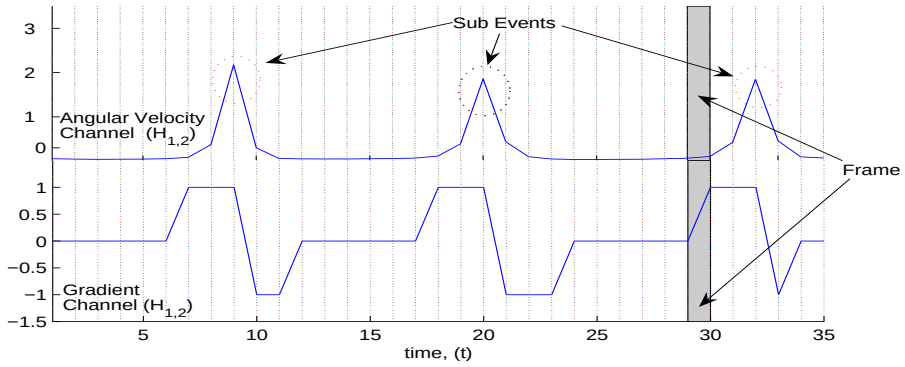


Figure 2.4: Fictitious temporal class C_1. The class, C_1 has two channels, the first of which is the angular velocity and the second one is gradient of the first channel. The period of class, or in other words, the number of frames in channels, l_1 is 35. 30^{th} frame is illustrated as an example. Three sub events in the first channel are also shown.

We clarify the problem definition notation with the help of fictitious temporal classes which are used to describe a synthetic problem. The synthetic problem has three temporal classes each of which has two channels. Let us assume that the first channel represents the angular velocity and the second one is its gradient. Subsequently, the number of features is two. The domain of the first and second features, are $[0, \pi]$ and $\{-1, 0, 1\}$ respectively. Then, the problem can be represented as $\Xi(C, L, H, F, B)$ where:

$$
\begin{aligned}
C &= (C_1, C_2, C_3) \\
L &= (35, 40, 25) \\
C_1 &= \{H_{1,1}, H_{1,2}\} \;\; C_2 = \{H_{2,1} \;\; H_{2,1}\} \;\; C_3 = \{H_{3,1}, H_{3,1}\} \\
H &= \cdots \\
F &= \{f_1, f_2\}, \qquad f_1 \in [0, \pi], \;\; f2 \in \{-1, 1, 0\} \\
B &= \{\widetilde{C_1}, \widetilde{C_2}, \widetilde{C_3}, C_{NoN}\}^+
\end{aligned}
$$

Since channels consist of huge data, for sake of clarity, they are skipped above. However, channels ($H_{1,1}$ and $H_{1,2}$) of the fictitious class C_1 can be seen in figure 2.4. In addition, the figure denotes the terms and the underlying structure of classes. The band or test data, B consists of the infinite combination ($\{.\}^t$) of defined class C_i or

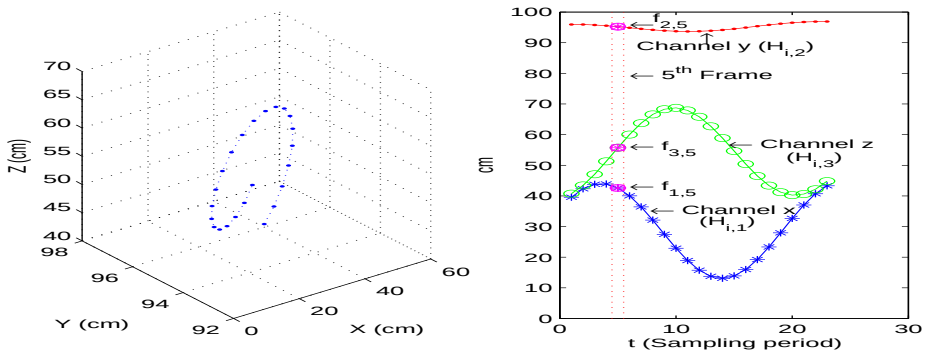


Figure 2.5: The right hand trajectory for *Lashing* gesture (left) is used to illustrate the notation of the problem definition. For simplicity, coordinate units (x, y, z) are selected as features (F=$\{f_1 = x, f_2 = y, f_3 = z\}$). Hence gesture has three channels (right). Period of the gesture ($l_{lashing}$) is 23. The fifth frame consists of the coordinates (x, y, z) at the fifth time step is shown.

undefined class C_{NoN}. As mentioned in the problem definition, $\widetilde{C}_i$ accommodates the variant (either because of noise or other intra, inter variance) of the referred class C_i.

As figure 2.4 demonstrates, a channel is a sequential collection of *frames*, each of which contains a feature for a specific time. In fact, in the study, a frame corresponds to values of all channels at a specific time in a class. Note that frame is used interchangeably with point and index in some parts of this thesis such as in the class model construction and recognition algorithm sections.

In a channel, some frames convey more dominants information than others. These frames, called *sub-events*, generally characterise the channel, and then the class. Therefore, sub events play an important role for analysing and modelling of the classes. They are especially useful for tackling the temporal variance, time alignment in model construction, and frame prediction in the recognition algorithm. Figure 2.4 illustrates the three sub events of class C_1. These sub events of interest accommodate sudden changes in angular velocity or more specifically the direction of components which accumulates angular velocity.

Having explained the notation over a fictitious temporal class, now an example from the FDO dataset can be given. Figure 2.5 illustrates the notation using the *Lashing* FDO gesture. For simplicity, raw coordinate data is used as features (x, y, z). Figure 2.5 shows the fifth frame. It is also worth noting that channels do not accommodate any sub-event.

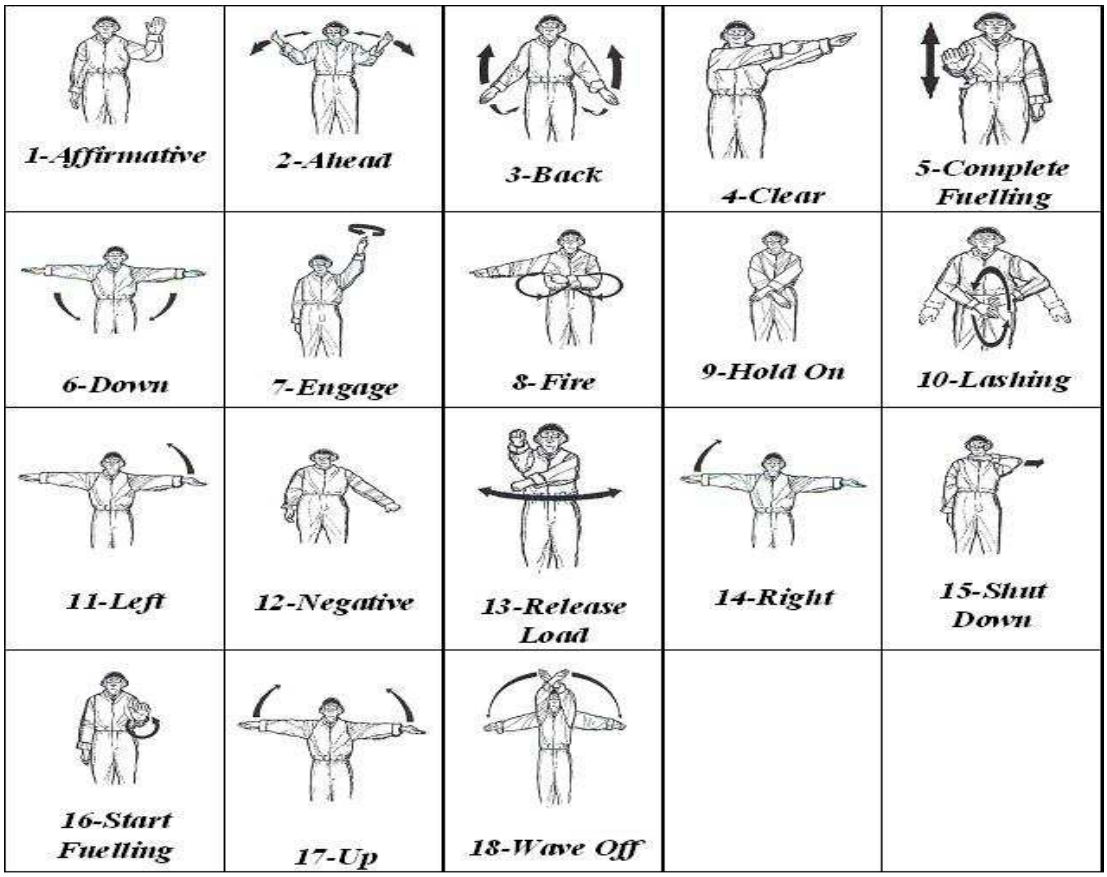


Figure 2.6: This study consists of 18 FDO gestures out of a total 94 FDO gestures. The subset FDO gesture of interest consists of four static (*Affirmative*, *Clear*, *Hold On* and *Negative*), six dynamic (*Ahead*, *Back*, *Down*,*Lashing*, *Up* and *Wave Off*) and eight hybrid (while one hand static, the other hand is dynamic, *Complete Fueling*, *Engage*, *Fire*, *Left*, *Release Load*, *Right*, *Shut Down*, *Start Fueling*).

2.4 FDO Gestures

The FDO gestures considered in this study are NATO FDO Marshalling signals [149]. FDO gestures are *semiotic*, *sign language* and *iconic* according to the taxonomy of Cadoz, Kendon and McNeill respectively. For various tasks such as launching, landing, handling, refuelling and recovering, FDO and pilots use approximately 94 gestures to interact. A full list of these gestures can be found in the appendix. FDO gestures are arm-hand based and both arm and hands are used. During nights, FDOs use two different coloured illuminated wands to give signals which are similar to the day-time arm-hand based gestures [28].

Although FDO gestures are hand-arm based gestures, (for example, in engage gesture, the number of fingers indicates the engine number), in this study, hand is considered as a whole (the configuration of fingers is omitted), because of the limitation

of the tracker input device used. Main emphasis is given to the trajectory of the hand motion in 2D or 3D Cartesian coordinate space.

In this study, a subset of FDO (18) gestures is selected as sample classes ($\varpi = 18, C_{FDO} = C_1, C_2, \ldots C_{18}$). Figure 2.6 illustrates four static gestures (*Affirmative*, *Clear*, *Hold On* and *Negative*), six dynamic gestures (*Ahead*, *Back*, *Down*,*Lashing*, *Up* and *Wave Off*) and eight hybrid gestures (*Complete Fueling*, *Engage*, *Fire*, *Left*, *Release Load*, *Right*, *Shut Down*, *Start Fueling*). In hybrid gestures, while one of the hands is static, the other hand is dynamic.

When these gestures are chosen out of 94 gestures, it is aimed to represent the major characteristics and the challenges posed by the complete FDO gestures. For example:

- The gesture subset, as mentioned above, proportionally consists of the samples of static, dynamic and hybrid gestures.

- Some gestures are performed in the same feature space (3D) with different starting points (Up, Wave Off) or temporal properties (Engage, Affirmative).

- While some gestures are performed when the arms are stretched fully, in some others, the arms are bent at the elbow which causes higher inter/intra spatial variance (Ahead, Fire).

- While some dynamic gestures are performed in a small volume, others are performed in a bigger volume (Engage, Up).

This subset is typical and sufficiently challenging. Once reliable and efficient results are obtained on this subset, the system can be extended to the rest of the FDO gestures.

The data for FDO gestures is acquired using both tracker and vision-based techniques. In both techniques, Cartesian coordinates of the hands, (x, y, z) or (x, y) are collected in the tracker and vision-based techniques respectively. Each gesture lasts between 3-4 seconds approximately. This amount to the number of frames data points for a gesture. The period of classes (L_i) ranges from 20 (*Affirmative*) to 39 (*Wave Off*) ($20 \leq L_i \leq 39$, where $1 \leq i \leq 18$). Figure 2.5 illustrates the three coordinate (spatial) channels ($H_{10,1} = f_1 = x; H_{10,2} = f_2 = y; H_{10,3} = f_3 = z$) for the *Lashing* gesture. Note that the feature set (F) is not only limited to three coordinates, it also has their gradient as a features, but for the sake of clarity, they are omitted in the figure.

2.5 Gesture Recognition Issues

Gesture recognition systems have some challenging issues to confront in order to achieve the above criteria. These issues depend on what is expected from the system. If a system wants to recognize real-time, continuous and dynamic gestures it should overcome these issues. As we will see later, these issues are not related to just gesture recognition but also other temporal pattern recognitions such as speech and handwriting. Some of the important issues are listed below [84]:

- Spatial Complexity and Variance: A gesture may not be performed the same way every time in 3D space due to intra-and inter-personal differences. These differences are:
  - Shape Variations: This is the variance about volume, area and size of gesture source (arm, hand). This is generally inter-person variance.
  - Rotational Variance: This is the variance when a dynamic gesture is performed in various rotations. It can be either inter-person or intra-person variance.
  - Translational Variance: This is variance about the position where gesture is performed. It can be either inter-person or intra-person variance.
- Temporal Variance: A dynamic gesture can be performed with varying speeds. Temporal variance generally occurs because of emotional context (information) of the gesture and environment. When this happens the order of sub-events can change from channel to channel. This variation can occur in the same person or from person to person.
- Start/End Detection: The system perceives and analyses data from the environment incrementally. Acquired data does not have any mark indicating when a gesture starts and ends, particularly when no external indicator is used. To know the start and end point of a gesture helps the recognition algorithm to recognize it more easily. For overcoming this problem, some systems indicate these points explicitly; however this limits its naturalness.
- Repeatability and Connectivity: For conveying a message plainly or strengthening the meaning of a gesture, the gesture can be performed repeatedly. And again, like in any sign language, another gesture can start immediately after another one. The repeatability and connectivity of gestures make the input data more complicated to be automatically segmented by recognition algorithms.

2.6 Related Problems

The gesture recognition problem is akin to the temporal recognition problem which has been investigated since the early years of computer science and artificial intelligence. Speech recognition, hand writing recognition, tracking and recognition of objects in a dynamic environment, emotion recognition are in the domain of temporal recognition problem. Especially, speech recognition has attained huge interest. One thing common to all these systems is that a signal (gesture, speech, handwriting) moves over space and time and leaves a trajectory.

Following are the parallel issues with temporal pattern recognition (especially speech and handwriting) [16], [143], [88]:

- Source Dependent vs. Independent: Resemblance to spatial and temporal variance issues of gesture recognition systems. Does a signal depend on its source? Recognition from a single source is easier to compare than from multiple sources. In the case of a single source some optimization and constraint can be applied. But that can use more resources and time to deploy in the case of multiple sources.

- Isolated or Continuous: Corresponds to repeatability and connectivity issues of gesture recognition systems. A continuous signal adds more complexity to a system because of the start/end and repeatability/connectivity problem. In isolated cases, prior knowledge of the start/end points of classes is known in advance.

- Vocabulary Size (Perplexity): Represents the number of classes in the system. The gist of the classification task is to divide feature space into segments in a way (linear or non-linear) that each segment corresponds to a class. Shared areas between segments directly affect the recognition task. Therefore, in a closed feature space, a large number of classes increases the probability of occlusion or intersection among the class segments. The smaller the vocabulary size, the more accurate and faster the recognition. In addition to that, the complexity of grammar and the length of vocabularies are inversely proportional to accuracy.

- Environment: During the generation of a signal or after a signal leaves the source, the environment in which the source is located, can distort the signal. It is not practical to provide a noise free environment. Hence, a filtering operation has to be applied over signals prior to use.

- Recognition Rate: Since humans recognize these signals with a 99.2 % accuracy, these recognition systems should also recognize at this rate in order to interact with humans without a problem.

Note that the issue of the start/end of classes in speech and handwriting datasets is not challenging as much as in the gesture recognition domain. In speech and handwriting, silence and space between utterances and words can be utilised in a way to spot the start and end of gestures. But transitions data between gestures in gesture recognition is more complicated.

2.7 Complexity of The Problem (Dataset)

The complexity of a problem, or dataset, analyses the similarity between class models and the samples data and class models. The complexity of a problem or dataset is strongly related to the degree of linear separability of the classes in the dataset. All temporal problems have not the same degree of complexity. As mentioned earlier, issues such as noise, intra/inter temporal-spatial variance are some of the main factors determining the complexity of a temporal pattern recognition problem. Although, some of these factors can be eliminated by using some restrictions (for example, a noise-free environment in the speech recognition domain), some domains are still too sensitive to these factors. However, these factors still have to be taken into account during the estimation of complexity.

These factors create two interactive complexities: spatial and temporal complexity. By interactive complexity, it is meant that, these factors interact with each other and catalyse or uncatalyse the temporal and spatial complexity. For example, in the domain of gesture recognition, the spatial and temporal complexity of arm-based gestures would be high due to inter personal variance (small and tall person). In many domains (for example handwriting recognition), temporal variance, in particular, could contribute to spatial complexity.

Although these two complexities are correlated to each other, and it is a challenging task, each of these complexities has to be analysed independently in order to obtain an accurate complexity analysis. Intra and inter spatial complexities of channels in a class model could be used for spatial complexity of the class. In a similar way, the temporal complexity of a class can be estimated by using samples of the class models with reference to their class model. And finally, the aggregated result of spatial and temporal complexity of a class can be used to analyse the complexity of a dataset or problem.

There are several approaches to characterise and measure the complexity of problems. These techniques include entropy analysis; Chi-Square, skewness and kurtosis analysis; linear discriminant analysis; principal component analysis; intersection volume analysis and temporal analysis. These techniques will be elaborated upon in Chapter 4 Analysing and Modelling.

In addition to the complexity of class models and its training instance, the complexity of the test data namely, complexity of band B, is also important to evaluate the success of the proposed algorithm. The complexity of a band can be estimated using the same approach which was applied to the class model. But, unlike the class model, the band also accommodates the complexity of the grammar which regulates the band.

2.8 Accuracy of the System

The recognition rate of a proposed system for a given problem can be evaluated using two different approaches. In the first approach, called isolated recognition, all classes in the band B are assumed isolated and segmented. In other words, the *start* and *end* points of the class are known in advance by the recognition algorithm. In the second approach, referred to as continuous recognition, the band is assumed as a continuum of two or more gestures. No prior knowledge of the start/end points of the gestures is known. Therefore, the recognition algorithm has to discover these points as well. In the literature, isolated and continuous recognition task are also referred to as *weak* and *strong* recognition respectively.

2.8.1 Isolated Gesture Recognition

In the isolated recognition, it is assumed that the *start/end* points of test classes ($\widetilde{C}_i$) in the band B are known in advance. At each *start* point of the test classes, the recognition algorithm reset its internal state for a new recognition. Similarly, at the *end* point of each class, a necessary operation is performed to see if the gesture has been recognized. The recognition resulting at the *end* point is used for evaluation. In the case of isolated gesture, the accuracy of the system can be measured by the ratio of correct recognition of class to total number of defined classes in the band. This is also referred to as word recognition in literature.

$$Rec_{iso} = \frac{Total\ \#\ CorrectRecognition}{Total\ \#\ Class\ in\ the\ Band,\ B} \tag{2.3}$$

Generally, the band, in the case of isolated recognition task, consists of testing classes which are organized according to some testing scheme such as cross validation and its variations such as k-Fold and leave-one-out cross validation.

2.8.2 Continuous Gesture Recognition

Unlike isolated recognition, in continuous recognition, the *start* and *end* points of each class on B are not known. In addition, the band can consist of undefined classes C_{NoN}. Since the start/end points of the classes are not known, the proposed algorithm has to detect approximately the *start* and *end* points of the classes automatically.

The accuracy of a proposed system involving continuous gestures can be measured in two ways. The first approach is similar to isolated recognition. Here, accuracy is measured by the word recognition such as in equation 2.3. In the case of continuous gestures, wrong recognition can be decomposed into substitution, deletion and insertion errors. Word recognition can be also written as follows [146]:

$$Rec_{cont_1} = \frac{N - S - D - I}{N} \tag{2.4}$$

where

- N=total number of classes in the band B
- S= Substitution error when the system incorrectly classifies a class.
- D=Deletion errors arise when the system fails to recognize a class
- I=Insertion error occurs if a recognition of a class is made and the class is not in the continuous test data.

The second approach is based on frame recognition. The task involves recognizing the class of frames on the band rather then recognition of segments or classes in the band. That scheme can be useful, in the case of evaluating the long non-defined class C_{NoN} on the band. Actually, frame recognition corresponds to the rate of correct prediction

$$Rec_{contWord} = \frac{Total\ \#\ CorrectFrameRecognition}{Total\ \#\ Frames\ on\ the\ Band,\ B}$$

For validating our system, the first approach based on word recognition, is used along side sentence recognition rate. A sentence is considered correctly recognized if

and only if all the classes in the sentence are recognized in right order and without substitution, deletion and insertion error.

$$Rec_{sent} = \frac{Total\ \#\ Correct\ Sentence\ Recognition}{Total\ \#\ Sentences\ on\ the\ Band,\ B} \tag{2.5}$$

In the case of long sentences, sentence recognition rate could be low if even word recognition in the system is high. The longer the sentence, the more likely that the sentence consists of one of the misrecognized classes in the sentence. Therefore, word and sentence recognition should be considered together.

2.9 Summary

The purpose of this study is to recognize FDO gestures in an on-line mode for training purposes. FDO gestures are a type of semiotic/iconic, special sign language. FDO gestures are either dynamic or static or hybrid. 18 FDO gestures are subjected to investigation, which can be readily extended to the complete FDO gesture set.

Gesture recognition problem is a temporal pattern recognition problem. It has similarities with other temporal pattern recognition problems such as handwriting and speech recognition. Therefore, methodology, modelling/analysing techniques and recognition algorithms, experimental designs used in these areas can be utilised for gesture recognition. Speech recognition researches, in particular, have achieved viable results for practical applications.

Like other temporal pattern recognition problems, gesture recognition problem has also to deal with the issues of intra/inter personal spatial and temporal variance, start/end detection, repeatability and connectivity. Gesture recognition must be person (source) independent used on-line and a continuous manner. Therefore, the recognition system should be able to detect the start/end points of gestures in the continuous gestures stream (band). FDO gesture recognition has an advantage over other temporal pattern recognition domains. Unlike other domains, in FDO gesture recognition, since training of FDO is performed in a special training room, the environment can be arranged to reduce unwanted factors such as noise and background and strengthen other useful features.

Since FDO plays an important role for the landing and recovery of planes with pilots. Hence, FDO must get an excellent training for the safety of both the pilots and planes. Therefore, recognition errors of FDO gestures must be as low as possible. In addition, the system must be able to give timely feedback for training.

An analysis on the complexity of the FDO gesture dataset can clarify and help to comprehend the problem. Therefore, prior to developing a recognition system, a systematic approach is needed to analyse the correlated temporal and spatial complexity of a temporal dataset.

The accuracy of the system is measured either by word or sentence recognition rate, in the case of continuous gesture recognition. In the isolated case, only a word recognition scheme is deployed. Word recognition rate corresponds to correct recognition which excludes substitution, deletion and insertion rates. In the case of long sentences and low word recognition rate, sentence and word recognition rate should be considered together.

Chapter 3

Literature Review

"Not to watch a person's mouth but his fists."
Martin Luther

Gestures are an important means for communication either used independently or in conjunction with other communication methods. Although using gesture as a means of communication dates back to the emergence of natural language, [90], research into gestural study started in the 19th century and has showed significant acceleration in recent decades. In the light of this research this chapter is an overview of gestural study therby making further chapters easier to understand.

Despite enormous efforts, construction of a common theoretical background to gestural study has not yet been achieved. Hence, each discipline has built its own theoretical basis and boundaries in view of its domain and functions. For example, in painting, gesture is a sketch used to block layout of a composition, but in linguistics; it means a medium to convey information. And, no doubt, from point of computer science it also has its own characteristic meaning. Therefore, our exploration into the gesture world starts with a definition, a taxonomy, and properties of gesture in perspective of major discipline in order to clarify the concepts. (This is followed by a reasons why we are interested in gesture with reference to computer science.)

With regards to computer science, gestural study or in other words gesture recognition, is a sub area of human-computer interaction (HCI). The motivation behind gesture recognition research is to develop methods to recognize gestures to interact with the environment efficiently and naturally. The enormous usage of computers in our daily lives has greatly increased the importance of HCI. As a result of this, gesture recognition research has accelerated over the years and several systems and methods are in use for recognizing gestures. The section following gesture concepts (definition, taxonomy, properties) describes the structure of a general gesture recognition system.

In this study, the gesture recognition problem is also subdivided into four parts just like the general gesture recognition system: modelling gesture, perceiving the performing gesture (gesture acquisition), analysing the acquired performed/performing gesture and recognition of gesture.

Modelling gesture is the first part and is the backbone of the recognition system. Modelling gesture is the process of expressing gesture in mathematical form using gesture information such as spatial, temporal and contextual. Gesture models, which can be defined either explicitly or implicitly in the system, affect performance of the recognition task and naturalness of the system.

Perceiving the performing gesture is the process of acquiring gesture information by a sensor device. According to the requirements of the recognition system sensor device can be a tracker-based device such as gloves, body suits or a vision-based device which is a non-contacted gesture source such as a video camera. Note that tracker-based gesture sources are often cumbersome.

Once the gesture data is acquired the next part is to compute the parameter or feature of the performed (off-line) or performing (on-line) gesture. This process consists of smoothing, normalization and extracting data from the acquired raw data and getting the necessary information such as some part of it (segmentation and localization) to construct the parameter set or feature vector.

The recognition of gesture part is responsible for deciding the computed parameters corresponding to that of a modelled gesture. This process is based on distinguishing and classifying parameters. This means that the gesture recognition problem can be reduced to a pattern recognition problem. Pattern recognition can be considered as representations of pattern (gesture modelling and analysing) and decision making [143]. Therefore, our decision making process highly depends on the previous parts, data acquisition, gesture modelling and analysis. In addition to describing the general pattern recognition concept, various pattern recognition techniques used in gesture recognition research with example studies are included in detail in this chapter.

Not only gestures, but also other applications, such as speech, handwriting, and facial and bodily expressions, used by humans to interact with each other have been shown to have migrated into HCI. A common thread in all these applications is that they all share similar challenges in the pattern recognition with varying degrees and emphasis. Briefly mentioning these issues in the next sections, will help us to comprehend the problem in detail.

3.1 Gesture Recognition System

A gesture Recognition System is a computational system in which a user's gestures, which are perceived by external sensor devices, are recognized by way of interacting naturally and directly with the computer. A gesture recognition system should have the following criteria in order to achieve reasonable outputs [133]:

- Natural and Direct Interaction: Since, gestures are used as a communication medium amongst people, the user should be able to interact with the system directly and naturally like gesturing with someone else. Type of gestures allowed and their effect on the system should be known by the user. When a new language is created, attention should be paid to make it as easy, understandable and intuitive as possible. In addition to that, it should be kept in mind, the more distinguishable the gestures, the easier it is to classify and recognize them.

- Feedback: The system should send a response back to the user explicitly stating whether the gestures have been recognized. Frequency of the response should not be so much as to burden the system or so little to keep the user waiting for too long.

- Multi-purpose: All the advantages of gesture should be evaluated. Not just those used for substituting conventional input devices as such the mouse and keyboard, but also those used for other operations such as navigation, pointing, and manipulating the environment. On the other hand, extraneous usability of gestures should be avoided. Conventional input devices are preferred.

- Technological trade-off: Each technology used has its cons and pros. Thus, the benefit and limitation of each technology should be considered. For example, if you need accurate finger data, glove-based sensors should be chosen rather than vision-based sensors.

- Flexibility: Because of the inter-person and intra-person variance, a gesture can be performed in several ways. This variance can be a spatial variance or temporal variance. Hence, the system should take into account variability in gesture performance.

- Fatigue: For a performing gesture, more muscular effort is needed than using a traditional input device. Gestures in terms of performing difficulties should be avoided. Gesture should be concise and simple in order to minimize effort and maximize performance.

- Intention Detection: The system should be well defined to detect the intention of gestures. The sensor device collects not only the desired gestures but also captures unintentional or undefined movement.

- Testable: Usability, reliability and accuracy of the system should be tested. The system should be easily integrated into other systems. Similarly, according to the domain, reliability and accuracy are also an important issues. For example, in a medical domain, a gesture recognition system will be more important, than in a game or training domain.

3.2 Gesture Recognition System Components

For solving a problem, which occurs in daily life, it's useful to consider how the similar problems are solved in real life. Imitation of similar problems' process or solution may help us solve it. Unfortunately, when it comes to gesture recognition, we don't have such a luxury. It has not yet been understood how people can recognize gestures in such a flexible, successful way. In literature, most of the gesture recognition researchers took inspiration from human recognition systems for example - speech, handwriting, face and emotion recognition.

But how does a human recognize a gesture or object? In order to explain the human recognition system, let us take the example of the recognition of letters. The human recognition system can recognize letters within a fraction of a second. It does not matter whether the letter is in upper, lower case, or has a different font, size, or position.

To recognize a letter it first has to be perceived. This can be done in several ways; for example by using a vision system like most people do or touching like blind people do. But perceiving letters is not enough to recognize them. Each letters' abstract or idea of it must be formed, modelled in some form and stored in the brain in advance. When a letter is perceived, it is analysed and modelled in such a way that the abstraction and idea of the letter are formed. Finally the abstract, idea of the letter and the formed model are compared by using some recognition algorithm [31].

The approach can be applied for the recognition of gestures in a similar way: First of all, gestures must be perceived by either using camera-like eyes,(vision system) or tracker and gloves, like touching. Abstracts or ideas of gestures must be represented in some form and stored for using later as is in letter recognition. And then acquired data must be analysed to form the data in the same form and the means by which abstract gestures are represented. And finally, the formed model and abstract gestures

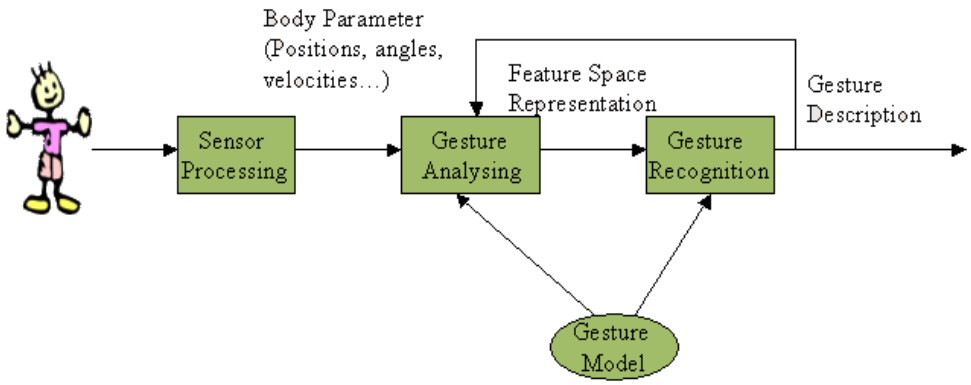


Figure 3.1: Component of a Gesture Recognition System

must be compared. As can be seen from the example in Figure 3.1, gesture recognition systems can be considered as a combination of the following components (Figure 3.1):

- Sensor Processing: Acquiring gestures in some parametric form (Vision-based or tracker-based)
- Gesture Model: Constructing abstract, idea of gestures
- Gesture Analysing: Analysing and forming data in order to represent it as abstract, ideal gestures.
- Gesture Recognition: Comparing, classifying abstract/ideal of gestures with formed data by using same pattern recognition methods.

In addition to the above components, especially in the case of sequential, temporal pattern recognition system, an extra component, grammar or language is also deployed. The grammar component utilizes prior knowledge about sequential statistics and rules in the recognition part to boost performance.

Approaching the background of the gesture recognition system from this perspective helps us to comprehend the proposed solution in this thesis, which is structured based on this approach. In the following sections, each component of the system will be described in detail.

3.2.1 Sensor Processing

Sensor processing is the processes of acquiring the key gesture parameters such as position, orientation, velocity and quantity. These acquired parameters are same parameter, which are used to construct abstract gestures. In literature, two major schemes are

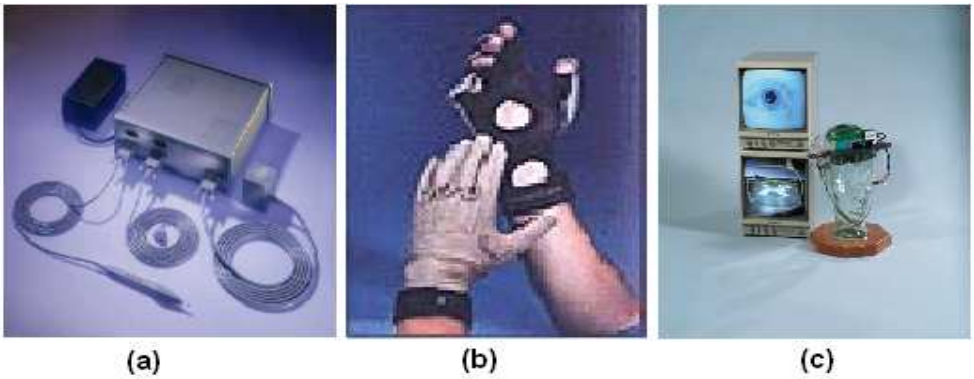


Figure 3.2: Tracker/Glove Based Sensors: (a) Polhemus FASTRAK, (b) CyberGlove, (c) Head and Eyes Tracker

used for sensor processing. Due to its naturalness, the preferred one is the vision-based approach. The second one is tracker/gloves based. Both have their own advantages and disadvantages. These pros and cons and expectations of the system should be borne in mind when choosing which sensor to use.

3.2.1.1 Tracker and Glove Based

Tracker and gloves-based sensors are used to capture the precise physical parameters such as orientation, velocity, and position. These devices are generally attached to the gesture's sources. In figure 3.2, some common trackers for gestures are shown. Tracker and glove-based sensors have the following advantages and disadvantages.

Advantages

- Direct measurement: Since this sensor is directly attached to the gesture's source, it provides just its parameters (joint angles, 3D spatial information, wrist rotation).
- Accurateness: As a result of direct measurement, precise data is acquired.
- High Frequency: This kind of devices provide high sample frequency. For example, Polhemus FASTRAK's frequency is about 120 samples per second.
- Translation-independent: Since these devices are mostly designed for a specific domain, especially gloves, these devices generally provide exact features, which are used for modelling gestures. There is no need to translate the acquired data and as a result it is ready to use immediately. As a consequence of that the system performance increases. Therefore, for a real time gesture recognition

system, tracker and glove-based sensors are generally preferred, compared to computer vision which involves segmentation.

Disadvantages

- Calibration: These are specific, complex devices. Hence, calibration or programming of these devices can be difficult.
- Expensive: As a result of developing for specific purposes, these devices are expensive.
- Cumbersome: To attach the sensor device to the gesture source causes uncomfortableness and reduces the range of motions.
- Unnaturalness: Wearing a glove or attaching a tracker limits the naturalness of the system.

It is worth noting that some of these disadvantages have been tackled by new technological advances. For example, the inertial measurement unit (IMU) designed by MIT not only reduces these disadvantages, but also provides an embodied gesture recognition system [4].

3.2.1.2 Vision Based

The most significant disadvantage of tracker and glove-based sensors is that they are cumbersome. To overcome this disadvantage, vision-based sensors are recommended. Vision-based sensors are inspired from the human vision system.

In the vision-based approach, one or more cameras are used to capture the gestures in various orientations. Typically, cameras are set at a fixed position even though they may be located on a moving platform. Both colour and monochrome cameras can be used depending on the recognition algorithm. Not just in gesture recognition, but also in other research areas, (face recognition, analysing facial expression, interpreting human activity) the vision-based approach is used [87]. Thus, in gesture recognition research, the results of these other research areas can be used.

The frame rate (fps) of cameras depends on what kind of gestures are examined and the expected recognition rate (real-time or not) of the system. If static gestures are used, the frame rate should not be that high (5-10). Whereas, if gestures, which have high temporal activity, are used, a higher frame rate is required (25 or more). In addition to that, the recognition rate determines the sensor frame rate. If a real time response is desired, the recognition rate should be high. One should not forget that

the higher the frame rate, the more analysis and recognition processing needs to/can be done. This can affect the performance of the system [122, 123].

The most significant disadvantage of the vision-based approach is the occlusion problem. The occlusion problem arises when the object under consideration is concealed by obstacles. Furthermore, self-occlusion prevents a full view of the arms, hands, and fingers. To overcome this problem, multiple cameras can be used. But this approach adds synchronization concerns. Apart from this issue, vision-based sensors have the following issues to contend with:

- Background:Cameras not only capture the image of the gesture source but also other things in the environment. Thus, distinguishing and extracting the gesture source from other things, and the background, overloads the system.
- Lighting conditions: Variation in lights affects the parameters captured from the environment.
- Dimension: Does the system construct a three-dimensional or two-dimensional model of the gesture source? For a three dimensional model, more than one camera must be used.
- Representation of time: How are temporal features of gestures presented?

When it comes to the advantages and disadvantages of a vision-based system, it can be said that they are in reverse of the tracker and glove-based system. But the disadvantages of vision-based sensors have been reduced with latest advances in technology. In summary, the following can be given as advantages and disadvantages of vision-based systems:

Advantages

- More natural
- Excellent body and hand tracking
- Versatility
- Comfortable
- Cheap

Disadvantages

- Low accuracy in pose determination

- Translation-dependent
- Sensitivity lighting conditions

3.2.2 Gesture Modeling

Gesture modelling is the process of representing gestures by a set of parameters. These parameters are determined based on the gesture's characteristic properties such as spatial and temporal.

As figure 3.1 depicts, this part of the gesture recognition system is the backbone of the system and is closely related to other parts: analysis and recognition. The modelling part follows the analysis part which determines the parameters and features and how they are used. The pattern recognition process in the recognition part obtains gesture class from the model part. Therefore, the gesture modelling part plays an important role in gesture recognition systems.

3.2.2.1 Temporal Modeling of Gesture

Human gestures are not only static gestures but also dynamic. Therefore, their temporal characteristic should also be considered when a dynamic gesture is modelled. In addition to these, Hummels and Stappers proposed that, "affect" properties of a gesture can also be used as auxiliary information for modelling a gesture [49]. An affect property indicates emotional quality of a gesture.

For modelling temporal characteristics of a gesture, results of psychological research can be utilised. In the psychology area, research has been conducted independently by several authors. But results of these studies are generally in agreement with each other. For example, according to Kendon [59], each gesture has some characteristics to distinguish it from unintentional movements. These characteristics are:

- Excursions: Each gesture has a starting and an end point, which are generally the same.
- Peak Structure or Stroke: All gestures have a centre where the gesture is performed.
- Well Boundedness: Every gesture is performed between the start and the end points.
- Symmetry: As result of peak structure and excursions, gestures have symmetric actions.

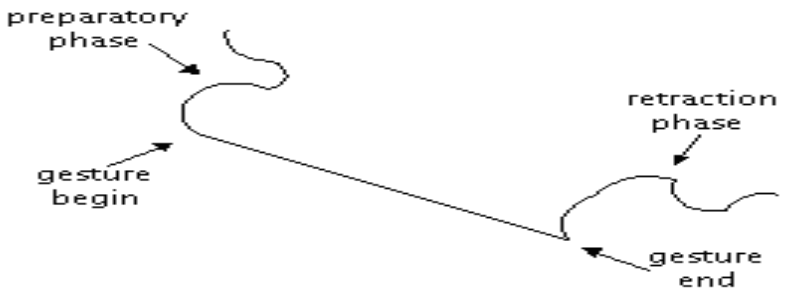


Figure 3.3: Phases of a Dynamic Gesture [16]

McNeill and Kendon defined three phases, which constitute a dynamic gesture: preparations (pre-stroke), stroke, retraction (post-stroke) (Fig. 3.3). Furthermore, Quek [92] proposed a set of rules (similar to Kindon's) to formalize how a dynamic gesture is performed. This set includes the following six rules:

- Gestures are contained in movements that start with a slow initial movement from the rest position, then continue with a phase with substantially increased speed (the stroke), and end by returning to the rest position.
- The hand assumes a particular configuration during the stroke
- Slow motion between resting positions are not gestures.
- Hand gestures should be constrained within a certain volume- workspace
- Static hand gestures require a finite period of time to be recognized
- Repetitive gesture can be gestures.

3.2.2.2 Spatial Modeling of Gesture

Since gestures are preformed in 3D space, its spatial aspects have an important role in modelling gestures. In literature, spatial characteristic of gestures are modelled by three methods: Tracker/glove based, 3D Hand model based, and Appearance based. (Fig. 3.4) [47]. As it can be understood from its name, tracker/glove based modelling is used when a tracker or glove sensor is used whereas the last two methods are used by vision-based sensors.

- Tracker/Glove Based: The model of a hand/arm is constructed by output capacity of tracker or gloves. For more sophisticated models, advanced sensor devices can be developed.

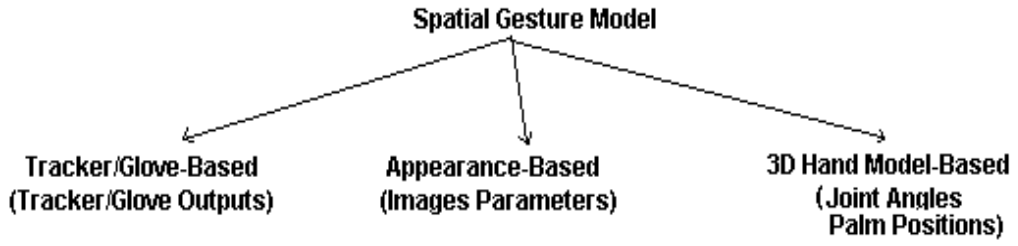


Figure 3.4: Spatial Gesture Modelling

- 3D Hand Model Based: By using the joint angles and palm parameter of the hand, a 3D model of a hand is built. The main idea behind this method is that the model is compared to images of a hand perceived by cameras, and then the parameters (joint angles, palm parameter) are estimated. Then, the parameters are used in recognition part. Due to the huge number of joint angles and palm orientations, the computational time of this method is high. To tackle this problem, marked gloves are adopted [122, 123].

- Appearance based: Instead of directly deriving model parameters from a 3D spatial description of a hand, gestures are modelled by extracting some of the properties that are associated with the images of hand postures. These image properties are: geometric, motion, fingertip position and motion. Each gesture template is represented in terms of these parameters. This is the most accepted and widely used technique.

3.2.3 Gesture Analysis

This part is responsible for obtaining data from the sensor part, processing the raw data to detect and extract features, constructing model parameters, building the model by using the constructed parameters and finally sending constructed model to the recognition part. Gesture analysis can be summarized under two titles: detecting and extracting features, and the building model.

3.2.3.1 Feature Detection and Extraction

This part is responsible for the detection of features, which are used to construct parameters, in associating a further meaning to the constructed model.

Feature selection and extraction is the core of all pattern recognition systems. Performance of the classification is highly dependent on these tasks. Although, in literature

these two concepts are used interchangeably, they refer to two different concepts: Feature selection refers to algorithms which identify and select (hopefully) the best subset of a feature set which performs the best for the recognition task. On the other hand, feature extraction refers to the transformation or combination of new features from existing features [52]. Both methods aim to reduce the dimensionality of the feature set which is important for cost of measurement and classification performance.

In literature, the curse of dimensionality and peaking phenomenon are well known concepts in statistical pattern recognition which deal with the number of features, samples size and classifier performance. The curse of dimensionality refers to exponential growth with dimensionality in the number of examples required to accurately estimate a function. The curse of dimensionality leads to the peaking phenomenon which states that, increasing the number of features does not necessarily increase the performance of the classifier. Actually, it is observed that paradoxically it decreases the performance. Therefore, there has to be a balance between the number of features and the sample size. In literature, taking a sample size as at least ten times the feature number is considered good practice [52].

Depending on which type of sensor is selected the detection and extraction process varies. This process is not complicated, if a tracker or glove-based system is used. On the other hand, if a vision-based sensor is selected, some problems are encountered.

One of the challenging issues of a vision-based system is the localization of the gesture source. In order to extract features, firstly the gesture source, hand/arm, must be found. This can be done using either colour cues or motion cues techniques. The colour cues technique uses human skin as a distinct feature to distinguish the gesture source from the rest of the images. The major disadvantage of this technique is that the human skin colour changes in different lighting conditions. To tackle this various solutions (restrictive background and cloths, scale filtering, positional filtering) are proposed such as in [122, 123]. The motion cues technique assumes that the background is usually stationary with respect to the gesture. Thus, the gesture source, hand/arm, can be located by using the motion of the gesture source. For example, the study [101, 29] utilises the difference matrix, motion or gradient, between two consecutive image frames to construct the feature vector. But, in this study, in the case of the vision-based system, distinguishable marks (light sticks) in the hands are used to reduce the complexity of hand segmentation.

Selecting the gesture features is crucial to the system. The features should be as distinctive and compact as possible to distinguish the gestures. In addition to these, the features must be robust to noise. Fingertips, colour, contours, hand and arm silhouettes, hand/arm/finger directions, textures, first or second order gradients,

angular velocity, energy, ceptral features are all examples of some commonly used features in the literature.

3.2.3.2 Model Construction

After detecting and extracting the features, the last phase of gesture analysis, namely model construction, is processed. In this part, model parameters are computed in order to construct the model. The model is constructed in the same way in which an ideal, abstract model is created. The constructed model is sent to the recognition part in order to make the recognition based on predefined models.

3.2.4 Gesture Recognition

Gesture recognition is the process of classifying, recognizing the constructed model, which is acquired from the analysis part, among the predefined modelled gestures as a specific gesture. The feature vector, in other words, the constructed model, is the input for the process. The definition of the feature vector is crucial to the recognition process. For a successful recognition, the feature vector should be as expressive and distinctive as possible. The performance of the recognition process depends on the gesture analysis and modelling parts. For a comprehensive discussion on static pattern recognition, readers are referred to [8, 52].

Various methods have been employed to recognize gestures in literature. The types of methods depends on what kind of recognition is required: dynamic, static, real time, and off-line. Although, in literature, several techniques have been used to recognize gesture, here, three of the most important techniques are reviewed.

3.2.4.1 Neural Network

Neural networks (NN) have been used not only in pattern recognition domain, but also in a wide spectrum of application domains. A neural network is inspired from our current understanding of animal and human brain function. Even though, there is not a consensus definition of NN, it has the following common properties. Unlike traditional von Neumann machines, NN is a different computing paradigm which is capable of modeling very complex non-linear functions. NN is an inherently parallel processing architecture with highly interconnected basic multiple processors (neuron). Messages between neurons are simple signals and the system has the ability to learn or adapt its parameters from the observations [8]. In this study, a brief introduction to NN for spatial and temporal pattern recognition will be presented. For further detailed

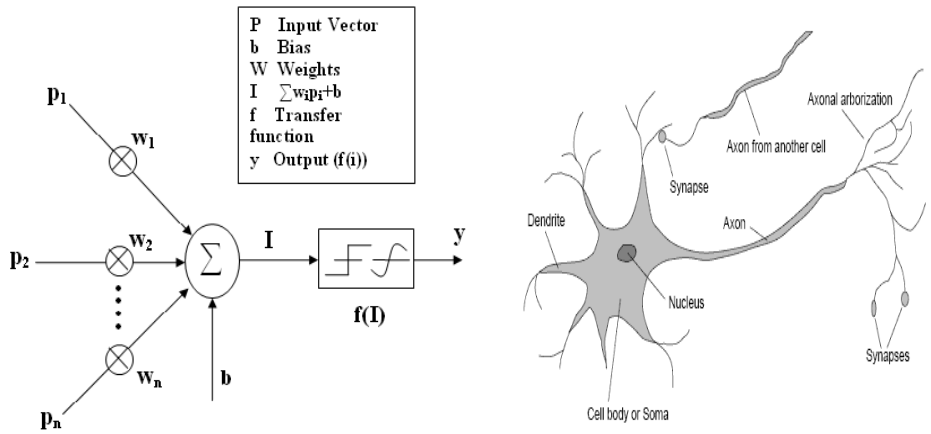


Figure 3.5: Structure of a common artificial and inspired biological neuron. Since the learning process in biological neurons is still a myth, various artificial neuron network architectures have been constructed based on this common artificial design.

information on NN and pattern recognition, the readers are referred to [8].

Although, wonders of how brain works goes thousands years back, latest advances in technology and science have made it possible to study the structure of the brain more comprehensively. Warren McCulloch, in 1943 and 1947 published two seminal papers on the subject. McCulloch demonstrated that simple neural circuits with computational capability and a Turing machine program could be implemented in a finite NN. These papers aroused huge interest in NN and consequently in the 1950s that lots of research was conducted on NN. But due to theoretical short comings of NN (published in 1969 by Minsky and Papert) and practical problems in technology, NN research did not produce satisfactory results. This led to a temporary halt in NN research for a couple of decades. But in 1974, a paper on backpropagation on the multilayer perceptron network (MLP) by Werbos [144] and another paper in 1981, associative neural networks by John J. Hopfield, renewed interest in NN. Hopfield successfully showed the capacity of a NN with a clear theory. Since 1981, NN has been subject of various studies in various domains. For a comprehensive history of NN, the reader is referred to [78].

In literature, a wide range of NN are proposed for different tasks. While some of them are in agreement to a certain extent with biological systems in terms of structure and process, some others are far beyond a designed connectionist computational system. Figure 3.5 illustrates a basic and common neural network intra structure in an artificial and biological neuron. Figures 3.6 and 3.8 show how neurons are connected in different topologies such as a feedforward multi level and recurrent neural network.

These topologies will be elaborated in detail in forthcoming sections. A common neural network consists of the following major components:

- Inputs (P) : It corresponds to the inputs on any layer either an input (real data source) or hidden layer (from other layers). The input layer serves to introduce value to the network. The larger the set of inputs, the bigger the neural network and training observation for modeling and approximation tasks. Bias (b) is a special input which is introduced for fitting the input data to the target function more efficiently.

- Weights (W): are the adjustable parameters representing the importance of input in the system. Each input is associated with a weight. With analogy to synaptic strengths in biological neurons, weights are dynamically adjustable to achieve the desired network behaviour. The adaptation or estimating of optimal weights is the core of the learning process to model and approximate the non-linear input/output relationship. Weights are updated according to networks topology, training set and learning rules.

- Summation Function($I = \sum$): Responsible for accumulation of weighted inputs by calculating the dot product of the weights and input vector (WP). Namely, it is the sum product of the input and their weights. Note that, in terms of geometrical representation, when two vectors have same the direction, the dot product of two vectors produces maximum value.

  In some applications, a different summation function can be used instead of the sum product. Various fuzzy aggregation operators, such as, minimum,,maximum, average, product, are also employed in the literature[36]. The type of summation operator relies on the problem, and architecture.

- Transfer (Activation) Function ($f(i)$): The result of the summation function (I), then, is filtered through a simple linear or non-linear transfer function. Generally, a threshold mechanism is used in the transfer function to determine the result of the neuron. If I is greater than the threshold value, the neuron will be fired or activated by emitting a value as the neuron is output. The value in most cases is not proportional to I. In another words, transfer functions are generally non-linear to accommodate any complex relationships between the inputs and the desired outputs.

  In literature, a variety of linear and non-linear transfer functions are used. Hard limiter (step), ramping, sigmoid and gaussian are some typical examples. In

a step function, the output takes two values, either 0 (-1) or 1. In a ramping function, for a band width, the output is linear, but for the outside of the band, it behaves like the step function. In the case of a sigmoid function, the output is in S shape between 0 and 1 or -1 and 1 (hyperbolic tangent). Since both the sigmoid function and its derivative are continuous and non-linear, it is preferred over other transfer functions. Note, while it is possible to use a different activation function for each neuron, the general trend is to use the same activation function for all neurons in the network.

In some domains, normal or uniform noise and scaling/limitation are applied before or after transfer functions. Noise is added to a summation function to obtain a more brain like transfer function. Scaling and limitation are applied to guarantee the result is kept in a desired range.

- Output Function(d): It corresponds to the output of the neurons. While in most cases, the outcome function is equivalent to the result of the transfer function, in some topologies, the result is modified to accommodate competition among neurons in the same layer. Competition is employed to determine the contribution of each neuron in the same layer for learning and adaptation process and input strength for the next layer.

- Learning Function, Cost (Error) Function: Learning in neuron network terminology refers to the adjusting of weight factors of each neuron according to a cost function (C) and some algorithms. The cost function $C : F \rightarrow \Re$ measures the distance between current solution (f) and the optimal solution (f^*) of the given problem $C(f^*) \leq C(f) \forall f \in F$. The learning algorithm adjusts weights systematically in order to reduce the cost function. The cost function is also referred to as an error function in the literature.

  In most cases, the cost function utilises the differences between the result of an output function(d) and target output. For a finite dataset, the cost function can be written as :

$$C = e(x) = \frac{1}{N} \sum_{i=1}^{N} |f(x) - t(x)|^2 \tag{3.1}$$

Learning algorithms in NN are mostly derived from optimization theory and statistical estimation. In training of an NN, back propagated gradient descent

based schemes are generally used. The error is back propagated to the previous layer after scaling with the derivative of the transfer function. Then, the weights in the layer are updated according to the back propagated scaled error.

There are two general learning paradigms: supervised and unsupervised learning for different domains.

Supervised Learning: This paradigm involves a teacher who assesses the performance of the network by utilising the error between prior given targets and outputs of the network. Namely, the network learns from its "mistakes". The cost function is minimised by approaching the optimal solution f^* through the guidance of the teacher. The mean squared error is commonly used to measure the error between the network's output and target value. For example, the Radial Basis Function and Multi-layer Perceptron neural networks employ supervised learning. Supervised learning is especially useful for pattern recognition and function approximation problems. For temporal pattern recognition problems, such as gesture and speech recognition, supervised NNs are mostly used [74].

Unsupervised Learning: Unlike supervised learning, no teacher is employed in this scheme. Networks learn from samples by minimising the given cost function. Therefore, it is referred to as self-organisation. It discovers and describes the underlying structure of the samples. Hebbian rule and the competitive rule are examples of unsupervised learning. The main idea behind a Hebbian rule is that, the strength of weights between two neurons are proportional to the amount of the interactions between them. If a neuron stimulates the next neuron, quite often the weight between them is increased proportionately. Similarly, in competitive learning, the associated weights are updated according to the popularity of the group or clusters. Unsupervised learning is the mostly used for clustering, vector quantization, and estimation problems. In the case of clustering, unsupervised-based network's output has to be interpreted by the user, because the network does not label the discovered clusters.

The major advantages and disadvantages of neural networks can be outlined as follows:

Advantages

- Learn from Examples: NN has capabilities to learn and generalise from samples. Therefore, it does not assume an underlying statistical sample distribution or

modeling, which in many real world problems, are non-trivial. The trained neural network itself is the representative model of the samples.

- Dynamic Structure: Unlike a rule-based and programmed system, NN can change its internal structure to adapt to new environmental changes in an on-line manner.

- Powerful Modelling: Provided a sufficient amount of samples are used, NN can model non-linear, multivariate complex problems [58, 12, 86].

- Performance: The time and space complexity of a trained NN is low compared to other paradigms. It does not require huge space and processor resources for computation since each unit, i.e neuron, does a simple process. Even on a personal computer, a trained NN could perform a complex analysis. In addition to these, by their very distributed structures, NNs are well suited for parallel computation over either software or hardware systems.

Contrary to all the above benefits, NN has unfortunately the following disadvantages :

- Black Box: Since the samples are modelled in weights, it is difficult to interpret all underlying representation. It would even be problematic to extract human comprehensible rules. A trained NN behaves like a black box - the NN produces good results, but lacks the ability of explaining how to produce the results.
  In cases where non-linear transition functions have been used, black box behaviour is observed more. For example, because of non - linear transition functions and cyclic paths (unlike feedforward paths from the input layer to the output layer such as in multilayer networks), in large recurrent neuron networks RNN, chaotic behaviour emerges which makes it non-trivial to analyse RNN, which is one of the main types of NN used for temporal pattern recognition [111].

- Training : Although a trained NN produces effective and fast results, training of an NN itself is not that trivial. A remarkable amount of consideration has to be paid to samples and NN structure. For an optimal result, prior to feeding samples to NN, samples have to be preprocessed and analysed to find a useful representation. Apart from that, an appropriate architecture, learning paradigm and training scheme have to be chosen. For example, the study [127], on static American Sing Language images with Multi-Layer perceptions with LSM, concludes that for better recognition, remarkable effort and expertise for optimal

parameters and structure are required. In addition sample size also has to be in proportion to the complexity of the problem in order to avoid overfitting or underfitting. The rule of thumb is that there should be approximately 10 times more samples as number of weights in the network. Even though an optimal structure, learning paradigm, parameters and training samples have been set the training time for an NN would take longer time to converge, compared to other conventional paradigms. Exhausted iterative loops are employed to adjust weights for training of NNs.

The black box effect and training issues overshadow the advantages of NN in some domains. Therefore, it is common to see NN as part of a hybrid system. For example in the SLARTI sign recognition system [135], for a comprehensible structure, multi-layer NNs are only used to extract features for further rule-based classification techniques.

Besides these disadvantages, neural networks have the capability to simulate complex non-linear functions. They have been employed for many tasks such as function approximation [46, 86], regression analysis, time series prediction [35], temporal and static pattern recognition [37], clustering [110], fault detection [138, 38], noise reduction [158], decision support systems [66, 54] and data mining [23].

For the above mentioned domains, various kinds of NN types are employed. For pattern recognition tasks, supervised learning based neural networks are deployed. According to the temporal nature of the patterns, two kinds of NNs have been applied for pattern recognition tasks in literature. In the case of non-temporal, static patterns, in which all pattern data are available at each discrete time step, memoryless neuron networks such as multi layer perceptron (MLP) are used. In the second case, temporal patterns require sequential, historic data of the pattern. Therefore, NNs must employ memory schemes to recall historic information in sequence data. Time-Delay Neural Networks (TDNN), Recurrent Neural Networks (RNN) are mostly used for temporal pattern recognition [70] Apart from these, hybrid systems are proposed to overcome the shortcoming of NN by utilising the best of NN and other paradigm and algorithms such as the Hidden Markov Models (HMM) and Dynamic Time Warping (DTW). In the following subsections, these neural networks architectures will be examined in more detail.

Multi-Layer Perceptrons

As its name implies, unlike single layer networks, a multi-layer perceptron neural network consists of multiple layers employing hidden layers between the output and input layer. The input flows from the input to the output layer in one direction, which is

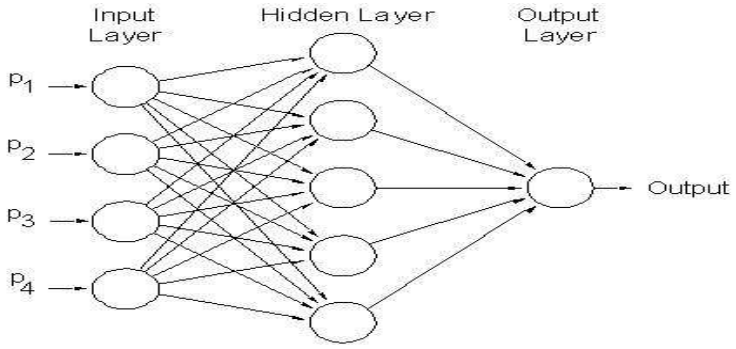


Figure 3.6: A feedforward multi-layer neural network with one hidden layer and one output neuron.

referred to as strictly feedforward. Each layer gets its inputs from the previous layer. For most problems, even though one hidden layer is enough,(occasionally it can be seen), two hidden layers are also used for some problems. But deploying more than two hidden layers can result in getting stuck in local minima. Figure 3.6 illustrates a feedforward multilayer neural network with a hidden layer and an output layer. Note that, for classification tasks, there are more than one output neuron. The following notation is used to represent a L multi-layer perceptron in this section:

$$x^0 \overset{W^1,b^1}{\rightarrow} x^1 \overset{W^2,b^2}{\rightarrow} x^2 \dots \overset{W^L,b^L}{\rightarrow} x^L \tag{3.2}$$

where $l = 0 \dots L$ and $x^l \in R^{n_l}$, W^l and b^l are the n_l dimensional input vector, $n_l \times n_l$ weight matrix and bias at the lth layer respectively. n_l corresponds to the number of inputs in each layer and x^0 is the network input P.

By using the Stone-Weierstrass theorem [124] it has been proved that, provided the large neuron is in the hidden layers, the multi-layer perceptron can approximate any non-linear function. While the hidden to output layer implements a linear discriminant, the hidden layer itself impliments non-linearity. It can be thought of as hidden networks discovering optimal features of input patterns for further linear classification in the hidden - to - output layer. In the gesture recognition domain, MLP is generally used for posture recognition [75, 151].

Training of a multi-layer perceptron is done by a powerful, efficient algorithm which is called backpropagation. In literature, a gradient descent-based back propagation technique is commonly used. It consists of the following four steps:

- Forward Pass: The input vector x^0 passes through all layers to output layer x^L

as follows:

$$x_i^l = f(u_i^l) = f(\sum_{j=1}^{n_{l-1}} W_{ij}^l x_j^{l-1} + b_i^l) \tag{3.3}$$

- Error Computation: By utilising supervised learning, error, the differences between target and actual output x^L is calculated.

$$\delta_i^L = f'(u_i^L)(d_i - x_i^L) \tag{3.4}$$

- Back propagation: Calculated error is then back propagated from output layer to input layer as follows:

$$\delta_j^{l-1} = f'(u_j^{l-1})(\sum_{n_l}^{i=1} \delta_i^l W_{ij}^l) \tag{3.5}$$

- Updates: Weights and biases are updated using the forward and backward passes:

$$\Delta W_{ij}^l = \eta \delta_i^l x_j^{l-1} \tag{3.6}$$
$$\Delta b_i^l = \eta \delta_i^l \tag{3.7}$$

where η stands for learning parameter.

The above algorithms are applied iteratively to obtain an error that is less then a error goal. Training can be done either in a batch or individually or in on-line mode. In batch mode, the average error of all training samples is used to update the weights. In single mode, after each sample of the training set, weights are updated. While batch training is faster and easier for a small dataset, single training is preferred in huge datasets consisting of redundant data [43].

In order to improve performance (free of over or under fitting) of an MLP neural network, the following practical points should be kept in mind when designing and training an MLP network. Initial weight values should be uniformly and randomly distributed around 0. Input values P should be shifted and scaled to have zero mean and unit variance. The learning rate should be small, typically 0.1, for a fast and reliable convergence. For more controlled learning, a momentum or a weight decay parameter (ranges between 0 and 1) can be introduced to prevent overfitting [104]. As a rule of thumb, when deciding the architecture of the network, the number of samples should be 10 times the number of hidden neurons (curse of dimensionality

and peaking phenomenon). But, in order to find the optimum network, some network pruning strategies can be deployed to reduce the network size [128].

Because of MLP's powerful ability to model non-linear functions, it has been commonly used for static gesture recognition in the literature. Moreover, by encoding time information into the input values, it would be possible to recognize dynamic gestures in an off-line manner. For example, in study [41], segmented dynamic gestures are encoded as static gestures by presenting all or partial sequence at a time, in an off-line manner.

In [75], an MLP is used over a benchmark hand static gesture (posture)dataset, Triesch. The dataset consists of 10 static gestures signed by 24 persons. Each person signed 10 samples of each gesture. A vision-based data collection scheme is deployed in three light and background conditions (Light, Dark, Complex). For each posture, an MLP is designed. The minimum and maximum number of hidden neurons and input neurons in MLPs are 50, 150, 400 and 1368, respectively. The study achieves 70 % recognition with MLP, which is far away from the result of other studies conducted using Gabor Wavelets (90 %) [131] and constrained Neural Networks (87 %) [75].

In the paper [33], Fels and Hinton use an MLP network to recognize 66 static gestures of American Sign Language. A glove-based technology is used to acquire 16 dimensional feature vectors. MLP networks consist of 16, 88 and 66 neurons in input, hidden and output layers respectively with a learning rate of 0.01 and momentum of 0.05. The input vector consists of two flex angles each finger (10) and sines and cosines of roll, pitch and yaw oriental angles (6) of the whole hand. The study achieves a 0.58 % and 2.25 % false alarm rate ; 0.47 % and 0.96 miss rate on training and test data. Training and test set size are 8912 and 2178, respectively.

In another study, a subset of static gestures in American Sign Language are again studied [127]. A vision-based approach has been used to capture data, and a local orientation histogram is used as features. A multi-layer perceptron with different number of neurons in the hidden layer with 19 neurons in the input layer is deployed for classification. The study concludes that it has not been able to achieve a robust recognition because of a shortage of expertise in optimal parameter settings and architecture.

The study [151] implements Dynamic Programming (DP) and MLP algorithms to recognize 26 static postures of the American Sign Language. A vision based data collection is implemented and for each posture 20 samples are collected for training. Image histograms are used for feature descriptions. A different number of hidden layers and neurons have been used for MLP experiments. Even though, DP has achieved better results (98.8 %) compared to MLP (96.7 %), because of the advantages MLP has such as memory, computation time and scalability, MLP is preferred to DP.

Temporal Neural Networks

According to the latest study, whatever human being perceive (speech, taste, vision, touch, etc.) and the subsequent processing in the brain (particularly in neocortex) is akin to temporal pattern recognition [42]. Although the perceived inputs have different properties and quantities during the pre-processing phase in the brain, all inputs are transferred to an equivalent representation. Then, in the recognition phrase, a common cortical algorithm, temporal but not a static neural network, is applied to both equivalent input representations and invariant representations of models that are stored in the memory.

Unlike static neural networks, temporal neural networks are designed to memorise historical sequential data. Memory can be accommodated in a feed-forward in two different ways, first of which accommodates a buffer or tapped delay line in the input layer to represent not only the current data but also several sequential data. This scheme stores sequential data in the finite buffer in order to eliminate time dimension of sequential data. In the second scheme, instead of storing historical raw data in the buffer in the input layer, the processed results of the hidden layers are also stored in tapped delay buffers as in time-delay neural networks. These two feedforward schemes suffer from a limited number of buffers in which to store historical data. As a totally different approach, instead of only using a feedforward connection, recurrent links (Rucerrent Neural Networks) are used to memorise historic information in sequential data. In the following section these most commonly used temporal neural networks for gestures (TDNN and RNN) are discussed in detail.

Time-Delay Neural Networks

Time-Delay Neural Networks (TDNN) is mentioned in the domain of speech recognition by Wabel [140]. Unlike tapped delay lines, in addition to the input layer, TDNNs accommodate also tapped delays in hidden layers to utilise the processed information in the hidden layers. Figure 3.7 illustrates a simple TDNN network with one neuron at each layer for the sake of clarity. In literature, generally two hidden layers are used for TDNN architectures.

Generally, a small number of tapped delay buffers are deployed in the input layer to reduce the number of connections from the buffers to the neurons in the first hidden layer. The reduced number of connections between the input layer and first hidden layer contributes to extracting temporally localised features. Once all the buffers in the input layer are filled, the neurons in the first layer are activated and start to fill the buffers in the first hidden layers. Similar to that process, neurons in the second

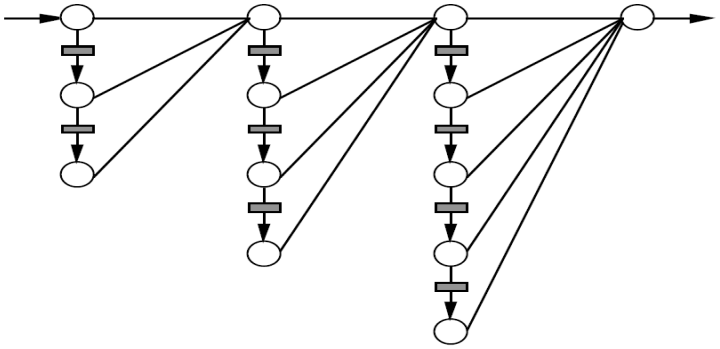


Figure 3.7: A Basic TDNN architecture with one neuron in input, two hidden and output layers. Delay lines are embedded in input lines and both hidden layers. Delay lines are represented as shaded boxes. [135].

hidden layers are activated and fill the buffers in the second hidden layer which will then activate the output neurons. In other words, the activation of a neuron in a layer is calculated after all the buffers in the previous layer are filled. Once the output has been obtained, similar to the feedforward multi-layer perceptron, gradient-based back propagation techniques are used to train the TDNN.

TDNN has the advantage of detecting temporally localised features. However it does not cope with huge temporal variance. Since TDNN employs a fixed number of buffers in layers upon the analysis on training data, it does not guarantee overcoming unseen temporal variances in test data. In addition to that, in the case of long sequences, TDNN needs a large amount of neurons and connections, which directly increases the training time.

The study [156] implements a classic TDNN for recognition of 40 dynamic and static American Sign Language gestures. Feature vector (x, y, ϑ (maginitude) and θ (angle of velocity)) are extracted from each frame of the image sequence (gesture). TDNN network has 50, 46, 37 and 18 neurons in input, first and second hidden and output layers respectively. TDNN has achieved 98.14 % and 93.42 % recognition rate on training and test data respectively. By employing a "voting" scheme, the recognition rates on training and test data are improved to 99.02 % and 96.21 % respectively.

In [114], TDNN is implemented to recognize a small set of static and dynamic American Sign Language gestures. Glove-based data acquisition is used to recognize 10 gestures. The dataset shows quite wide variance since each gesture is performed 10 times by 10 different users. The dimension of the feature vector is 22. Therefore, TDNN networks consists of $22 \times 2 \times q$ input nodes in order to hold 2 second-long average gestures with q sample rate and two hidden layers. In addition to TDNN, in this study,

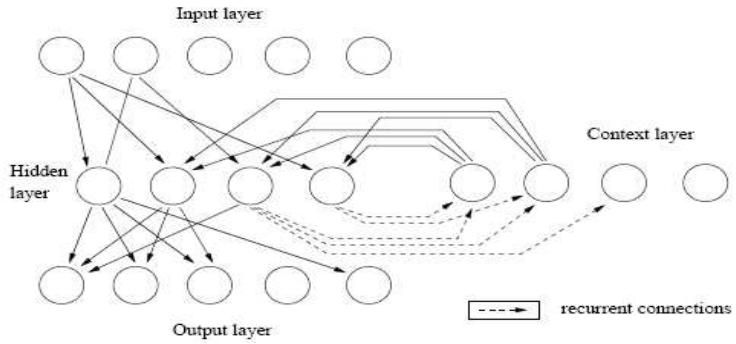


Figure 3.8: A Recurrent Elman neural network with a hidden and content layer [62].

a C.45 decision tree and Bayesian classification schemes are tested. TDNN, C.45 and Bayesian classification techniques obtain 20.18 ∓ 7.92, 22 ∓ 8 and 15.34 ∓ 2.94 mean and standard derivation recognition error rates respectively. Unfortunately, TDNN does not obtain better results than Bayesian and its output is not as meaningful as those of C.45 and Bayesian.

Recurrent Neural Networks

Bidirectional, (forward and backward), cyclical links between neurons are allowed in RNN unlike feedforward neural networks and TDNN. Therefore, RNN is more biologically plausible [42]. In theory, RNN can map any input sequence to an output sequence. Neurons in the context layers work as memory units to hold sequences.

Even in a small network, cyclical links and training by backpropagation create chaotic behaviour which makes it non-trivial to analyse RNN [111]. In order to tackle or restirict this shortcoming and other mentioned disadvantages of a neural network, in literature, four main type of recurrent neural networks have been described: Simple Recurrent Networks (Elman), Backpropagation Through Time (BPTT), Real Time Recurrent Learning (RTRL) and Long - Short Term Memories (LSTM).

The first and simplest RNN was proposed by Elman [30]. Figure 3.8 illustrates a simple recurrent network which is also referred to as the Elman network. In fact, SRN is a partial RNN, because, during the training process, cycles are eliminated and hidden layer activation is only copied into another hidden layer (context layer) to use later. Another commonly used SRN is the Jordan network [55, 56]. Unlike the Elman network, context neurons are the input layer and there is a connection from output neurons to get feedback. The weights of recurrent links between the context and hidden layers are always 1, namely the training process does not update the recurrent links. Eliminating the cycles uses a powerful conventional backpropagation algorithm

for training. Since Elman stores the latest results of the hidden layer, it fails to find the structure in long sequential data. Furthermore, to obtain a target output to train RNN is a non-trivial task[70]. For each input sequence, a meaningful target output has to be defined. In addition, representation of time is task dependent. In spite of those limitations, Elman networks obtain good results on short sequential patterns. The main advantage of Elman networks over other RNN architecture is that, in terms of space and time, the cost of training an Elman network is lower compared to other RNNs.

Backpropagation Through Time (BPTT) networks are proposed by Rumelhart et. al. [106] and perform better than SRN. Each discrete time step is represented by a hidden layer in BPTT. In other words, the recurrent network is unfolded to a fixed number (length of sequence) of feedforwarded layers. Unlike MLP, the neurons in layer t are connected to the neurons in layer $t-1$, input and output. A standard backpropagation algorithm is used to train the network with the following difference: For each layer at each discrete time step, local gradients, activation and output results are calculated and stored. These locally computed results are then utilised to calculate the true error of weights, once all the sequences are fed to the network. Since each time step is represented by a layer in BPTT, it may fail to extract earlier temporal sub-events in long sequences. In addition, space complexity of BTPP can be an issue for longer sequential data.

Like BPTT, Real Time Recurrent Learning (RTRL) computes the true error derivative of each weight. During the forward phase, by utilising the previous time step's weight, the current error derivatives of each weights are calculated. Therefore unlike BPTT, it is not required to process all the sequence and store the intermediate local activation, output and gradient results at each time step. One clear advantage of this technique over BPTT is that it has more efficient space and time complexity.

Generally SRN, BPTT and RTRL obtain good results on short sequential patterns. In practice, these networks do not work as well on real world datasets. In the last decade, a more robust network, Long - Short Term Memories (LSTM) was developed by Hochreiter and Schmidhuber [44]. It was observed by Schmidhuber in 1990, that an error path integral decays exponentially in BPTT and RTRL. For example, while a RNN can recall 10 steps back of a problem, with LSTM, as many as 1000 steps back of information can be recalled. A simple LSTM has a single input,a single output and a single memory block which corresponds to the hidden unit in a typical RNN. A memory block can contain one or more memory cells, each of which has a unit weighted self-recurrent link to keep the content for a long period of time. The content of memory cells are controlled by gates which allow or prevent error flowing. For

training, a combination of RTRL and BPTT is used. For further discussion see [44].

In [82], Murakami and Taguchi implemented an Elman based RNN to recognize 10 dynamic gestures of Japanese sign language. Thirty one dimensional feature vector (10 for bending, 9 for relative and absolute coordinate and 3 for angular orientation) extracted from a glove input device were fed into the Elman network with the previous two frame data. The network has 93 (which consisted of data at the time of $t, t-1, t-2$), 150, 150 and 10 neurons in the input, hidden, context and output layers respectively. The traditional backpropagation techniques were used to train the network. It took 4 days to train the network with the existing technology of the time and it achieved an impressive 96 % recognition rate.

The study [19] combined the TDNN and Jordan networks to recognize pointing gestures in the context of a drawing application. A magnetic tracker-based scheme was used to capture four pointing and non-pointing gestures. The feature vector is three dimensional and consisted of sign of gradient $(-1, 0, 1)$ of positions (x, y, z) between consecutive frames. The window length accommodating the previous frames in the input layer as part of the TDNN implementation were 5. Weights between context and output neurons was unit weights and were not updated. The rest of the weights were updated by backpropagation with the least mean square error function. The sigmoid activation function was used in the hidden and the output layers. On pointing and non-pointing test data, the system achieved 89 % and 76 % recognition rated, respectively. In a previous study on a different dataset with limited vocabulary size, the authour had obtained better results by employing the same RNN architecture as an emission probability estimator for states as a part of a hybrid HMM/RNN system [20]. Further details of this study are presented in Hybrid Classification Techniques section 3.2.4.5.

Vamplew and Adams implemented a RNN architecture similar to Elman [137] as part of the SLARTI sign language system[136]. The network is designed to recognize 16 isolated dynamic gestures polled from a glove-based input device. The training and test dataset consist of 560 and 320 samples from three different people. The network has 3, 14 and 16 neurons in the input, context and output layers respectively. The difference of the network from a typical Elman system is that, it does not have a hidden layer, and recurrent links from the context layer are connected bidirectionally to the output layer. A backpropagation through time (BPTT) algorithm is used to train the network. It learnt to recognize test and training data with 95.9 % and 98.9 % success rates. In addition, with a contribution of pre-segmentation, the network is also able to recognize gestures 40 % earlier. This also depends on the correlation in the dataset.

3.2.4.2 Hidden Markov Model

Hidden Markov Models (HMM) are widely used in temporal classification tasks such as speech, handwriting and gesture recognition, because of its stochastic and statistical framework. The stochastic framework is based on two assumptions- first order Markov chain over a finite number of states, and a finite set of observations each of which is associated with a state with an emission distribution density.

Although, in literature HMMs are mostly used for classification tasks, they can be used for generative purposes such as speech synthesis [129]. Because of their potential, the variants and extensions of HMMs have been investigated in several disciplines for many years. An introductory tutorial and text on HMMs and their use for recognition tasks can be found in [96], [97], [91] and [9].

HMMs are a stochastic finite state automata (SFSA), in which the emission of observations and transitions between states are expressed in a non-deterministic (probabilistic) manner rather than deterministic [12]. HMMs can also be represented as a dynamic Bayesian Network (DBN) [83, 107].

As its name suggests, HMMs inherit several properties from Markov Chains (property). In spite of that, there are two major differences. First of those is that in Markov chains, the probability of being at state (S_i) and time t, is dependent on the previous j states. But in HMM, the current state is only dependent on the predecessor state (j=1). The equation 3.8 describes how an jth order Markov chain is reduced to first order Markov chain in HMM. The second important difference is that in Markov chains, states correspond to observable (physical) events. Whereas in HMM, states correspond to a probabilistic density process which generates an observation with a probability ($b(O)$).

$$P(q_t = S_m | q_{t-1} = S_n, q_{t-2} = S_p, \cdots, q_{t-j} = S_r) = P(q_t = S_m | q_{t-1} = S_n) \tag{3.8}$$

In fact, the word "hidden" in HMMs means the approximation of real states by stochastic means. There are several real life examples in which the structure of states and its processes are indirectly and not directly observable, and probabilistically observable through another set of processes. Since the real state of the source is not known directly, these real states are approximated by "hidden" states. For example, in a speech recognition system, hidden states accumulate to approximate the process of real states (throat, vocal chords, tongue and other organs).

The basic idea behind HMMs is to build a model, called λ, underlying the stochastic

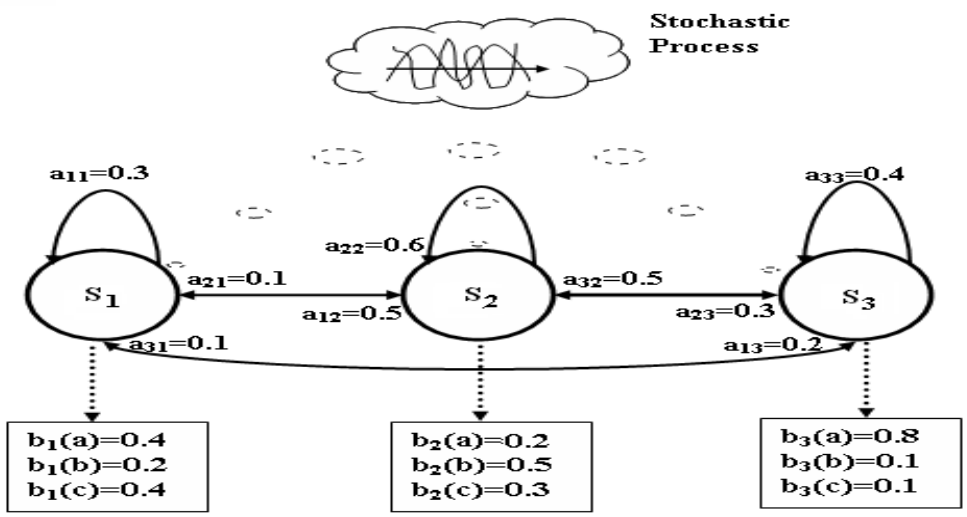


Figure 3.9: Hidden Markov Model with three states and three observation symbols

frameworks to characterize and generalize the sequences of observations [96]. The model consists of states ($S = s$), transition probabilities ($A = a_{ij}$) between states and emission probabilities ($B = b_j(O_k)$) of the observations at each state. Subsequent to optimizing the model to maximize the probability of producing training sequences, the probability of a given sequence of observation (O) produced by the model (λ) can be calculated ($P(O|\lambda)$). Figure 3.9 illustrates an HMM which has three states, transition probabilities and emission probabilities of three observations at each state. HMM, in a compact notation λ=(A, B, Π), can be described formally as follows [97]:

1. N: Number of the states in the model. The set of state is $S = \{S_1, S_2, S_3 \cdots S_N\}$. The state at time t is q_t.

2. M: Number of the distinct observations or output symbol per state. $V = \{v_1, v_2, v_3 \cdots v_M\}$

3. $A = \{a_{ij}\}$: The state transition probability distribution where
   $a_{ij} = P[q_{t+1} = S_j | q_t = S_i], \quad 1 \leq i, j \ \leq N \ and \ a_{ij} \geq 0.$

4. $B = \{b_j(k)\}$: The observation symbol (emission) probability distribution in state j where
   $b_j = P[v_k \ at \ t | q_t = S_j], \quad 1 \leq j \ \leq N \ and \ 1 \leq k \ \leq M.$

5. $\Pi = \pi_i$: Initial State distribution where
   $\pi_i = P[q_1 = S_i], \quad 1 \leq i \ \leq N.$

Note that stochastic constraints are preserved as follows:

$$\sum_{i=1}^{N} \pi_i = 1 \ ; \quad \sum_{j=1}^{N} a_{ij} = 1 \ ; \quad \sum_{k=1}^{M} b_j(O_k) = 1 \ ; \ 1 \leq \ i \leq N \ ; \ 1 \leq \ k \leq M$$

Prior to the use of HMM as a recognizer or generator of an observation sequence ($O = O_1 O_2 O_3 \cdots O_T$), some key problems have to be addressed. These problems are as follows:

1. The Evaluation Problem: Determining the probability with which a given sequence of observable symbols would be generated by an HMM ($P(O|\lambda)$). For example, what is the probability of producing the sequence of *acbbca* by the HMM in figure 3.9?

2. The Decoding Problem: Determining the most likely sequence of internal states in a given HMM which could give rise to a given sequence of observable symbols

3. The Training Problem: Generating an HMM, in other words, optimizing the λ parameter, that best explains a sequence or set of sequence of observables.

In order to comprehend the HMM fully, the solution to these three problems have to be investigated in detail. This investigation also leads us to make an analogy between HMM and the proposed algorithm in the following chapters.

The Evaluation Problem

The problem of interest is how to calculate $P(O|\lambda)$ as part of a pattern recognition task. $P(O|\lambda)$ are the essential measurements which indicate similarities between a sequence of observations O and HMMs. As a matter of pattern recognition, in a nutshell, the HMM among other HMMs, which maximizes $P(O|\lambda)$, is chosen as the recognized class or model.

Since HMMs are stochastic finite state automata, the calculation of $P(O|\lambda)$ is time consuming. Let us look at the calculation with an example: the HMM illustrated in figure 3.9 and the sequence of $O_e = acbbca$. The probability of O_e produced by λ, $P(O_e|\lambda)$ is the sum of all joint probabilities which emerge through all the state sequences of length six (the length of O_e). It is obvious that this calculation involves an exhaustive search. For simplicity, assume that the state sequence is known as follows:

$$Q = (S_1, S_2, S_2, S_1, S_3, S_3)$$

In that case, $P(O_e|Q, \lambda)$ would be :

$$P(O_e|Q, \lambda) = \pi_1 * 0.4 * 0.5 * 0.3 * 0.6 * 0.5 * 0.1 * 02 * 02 * 0.1 * 0.4 * 0.8$$

where π_1 is the initial state distribution of S_1. Briefly, the probability of $P(O|Q, \lambda)$ is the joint probabilities of transition probabilities and emission probabilities through the state sequence. It should be noted that two assumptions have been made here:

1. First order Markov property: The probability of being at a certain state is only dependent on the predecessor state. In addition, state transition probabilities are time invariant.

2. Output Independent: The observations are independent of their neighbour observations but are dependent on the states generating them.

In general, the probability that a sequence of observations O, is produced by a HMM (P($O|\lambda$)) is the sum of all the possible paths of observation of length T:

$$P(O|\lambda) = \sum_{all\ \ S} \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_t}(O_t)$$

This straightforward scheme, exhaustive search, has a polynomial time complexity of $\approx 2T * N^T$, where N is the number of states. Therefore, a more efficient *forward-backward* algorithm is proposed in [1], [2]. In fact, just using the forward procedure is enough to calculate the probability. Hence, for the sake of clarity, backward procedure will be omitted.

The forward algorithm defines a forward variable $\alpha_t(i)$ which represents the probability of getting the state S_i at time t. The forward algorithm is as follows:

1. Initialization: Forward variables are initialized as the joint probabilities of initial state distributions and emission probabilities of first observation.

$$\alpha_1(i) = \pi_i b_i(O_1) \ ; \quad 1 \leq i \leq N$$

2. Induction: In a recursive way, until the end of the sequence $(T-1)$, the forward variable of each state at time t is calculated as follows:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(O_{t+1}) \ ; \quad 1 \leq j \leq N \ ; \ \ 1 \leq t < T-1$$

The gist of the induction step is that the probability of being in a certain state S_j at time $t+1$ is the joint probability of emission probability $b_j(O_{t+1})$ and previous partial probability, $\mathrm{P}(O_1, O_2, \cdots, O_t|\lambda)$ weighted by the state transitions to the state S_j (a_{ij}). The partial probability is the sum of all the recent forward variables, $\alpha_t(i)$, which encapsulate the probabilities of observations $(O_1, O_2, \cdots, O_t)$ along all possible paths terminating at the state i (S_i).

3. Termination: Finally, the sum of all the forward variables at time T, $\alpha_T(i)$, gives the desired probability.

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Since the forward algorithm implements a recursion scheme to avoid exhaustive search, the time complexity of the solution of this problem is reduced to N^2T from $2T * N^T$.

The Decoding Problem

This problem is to uncover the hidden part of HMM, in other words, to find the optimal state sequence in the light of a given observation sequence. Optimality criteria are dependent on the domains. Therefore, there can be several optimal state sequences. One of the most practical optimality criteria is the one in which $P(O|\lambda)$ is maximized. For finding this state sequence, a dynamic programming based algorithm, *Viterbi*, is used[139],[34]. Informally this means, the algorithm iteratively selects the path on which the joint probability of the state transition and emission is maximum,δ, and stores the path in a variable, ψ, in order to restore the path. Formally the algorithm is stated as follows [97]:

1. Initialization:

$$\begin{aligned} \delta_1(i) &= \pi_1 b_1(O_1) \ ; \quad 1 \leq i \leq N \\ \psi_1(i) &= 0; \end{aligned}$$

2. Recursion:

$$\begin{aligned} \delta_t(j) &= \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(O_t) \ ; \ 2 \leq t \leq T \ ; \ 1 \leq j \leq N \\ \psi_t(j) &= \operatorname*{argmax}_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] \ ; \qquad 2 \leq t \leq T \ ; \ 1 \leq j \leq N \end{aligned}$$

3. Termination:

$$\begin{aligned} P^* &= \max_{1\leq i\leq N}[\delta_T(i)] \\ q_t^* &= \operatorname*{argmax}_{1\leq i\leq N}[\delta_T(i)] \end{aligned}$$

4. Backtracking:

$$q_t^* = \psi_{t+1}(q_{t+1}^*)\ ;\ t = T-1, T-2, \cdots, 1.$$

It is obvious that the Viterbi algorithm is a restricted version of the forward algorithm where instead of taking all paths, only the maximum path at each time step is taken. In practice, the Viterbi algorithm is also used for evaluation $P(O|\lambda)$ at the recognition phase. Although, the Viterbi algorithm reduces the time complexity of the evaluation phase, the accuracy is not as robust as the forward algorithm, since $P(O|\lambda)$ is approximated by only considering the maximum path. But the study [96] shows that the accuracy provided by the Viterbi algorithm is adequate in practice.

The Training Problem

This problem is the most difficult one of all the steps. It attempts to find optimum HMM parameters, (A, B, π), to maximize the training observation sequences, $P(O|\lambda)$. Apart from these parameters the number of states, structure of state (intra state connections), observation symbols (continuous, discrete, single or multivariate) play vital roles. Unfortunately, there is no analytic solution for the latter factors. These are domain dependent and have to be set by trial and error.

On the other hand, the model parameters, (A, B, π), can be optimized by using the *Baum-Welch* algorithm in advance. In fact, the Baum-Welch algorithm implements the EM (Expectation, maximization) optimization algorithm. The Baum-Welch algorithm recursively adjusts the parameters to maximize $P(O|\lambda)$ locally until some limitation point is reached [6], [97]. The main idea behind this iteration is as follows: Maximize the state transitions and the emission probabilities of the most frequent (expected or likely) paths or states given the observation sequences. In other words, for example, in the case of a state transition, an a_{ij} is modified proportionally to its frequency or expectation (the number of transition from states S_i to the state S_j out of total state transitions) during the production of the observation sequence. In a similar way, emission (B) and initial distribution (Π) probabilities are also optimized.

It should be noted that if these frequencies or expectations of parameters are known

in advance, the parameters can be initialized without using the Baum-Welch algorithm [53]. In addition, the correlation between parameters can be available in advance. In this case, a technique called parameter tying can be used. In parameter tying, parameters are set according to these intra linear or non-linear relationships.

Types of HMMs

It is possible to classify HMMs according to their structure of state transitions. As mentioned earlier, the structure of the state transitions is one of the important factors among others affecting the recognition phase. Therefore, structures of state transitions are dependent on the application domain, as there is no analytical solution for deciding the structure of a state transition.

Hitherto, fully-connected, ergodic HMMs have been considered. An example is shown in figure 3.9. In fully-connected HMMs, at any discrete instance of time, any state can be accessed from others according to state transition probabilities. By imposing some constraints over the state transitions, several variants of HMMs can be built. For example in the left-to-right (lr) model, the transitions are allowed only from left to right states. The other transitions are set to zero ($\pi_i = 0$). Furthermore, in the case of left-to-right HMMs, the transitions between states can be organized so that some states are skipped. For example, in skip 2 lr HMMs, only the transitions between, S_i and S_{i+2} are valid.

Because of the wide usage of HMMs in various areas, several variants and extensions of HMM have emerged such as the Hybrid HMM with artificial intelligence, input-Output HMMs, weighted transducers, variable-length Markov models, Markov Switching Models, switching state-space models, coupled HMMs, Factorial HMMs and Hierarchical HMMs. The structure and connection of states, probability distributions, markovian assumptions are the main factors distinguishing these classical HMMs models. For a further discussion, the reader is referred to [83] and [5].

In the light of application domains, especially in cases where the state transitions are predicted from observations, the structure of HMMs can be determined.

Discrete and Continuous Observation

So far, observations are considered as discrete symbols chosen from a finite alphabet which directly implies using a discrete probability density in each state for calculating the observation probabilities. In HMM data are generally discretised by vector quantization algorithms.

Vector quantization is a data compression method in which continuous or discrete

data are mapped into predefined clusters, or codes [53], [39]. Vector quantization is similar to the K-means clustering algorithm in a way that data are mapped into the nearest cluster by using some distance metrics such as Euclidean distance. In the domain of HMMs, the set of clusters, codebook, can be thought of as a finite observation alphabet. The vector quantization algorithm maps the input data (acoustic speech, gesture data) into symbols from the finite alphabet (namely a code from the codebook).

Unfortunately, information lost in vector quantization is enough to affect the performance and robustness of the systems. Hence, in some applications, due to the nature of the signal and discretisation method, using discrete probability densities for observations (discrete symbols), leads to an incomplete representation of the data, and degrades performance. Therefore, continuous observation density is generally preferred for the observation probabilities.

In the case of continuous observation density, the probability that the symbol produced in a state is based on a continuous probability density function (pdf) with some underlying parameters. For example, in the case of Gaussian distribution, which is the most used state distribution in HMMs, the parameters are mean and variance for each state. In the case of multivariate data or vectors, Gaussian mixtures are used. In the case of Gaussian mixtures, besides the mean and covariance matrix, the mixture coefficient plays an important role in calculating observation probabilities. In a nutshell, the mixture coefficients indicate the degree of participation of channels.

Advantages & Disadvantages of HMM

Advantages:

- Statistical Data Representation: Data are represented with well established statistical techniques and distributions. On the other hand, assuming an underlying distribution can be a disadvantage, because in some real world problems, the data does not behave according to any well established statistical distribution. Actually, this property of HMM is its biggest weakness. Most of the hybrid system, address this weakness of HMM.

- On-line Recognition: HMM can be modelled and optimized (embedded re-estimation) to achieve on-line recognition by using grammar easily.

- Transparency: HMM is not a black box like other machine learning techniques such as Neural Networks. The models can be analysed, developed and maintained rigorously.

- Modularity: Once a HMM is constructed, it can be used as an atom or token of any other system.
- Incorporation of Prior Knowledge: Prior knowledge can be readily imported to optimize the parameters, structure.
- Synthesis: The model built for recognition purpose can be easily adapted for sequence synthesis [142]. For example, a speech synthesis system (HTS) is implemented using a hidden Markov toolkit (HTK)[129].

Disadvantages:

- First order Markovian: Only the last visited states affect the current transition, which is not realistic. However this assumption reduces computation cost and complexity.
- Independent Output Assumption: As first order markovian, outputs are assumed independent from each other, which is not realistic in many domains.
- Over or Under Fitting: A huge number of training sets are needed for a reliable statistical representation, especially in the case of large parameters.
- Huge number of parameters to be optimized: Initial transition, transition and observation parameters λ=(A, B, Π) have to be optimized before using.
- Domain dependent models: HMMs structures, observation symbols, and the number of states are all domain dependent and there is no easy analytic way to estimate these parameters. These parameters are set by trial and error.
- Using only positive data: A good classifier has to maximize the probabilities of observations of a certain class while it minimizes the probabilities of the observations belonging to other classes. HMMs uses only positive data to maximize the probabilities of observations of its classes. It does not minimize the probability of being a member of other classes. It suffers from the disadvantages of all generative classifiers [10].
- Weak criteria to announce classification: The probability of models $P(O|\lambda)$ is the only criteria (maximum likelihood $P(O|\lambda)$) to announce a recognition. This criterion is weak in the case of on-line recognition in which the start and end points of a pattern is not known in advance. HMM needs additional heuristics to overcome this shortcoming.

The technical report [7], represents a detailed discussion of the capacity of HMMs.

Gesture Recognition Applications by HMM

In the last few decades, the Hidden Markov Models, its variations and extensions thereof, have attracted a considerable attention for gesture and other temporal sequence recognition. Many studies have been conducted. Especially, with the availability of of HMM toolkits such as HTK [157] several applications have been developed.

Jie Yang and et al have developed an interaction prototype system consisting of nine gestures, each of which corresponds to a digit, in the context of telerobotics and human computer interaction [154]. The mouse is the used as input device. The authors applied discrete Hidden Markov Models for isolated and continuous recognition tasks. Sixteen dimensional Fourier Transformation (FFT) amplitudes are used as feature sets. Since the discrete HMM is used, feature vectors are discretisized by the vector quantization technique, prior to training and recognition phase. The prototype system achieves an impressive result (99.78 %) in the case of the isolated recognition task. This recognition achievement can be accommodated to data correlations in the dataset. Nine gestures, each of which corresponds to a digit, in 2D dimension have quite a high disparity degree. It is also worth noting that the Fourier Transformation (FFT) based feature set does not easly allow itself for on-line recognition.

Lee and Hu developed a dynamic system using HMM which is able to learn on-line and interactively with a small training set[68]. In other words, it updates its parameter iteratively during the recognition phase. The system is designed to recognize hand gestures acquired by a glove device which provides 20 merits from hand configuration. In the recognition phase, if the system cannot reach a reliable recognition about the test gesture, with a supervised approach, it seeks help from the user to recognize the gesture. Based on this feedback from the user, it updates its model parameters which are the core of on-line, interactive learning. Discrete 5 state left to right, 1 or 2 skip HMMs are used for gesture modelling and the iterative Baum-Welch algorithm is applied for training. During the experimental phase, 14 isolated gestures, which are acquired by a glove-based input device are used. The system obtains accurate results even after two - four training samples for each gesture. Limitations of this system are as follows: First, it is designed for isolated gesture recognition. Between each gesture the user has to stay still in order to indicate separation between gestures. This prior knowledge is utilised for segmentation. In the case of continuous gesture recognition, performance will be affected dramatically. In addition, the recognition operation is conducted after all the data is acquired. In another words, it is off-line based. Another point worth noting is that hand gestures data are unambiguous in respect of hand orientation. Gestures, 14 letters from a sign language, do not accommodate high inter

similarity.

Rigoll et al designed a real-time image-based dynamic isolated gesture recognition system by using a continuous HMM [101]. 24 isolated gesture classes from 14 people comprising a 336 sample-sized dataset. In the pre-process phase, the difference between consecutive frames is used to extract background from images and to construct seven dimensional feature vectors. Various HMM topologies have been used for gesture models. The study achieves 92.9 % recognition rates. The authors, in a subsequent paper [29], extended the work to address non-defined gestures, position independent and continuous recognition with some restriction and assumptions. For example, in order to eliminate position dependency, this time, background information is obtained in advance and the subject is directly extracted from the image by utilising this background knowledge. In case of continuous recognition, it is assumed that decreasing probabilities of all HMM models for gesture sequence at a point indicate a separation between gestures. This assumption is true in large datasets and noisy long sequences. The paper does not report a certain recognition rate in case of continuous gesture recognition.

In [84], Yam and Wohn developed a HMM-based recognition system for continuous dynamic and static 3D gestures in an off-line manner. A Polhemus tracker and a glove input device are used to acquire angle of fingers and orientation/position of a hand. 3D data is mapped to a 2D plane before normalization to a common size. In the experiment part, 10 gestures are used to validate the system. Left - right topology is used for modelling the gestures. HMMs are connected with juncture HMMs, which stand for non-defined gesture or, more precisely, connection motions between gestures in a continuous gesture stream. For each gesture, 300 samples, (while third of them are used for testing, the rest is used for training) are collected. The system obtains 99.01 % gesture recognition rates on segmented isolated gestures. But in the case of continuous gesture sequences, which consist of 3 gestures and a juncture between them, the system only achieve 80 % recognition rate, even though, context and context information is also utilised during continuous recognition.

Starner and Pentland developed a real-time system to recognize forty dynamic American Sign Language gestures [122, 123]. The system is aimed to recognize continuous gestures in a sentence. A single colour camera is used to track specially marked hand gestures (Yellow and orange gloves for right and left hand respectively). Having pre-processed single colour images, a threshold computer vision mechanism is used to segment the hand from the rest of the image. Then, eight dimensional feature vectors are extracted from the segmented hand in each frame. The HTK tool-kit [157] is used to train the model with non-segmented continuous gestures (sentence). Training

the model with continuous, non-segmented gestures incorporates to discover or accommodates context information (in this case ASL grammar) into modelling. Firstly, a uniformly divided sentence is used to initialize models by using the Viterbi algorithm. Then the Baum-Welch algorithm is deployed to re-estimate the model parameters. 494 sentences from a singer are used to test and train the system. The experiment on training data (all 494 sentences are used) attains 99.5 % and 92.0 % word recognition rate with grammar or without grammar, respectively. On the other hand, the experiment with a subset of dataset (99 for test,the rest (395) for training) obtains 99.2 % and 91.3 % word recognition rate with grammar or without grammar, respectively. Although, the system obtains reliable results with grammar, in case of no-grammar and multiple singers, it shows that the system suffers from high temporal and spatial variance.

In [69], the authors introduced an automatic gesture recognition system for human-robot interaction. They were whole body gestures (such as walking and running on a mobile robot) rather than just hand gestures. Two stereo cameras were used to capture 3D human body's joints and construct a 3D prototype of body motions in real-time. 13 angle features from various part of body were used as feature vectors. The angles were between an interest of the body part and the yz, xz, and xy planes. Trajectory features (feature sequence in a gesture) were reduced to a clusters sequence (with an EM based algorithm) in order to reduce dimensionality and obtain real-time recognition. A left-right HMM-based recognition algorithm was used as the recognition algorithm. An ergodic garbage HMM model was used to represent undefined gestures. If the likelihood of any defined gesture HMM was bigger than the likelihood of a garbage model for a given input gesture, it was assumed that a defined gesture was spotted. The system achieved 94.8 % spot and recognition rates over ten isolated gestures.

3.2.4.3 Dynamic Time Warping

Dynamic time warping (DTW) was among the first techniques used to solve temporal classification problems. Besides temporal classification, DTW is applied in many other areas such as data mining, manufacturing, medicine and robotics.

In fact, the idea behind DTW is inherited from dynamic programming which is a well-studied subject in operational research. Hence, a brief explanation on dynamic programming is appropriate here.

Dynamic Programming

Dynamic programming is a generic bottom-up problem solving technique which approaches problems as a sequential decision process. The essence of dynamic programming is based on the Principle of Optimality which is developed by Richard Bellman, the founder of dynamic programming, at RAND in the 1950's. The Principle of Optimality is as follows [3]:

> "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."

The Principle of Optimality implies that the optimality of a problem depends on the optimal solutions of its sub problems. In dynamic programming, problems are decomposed into sequential sub-problems or stages, of which solutions, once again, are based on sequential decision models and intra relationships. Basically, dynamic programming solves a multi-variable problem by solving a series of relatively simple, (single variable) problems. Dynamic programming consists of the followings steps:

- Decomposition: The problem is divided into small sequential stages or sub problems. It is assumed that the underlying process of the problem is *Markovian* process. It implies that a decision at a particular stage only depends on the recent preceding stage.

- Recursive: A recursive function is constructed to find the optimal solution of each sub problem and in the end it finds the ultimate optimal solution.

The decisions taken at these stages can be deterministic or stochastic. In the deterministic case, given the initial stage and decisions, the final stage is calculated precisely. But in the case of the stochastic, the final result is not calculated precisely but with some probability. The Viterbi algorithm, which is elaborated in detail in the HMM section, is an example of stochastic dynamic programming [147],[116].

A dynamic programming scheme approaches problems in the same way a human being does. In daily life, lots of problems are solved subconsciously via dynamic programming.

A more comprehensive discussion on dynamic programming can be found in [116]. Furthermore, a non-technical text about the birth of dynamic programming and its founders can be found here [26].

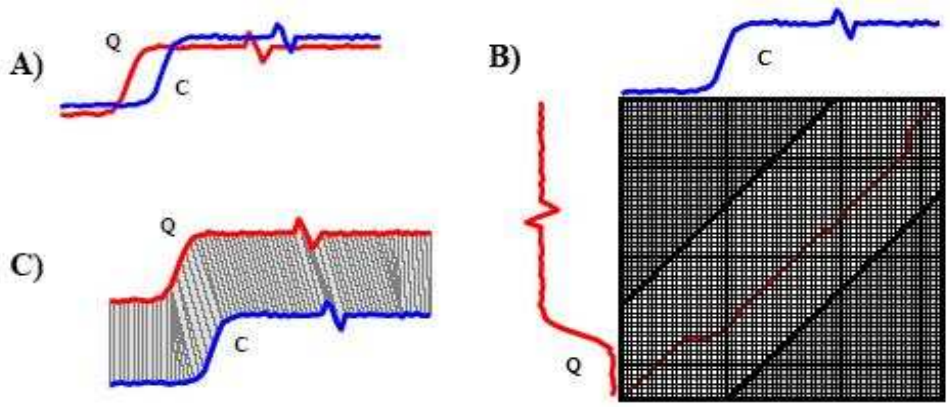


Figure 3.10: Dynamic Time Warping, A) Two similar time signals, reference (Q) and input (C). B) A warping matrix is constructed out of Q and C. The optimal warping path is searched by using Sakoe-Chiba Band windowing scheme and other constraints. The dark gray corners (top-left and bottom-right) are excluded from the search space. C) Aligned indexes of the input signal C. [99]

Dynamic Time Warping

Dynamic programming is a general and versatile technique which is used in many areas. Its specific applications on time series is called dynamic time warping (DTW) in computer science literature.

The main idea behind DTW is to warp the time indexes of two given signals to minimize the intra distance. The first early, temporal classification (in speech recognition) techniques used Euclidean distance with simple linear time alignment schemes by stretching and compressing the input signal prior to comparison with templates [147]. But, dynamic time warping uses a non-linear time alignment for the input signals.

In order to elaborate the classic dynamic time warping algorithm, let us assume we have a reference (Q) and input (test) signal (C) (Figure 3.10-A) [61]:

$$Q = q_1, q_2, q_3, ..., q_n$$
$$C = c_1, c_2, c_3, ..., c_m$$

For aligning these two signals, an $S_{n \times m}$ matrix is constructed (Figure 3.10-B). The matrix constitutes the search space, where each elements of the matrix $S_{i,j}$, corresponds to the distance between two points, d $(q_i,\ c_j)$. As a distance function, Euclidean distance is generally used.

$$d(q_i,\ c_j) = (q_i - c_j)^2$$

Matrix elements, in other words the distances, reside as an alignment or warping measurement between the signals. Upon that distance matrix, a warping function is defined to find the path W from $S_{1,1}$ to $S_{n,m}$ minimizing the total cumulative distance $\gamma(i,j)$ or warping cost. In other words, W corresponds to the total distance between the warped signals Q and C.

$$\gamma(i,j) = d(q_i, c_j) + min\{\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)\} \tag{3.9}$$

Each point in the warped path, W, consists of time indexes i and j, $w_n = \{i, j\}$, and W is defined as follows:

$$W = w_1, w_2, w_3, ..., w_k \quad max(m,n) \leq K < m + n - 1$$

The DTW algorithm seeks an optimum path which minimizes the total cumulative distance or in other words the total warping cost.

$$DTW(Q,C) = min\left\{\sqrt{\sum_{k=1}^{K} w_k}\ /K\right. \tag{3.10}$$

Since there are many possible paths, and in order to reduce the complexity of the searching, the following constraints are applied to the path [109].

- Boundary constraint: As was mentioned earlier, the warping path starts from $S_{1,1}$ and ends at $S_{n,m}$.

  $$w_1 = S_{1,1}\ ,\ w_k = S_{n,m}$$

  Although, certain boundary constraints speed up DTW, it degrades accuracy in speech recognition [98].

- Continuity: Each element of the warping path is adjacent to its succeeding and preceding elements.

  $$w_t = \{g, h\} \text{ and } w_{t+1} = \{g', h'\} \Rightarrow g' - g \leq 1 \text{ and } h' - h \leq 1$$

- Monotonicity: Apart from continuity, each step has to be non-decreasing in time indexes.

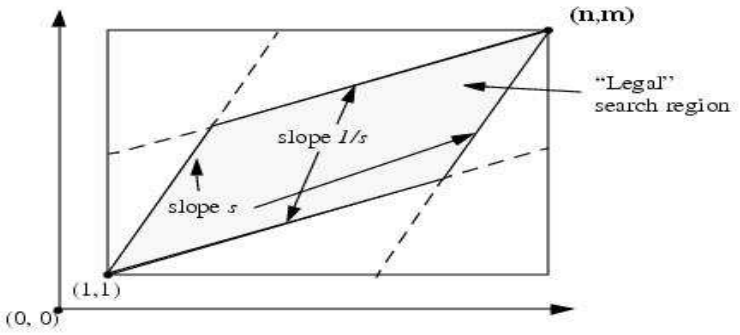


Figure 3.11: DTW Adjustment Window:Itakura Parallelogram [159]

$$w_t = \{g, h\} \text{ and } w_{t+1} = \{g', h'\} \Rightarrow g' - g \geq 0 \text{ and } h' - h \geq 0$$

- Adjustment Window: Since the possible optimum path is less likely to be at the corner of the matrix, these parts are pruned by using windows around the diagonal. In the literature, the most widely used windowing schemes are: Sakoe-Chiba Band and Itakura Parallelogram.
  The Sakoe-Chiba Band satisfies that $w_t = \{g, h\}$ and $|g - h| \leq r$ where r is a positive integer number called window size [109]. Figure 3.10-b illustrates a Sakoe-Chiba Band. The corners, illustrated as dark gray, are excluded from the search space.
  In the case of the Itakura Parallelogram, the valid region is in the shape of parallelogram around the diagonal [51], [159]. While the left bottom corner of the parallelogram passes through $S_{1,1,}$, the top right one passes through $S_{n,m}$. The slopes of the parallelogram's lines are s and $1/s$. Figure 3.11 illustrates the Itakura Parallelogram adjustment window scheme.
  It seems that there is a trade-off between accuracy and speed of DTW in terms of window size. In literature, there is no concurrence on the affects of adjustment windows to accuracy. For example, according to a recent empirical study [99], using a narrow window size provides better accuracy in data mining applications. On the other hand, according to a study in isolated speech recognition, a large window size provides more accuracy [98].

- Slope Weighting: The path should be neither too steep nor too shallow. Therefore, the total cumulative distance equation, 3.9, can be updated as follows :

$$\gamma(i, j) = d(q_i, c_j) + min\{\gamma(i-1, j-1), X\gamma(i-1, j), X\gamma(i, j-1)\}$$

Figure 3.12: Step Patterns: Classic (A) and Alternative Step Pattern (B)

where X is a positive real number for biasing a diagonal warped path.

- Step Pattern: It constrains the pattern (size and direction) of steps globally. In other words, the calculation of the total cumulative distance function is modified to take into account other cells rather than succeeding and preceding adjacent cells. Figure 3.12 denotes a new pattern step (B) with a classic one (A). While the classic step pattern, 3.12-A, denotes the equation 3.9, the new step pattern 3.12-B denotes the following equation :

$$\gamma(i,j) = d(q_i, c_j) + min\{\gamma(i-1, j-1), \gamma(i-1, j-2), \gamma(i-2, j-1)\}$$

The text [109] presents a detailed discussion about step pattern.

Figure 3.10 illustrates the DTW algorithm over a reference and input signal using the Sakoe-Chiba Band adjustment window and other constraints. A further discussion on classical DTW can be found in [65]

The great advantage of DTW is being able to deal with different lengths of signals by taking care of distortions along the time axis. In contrast, DTW has the following disadvantages:

- Slowness: Since time complexity of DTW is $O(N^2)$, where N is the length of the signals, it is not suitable for real-time applications especially in the case of large datasets.

- Template construction: Creating a template, which is generic,(as much as specific) is a non-trivial task in many domains.

- Distance Unit: In the case of a multi-variable template, it is non-trivial to find a common distance function or to aggregate different distance functions or units.

- Off-line Vocabulary Recognition: It is generally used for off-line vocabulary recognition because the start and end points of an input signal has to be known in advance to calculate the distance matrix. For continuous on-line recognition, the algorithm needs some modifications to predict automatically the start and end points of the vocabularies in an input signal.

For tackling these disadvantages, several variants of DTW are developed. For example, Piecewise Dynamic Time Warping (PDTW) is used to speed up DTW [61]. PDTW introduces a dimensionality reduction scheme, PAA (Piecewise Aggregate Approximations), in which signals are approximated by dividing them into segments. Then, the mean of the segments is used as new data for each data on that segment. PDTW speeds up DTW by a large constant, one to two orders of magnitude, $O(2N)$ (N is length of the signal), depending on the dataset. In addition, PDTW provides the same accuracy as DTW in the case of optimal the number of segments. The main problem associated with PDTW is that accuracy is very sensitive to the number of segments which are normally chosen manually.

In order to eliminate the lack of PDTW, Iterative Deepening DTW (IDDTW) is proposed [15]. IDDTW iteratively implements PDTW by using a predefined segment number. At first sight, it is expected that IDDTW would be slower than PDTW and DTW. But IDDTW dynamically applies an optimization scheme to eliminate signals whose cumulative distances are greater than the hitherto observed minimum distance. In the worst case, the time complexity of IDDTW is $O(4/3N)$. Furthermore, IDDTW preserves the same accuracy which the classic DTW and PDTW both have.

So far, the techniques used for tackling the slowness of DTW have been discussed. But in literature, there have been some other attempts to increase the accuracy of DTW. For example, Derivative Dynamic Time Warping (DDTW) is used for better alignment or warping [60]. In DDTW, the distance matrix $S_{n,m}$ corresponds to the distance between the first derivative of points rather than the raw data like in the classic DTW. In other words, DDTW considers the shape or gradient of signals. It has been shown that DDTW performs approximately 10 times better than classic DTW.

According the a latest comprehensive experimental study [99], some of these advantages and disadvantages, which have long been known, are not beyond some myths for data mining applications. The first of these myths is the one that DTW has a great ability to deal with different lengths of signals. It has been empirically shown that, when the DTW applied over different lengths of signal and interpolated to equal lengths of signals, in both cases, similar accuracy is obtained. The second myth is about the size of adjustment window. It has been believed that, a large window size

provides better accuracy. But, it has also been shown that, in many domains, the narrow window size gives better accuracy. The last myth is about the time complexity, namely the speed of DTW. It has been a common belief that DTW has to be speeded up for data mining applications. But it has also been shown that, for large datasets, DTW is already at the limit of its speed. Furthermore, regarding the speed, in the case of lower window size, the time complexity of DTW is closer to $O(N)$ than $O(N^2)$.

Gesture Recognition Applications using DTW

In literature, many experiments have been conducted on gesture recognition using DTW. Because of its nature, DTW is commonly used for isolated gesture recognition in an off-line manner. Generally these studies lend themselves to compare DTW with other techniques on an experimental basis. Moreover, these studies are commonly used to demonstrate the weakness of DTW on gesture recognition rather than its strength.

For example, in the study [17], even though vocabulary size is small (five gestures) and the start/end points of gestures are explicitly known, the results could not be any better than 92 % recognition. As was mentioned earlier in the Multi-Layer Perceptron (MLP) neural network section, as another example, the study [151] points out that, even DP achieves better results on a small set of static gestures of American Sign Language. Since the memory computation time and scalability disadvantages of DTW for large datasets is well known, DTW could not be considered for a comprehensive system. The study concludes in favour of MLP over DTW for large datasets.

In [71], Li and Greenspan utilised DTW for the recognition and segmentation of gestures in a continuous video stream. In order to tackle spatial and temporal inter/intra personal variance, a large number of reference templates (gesture models) were created synthetically with multiple scales from a main reference template. In addition, transitions between gestures were also represented with different reference templates (compound gesture model). That approach obviously increases the time and space complexity for large datasets. But in the study, a small dataset, eight gestures from five different subjects were used. While 90 gestures from three users were used for training, 60 gestures were used for testing. During data collection, temporal variance were kept as minimum as possible by letting users perform gestures at similar speeds. 88.1 % and 93.3 % recognition rates were obtained from multiple and single scales, respectively.

In a recent study [25], the dynamic programming technique (DTW) was used to track sign language. The study used DTW for tracking purposes rather a than recognition task, for which authors suggested to integrate DTW and the HMM scheme,

similar to the Recognition Machine (RM), as is presented in this thesis. The study deployed computer vision technique for data acquisition and it was assumed that the background was inhomogeneous, and that occlusions between hands and face occurred frequently. Motion, skin colour and eigenfaces [134] based features were used for tracking in an off-line manner. The system requires prior knowledge of the target, but if the target is not specified, it is noted that a higher computational resource is needed to track gestures.

3.2.4.4 Other Recognition Techniques

In addition to the above techniques, in literature, a variety of other techniques have been implemented for static and dynamic gesture recognition. The Bayesian classifier and classification trees are among these techniques. These two techniques are generally useful for recognition of static gesture in an off-line manner.

The latest theoretical and empirical studies have shown that with the simple, computationally trivial naive Bayesian classifier it is possible to achieve results as good as other state of art techniques [40, 103]. The Bayesian classifier is a generative model incorporating prior probability and observed data. A sample is assigned to the most likely class according to Bayes rule, which is based on joint probability of features and class prior probability. The Bayesian classifier makes two strong assumptions: first of which, there is no correlation between features, or in other words features are independent. Secondly, each feature is represented with a probability distribution function. One of the advantages of this technique is that training of the Bayesian classifier is trivial. Class prior or distribution parameters are estimated with a supervised learning approach (for example maximum likelihood).

Decision or classification trees are another commonly used static classification technique because of their trivial interpretability (white box) unlike, for example, in case of neural networks. A classification tree models the classification task as a one directional graph of a decision process given the attribute-value At_i, v_i pair, which represent samples. It maps each sample as a path from its root to a leaf node which correspond to a pre-defined and discrete class. Interior nodes and descending arches corresponds to attributes and attributes values respectively. A classification tree, in other words, corresponds to a rule-based induction or nested *if-else* conditions such as *if* $At_i = v_i$ *then if* $At_j = v_j$ *else if* $At_k == v_k \ldots$ *end*. Frequently, the information entropy-based C.45 algorithm is used for tree generation. For a further discussion on tree generation by C.45 readers are referred to [94, 95].

The study [14] implements the information entropy-based decision tree algorithm

to recognize isolated 65 hand gestures of Chinese Sing Language in an off-line manner. Gestures are collected by a glove and tracker-based input device (DataGlove and Polhemus). The C.45 algorithm is used to build the classification tree. On unseen raw test data, the system obtains 86.2 % recognition rates. As an improvement, when raw data is normalized, the system achieves 92.3 % recognition rates. Unfortunately, even the improved system cannot outperform neural networks which achieves 96.4 % over raw unseen test data. Similarly, as mentioned in the TDNN section, the study [114], C.45-based decision tree technique does not perform as well as NN or the Bayesian classifier.

3.2.4.5 Hybrid Classification Techniques

As has been mentioned in the previous sections, all recognition algorithms have their own specific advantages and disadvantages. In the literature, a variety of hybrid systems are utilise the best aspects of well-established techniques, while the disadvantages are suppressed. For example, in literature generally, neural networks have been used for their discriminative abilities and better generalization of unseen and noisy data. On the other hand, HMM is commonly used for time warping ability [100, 24, 145]. In most hybrid systems, NN is used to estimate Gaussian emission probability of HMM states. While HMM assumes the underlying statistical distribution is a Gaussian, NN try to approach the distribution without any assumptions. [21, 32, 18]. Another usage of NN in hybrid systems is to discover the optimal feature set or reduce input space by quantization [20, 135].

The study [100] combines the best aspects of Long-Short Term Memory RNN, (LSTM-RNN) and HMM for automatic segmentation and recognition of events in a meeting. While LSTM-RNN is employed for discriminative ability and long-term memory recalling, HMM is used for time warping and sequential analysis such as start/end point estimation. In other words, LSTM-RNN is used for optimal feature extraction from sequences. The dataset of the interest consist of six dynamic gestures. Besides LSTM-RNN/HMM hybrid system, the study also conducts experiments with pure LSTM-RNN and HMM. Stand-alone LSTM-RNN with manually predefined start/end boundaries achieves 96.33 % recognition rates. On the other hand, stand-alone HMM with automatic segmentation obtains 83.49 %. In the case of LSTM-RNN/HMM, by using raw data of LSTM-RNN, the hybrid system obtains 87.16 %. For a further improvement, when the output of LSTM-RNN is averaged with five bandwiths, the hybrid system obtains 92.66 % recognition rate. Without LSTM-RNN, the study also implements an MLP-based feature extraction technique which fails dramatically.

The study [20] uses four hybrid systems to recognize six dynamic and static gestures of customers interacting with shop assisting robots in a supermarket environment. While the first three hybrid system utilise HMM and different neural network architectures, the last one utilises a dynamic programming-based technique. These hybrid systems consist of two main phases: The pre-processing (feature extraction) and classification. Pre-processing phase implements a multi-cue approach to localise and extract the 14 dimensional feature vectors from images. The multi-cue approach is based on skin colour, facial structure and the structure of head-shoulder contours. For the classification task, the first hybrid system deploys a self-organizing neural network (SOM) and HMM. SOM clusters or maps continuous 14 dimensional feature vectors to two dimensional predefined discrete sub gesture regions which are then fed to the left to right HMMs to estimate the likelihood of the gesture. In other words, while SOM is used to reduce and emphasize the feature vector, HMM is used to compute the likelihood of the test gesture for each gesture of HMM. The second approach is based on HMM and the radial basis function (HMM/RBF). RBF is employed to estimate the state emission probabilities of HMMs. Therefore, for each state in HMM, an RBF is assigned. The training of RBF and parameter estimation of HMM are isolated from each other. The third approach uses HMM and simple recurrent neural networks (RNN) consisting of the Jordan network with additional time windows in the input layer as in TDNN. Similar to HMM/RBF, RNN is used to estimate the emission probabilities of each state in HMMs. In the last approach, dynamic time warping (DTW) is used independently. The length of the reference templates of a gesture is the average length of all samples of that gesture. Similarly, the reference template of a gesture is constructed by taking the mean vector of all the samples at each frame of that gesture. For testing, 225 gestures from five users are used. For the other three approaches, HMM/RBN outperforms SOM/HMM and DTW with 88.00 % recognition rate compared to 84.70 % and 75.86 % recognition rates respectively. The study has not published the HMM/RNN results yet. But in a later study, with a similar RNN configuration [19] the author obtains approximate results on a different and limited gesture set without employing HMM. Once again it is also observed that DTW is not as robust as other statistical and connectionist approaches for gesture recognition.

The study, [112], uses a hybrid system which employs a time delay radial basis neural network, and the Bayesian neural network classifier for a small size dynamic and static gesture set (14 gestures). Radial basis neural networks are deployed to extract the posture or features of the gestures. The centres of the RBF function is calculated by a vector quantization scheme in an unsupervised manner. In a further attempt, delay taps similar to time delay networks, are introduced in RBF to address

the temporal properties of the dynamic gestures. Finally, the discovered features from RBF are fed into the Bayesian neural network classifier to make a link between the features and the gestures. The study achieves remarkable results on static gestures, but after introducing delay taps, it suffers from recognizing dynamic gestures. It suggests including the context information to the input vector for a better recognition.

SLARTI, a hybrid sign language recognition system, deploys four different neural networks for each sub group of static and segmented AUSLAN gestures [135, 136]. In this system, NNs are used to extract optimal features which are, later used in different classification methods such as K nearest neighbour lookup and C 4.5 inductive learning algorithms. In all four neural networks, fully connected, feed-forward architecture with a single hidden layer is used. In this study, NN is used for reduction of feature space rather than classification.

In [22], a rule and MLP neural network-based gesture recognition framework (*GesRec*) are proposed to recognize gesture and hand-written character of speech and motor disabled people in an on-line manner. In the study, MLP is preferred due to its power of classification and relatively trivial training (compared to rule-based training). On the other hand, rule-based scheme is deployed to tackle the black box disadvantages of MLP for better interpretable output. The system deploys MLP and rule-based classification schemes at the same time independently from each other and then aggregates their scores in a weighted fuzzy scheme. Polhemus Fastrak and a graphic tablet are used to capture gestures and hand writing respectively. Both 2D and 3D data from the graphic tablet and tracker device are normalized and transformed to a 10×10 grid. To validate the system, 150 samples consisting of 26 the letters of English alphabet and four gestures (total 30 classes) are used. An MLP architecture 100-12-30 with sigmoid output and logistic function is used. For the rule-based system 15 binary attributes are used. The system achieves 92.5 % recognition rate while the self-deployed NN and rule-based systems obtain 89.6 % and 82.7 % rates respectively.

3.3 Previous Work on FDO Gestures

There have been several MSc projects since 1999 that have investigated various aspects of gestures arising in FDO training. These studies generally aimed to recognize isolated dynamic gestures in a off-line manner with simple static neural network architecture and they are described briefly in this section.

Trott's work is the first one that attempted to investigate gestures of FDO and develop a prototype system for FDO training within a virtual environment [132]. He

surveyed existing virtual environment software for prototyping. He developed a prototype by using Division virtual environment software. In his study, he prototyped six FDO gestures in various scenarios (environmental and weather conditions). Interaction between the user and the virtual environment is based on keyboard input.

Trott concluded his study with two main proposals, firstly is to use tracker-based Polhemus FASTRAK as the input device, and secondly is to conduct research into automatic gesture recognition by neural networks.

Following Trott's work, Suresh conducted research into recognition of six distinct and commonly used gestures of FDO by Hebbian Supervised Neural Network (HSNN) [126]. Asn tracker-based input device Polhemus FASTRAK was used to capture one hand position (x, y, z). The other hand's position is synthesized by taking a mirror copy.

The research focused on optimal input representation and HSNN classification technique. Two different approaches were used to represent gestures. In the first approach, all gestures were represented as a template, or more precisely, a binary, normalized 10×10 matrix or 2D grid. In the second approach, gestures were represented as 23 dimensional feature vectors consisting of coordinates, area, slope, min-max ratio and correlation between the arm parameters of both hands. The author concluded his study by pointing out that the feature-based approach was more robust, (in real-time) and more computationally efficient (time and space) compared to grid-based representation. He has also conducted more research into dynamic gesture recognition using other neural networks and algorithms.

In the light of previous work, Scarcia combined the recommended results of Trott's work on virtual environment prototyping and Suresh's work on gesture recognition [113]. The author developed a prototype virtual environment using the Maverik virtual environment toolkit (The MAnchester Virtual EnviRonment Interface Kernel) where six dynamic gesture of FDO, captured by Polhemus FASTRAK, are recognized automatically by the Hebbian Supervised Neural Network in an off-line manner. Scarcia represented gestures with a reduced feature vector (dimension 17).

Scarcia obtained about 67.6 % recognition rates over the six-gesture set and used 17 features. For improving the recognition rate he employed neural networks.

In addition to these MSc studies, a similar research has recently been conducted in the Department of Defence in USA [105]. This study has constructed a framework for recognizing FDO hand signals for helicopters on Navy ships in a virtual environment for training purposes. A computer vision technique was used for acquiring 17 dynamic and static hand signals. Several restrictions (constant, distinguished background, colourful external markers on hands and body) were put in place to improve segmentation and

image processing tasks. The study used HMM as the recognition algorithm. The study reported an approximately 80 % satisfactory rate among students who tested the system in a class environment.

3.4 Summary

Throughout the last few decades, the gesture recognition community along with other researches in temporal pattern recognition (speech, hand writing recognition), have investigated and implemented various theoretical techniques and practical systems. Although, these systems and theories have their limitations, they have produced valuable knowledge and resources for further researches. These can be summarised as follows:

- System Framework: A general pattern recognition system approach is suitable for dynamic and static gesture recognition systems. Sensor processing, analysing and modelling components highly effect the recognition component. A distinctive and compact feature set should be used for modelling and better recognition. But prior to construction of the feature set, if it is possible (unlike off-line mode, in on-line, real-time mode, available data would be small or incomplete), the raw data must be pre-processed, normalized, smoothed and transferred. The recognition machine proposed in this study is based on a generic pattern recognition framework.

- Input Device : While computer vision based (CV) input devices are more natural than point tracker/glove (PT) based devices, the data obtained from a CV based device needs more time and space resources for pre-processing. Furthermore, in many cases, even state-of-art pre-processing techniques do not obtain as accurate and reliable data as tracker/glove-based devices. Therefore, in this study, emphasis has been given to tracker-based input devices because of their effectiveness. But later the study is extended to validate a recognition machine on image-based FDO gestures collected by a simple web cam.

- For recognition algorithms, several techniques with various degrees of success have been implemented. For static gesture recognition, robust and effective algorithms have been developed. But in continuous dynamic gesture recognition, research is still ongoing for improved performance.

- A neural Network approach is one of the commonly used techniques for isolated/continuous and static/dynamic gestures. For static gestures multi-layer

feed forward networks are mainly used. For continuous or isolated dynamic gestures recurrent or other temporal networks are mostly preferred.

- NN is a discriminative and black box system: NNs have several powerful advantages such as ability to learn from samples; ability to approximate any linear/nonlinear functions, deploy negative and positive data to accommodate discrimination and not be sensitive to the noisy data. But despite these advantages, training of NN requires substantial time and expertise. In addition, NNs are a black box system, namely, the knowledge in the network is not easily interpretable. In most cases, these disadvantages overshadow the advantages of NN. Therefore, nowadays, NN is used more often as part of a hybrid system where the advantages of NN have been utilised.

- Static NN: A Multi-layer perceptron and Radial Basis network are used for isolated static gesture recognition. These networks are not capable of recalling and utilising the historic sequential data in internal weights. But they are capable of robust approximation, quantization, clustering and discrimination. MLP and RBF are commonly used for static gesture recognition.

- Feed-forward Temporal Networks: One of the methods utilised for temporal recognition is to employ delay tap lines or buffers either in the input (tap delayed NN) or hidden layers (TDNN). These architectures suffer from high temporal variances and their time and space complexity are high in the case of long sequences.

- Temporal NN: Hence RNN (SRN, BPTT, RTRL, LSTM), which are capable of memorising sequential data, are proposed for temporal pattern recognition, and consequently for dynamic gesture recognition. RNNs are more powerful but they have chaotic behaviour. In addition to that, interpretation of their results and training tasks are challenging. In this study, SRN based architecture (Elman) will be investigated empirically over a real world dataset to assess RNN for gesture recognition.

- Markovian process: HMMs superiority comes from its success on a markovian process-based sequential decision analysis for temporal patterns. But it suffers from a lack of sufficient discriminating power, relies on only positive data, and assumes an underlying distribution for emission probabilities. It also suffers from representing undefined movements in case of continuous gesture recognition. Its

challenge is to model a universal HMM for connected, undefined movements between gestures.

- Time Warping: DTW is capable of dealing with different lengths of signal by taking care of distortions along the time axis. But slowness, template construction, common distance unit between templates are some of the disadvantages. DTW is ideally suited for off-line recognition tasks.

- United front: Hybrid systems have been proposed to combine the advantages of the powerful aspects of HMM, NN and other techniques. NNs can be used for discriminative feature extraction, emission probability estimation in case of even noisy and unseen data. On the other hand, HMM is used for temporal modelling. In this study, a hybrid approach architecture consisting of NN and Markovian-based state automata is also investigated empirically as a variant of a developed recognition machine (RM). The NN part in a hybrid system is based on multi-layer perceptron and is responsible for estimating transition and emission probabilities, whereas a Markovian-based state automata is responsible for modelling the temporal characteristics of gestures.

- Applications: Table 3.1 and 3.2 summarise some the gesture recognition applications in literature with deployed techniques for recognition algorithms and data acquisition, dataset type (static, dynamic gestures), vocabulary size and recognition rate. While, DTW suffers even on static gestures, hybrid systems perform better recognition.

Authors	Dataset	# Class	Algorithm/Sensor	$Rate_S$ %	Remarks and Limitations
Marcel, [75]	static, Triesch	10	MLP/CV	70	Designed only for static gesture and has poor performance
Fels, [33]	static, ASL	66	MLP/Glove	99.42	Designed for only static posture
Wysoski, [151]	Static ASL	26	MLP+DP/CV	96.7	DP (Only Static Gestures and Dynamic Programming obtains better results (98.8).)
Yang, [156]	Dynamic/Static ASL	40	TDNN/CV	93.42	Continuous recognition is skipped
Cyrus, [114]	Dynamic/Static ASL	10	TDNN/CV	79.82	Does not obtain good performance and meaningfull output as much as Bayesian network (84.64 %).
Murakami, [82]	Dynamic Japonese SL	10	Elman/Glove	96	So slow to train and focused on only isolated recognition.
Corradini, [19]	Drawing gestures	4	Jordan/PT	84	Only off-line, segmented and small vocabulary size
Vamplew [137]	SLARTI	16	Elman/Glove	98.9	Isolated recognition
Yang, [154]	Digits	9	HMM/Mouse	99.78	Off-line and isolated recognition
Lee, [68]	Static/Dynamic Hand Gestures	14	HMM/Glove	Good	Isolated and off-line recognition. High disparity and unambigious among dataset.
Rigoll; [101, 29]	Dynamic/Isolated	24	HMM/CV	92.9	Prior knowledge of background; Start/End problem in continuous recognition for long and noisy sentences.

Table 3.1: Gesture recognition algorithm in literature with deployed recognition algorithm (HMM, Elman, Jordan, DTW, SOM, RNN, RBF,MLP, TDNN), data acquisition method (Sensor: Glove, mouse, Computer Vision (CV)), dataset type (static, dynamic gestures), vocabulary size, recognition rate and some remarks.

Authors	Dataset	# Class	Algorithm/Sensor	Rate %	Remarks and Limitations
Starner, [122, 123]	Dynamic/Static ASL	40	HMM/CV	91.3	Single user. In case of multiple user system would perform worse;Marked hands to segment hand positions
Corradini, [17]	Static ASL	5	DTW/CV	92	Isolated, off-line recognition
Greenspan, [71]	Continuous Video Gesture	8	DTW/CV	88.3	Limited temporal variance (Constant speed)
Yiqiang; [14]	Isolated Chinese SL	65	Baysian/Glove	92.3	Off-line recognition, isolated gesture. NN and C.45 decision tree perform better [114].
Reiter [100]	Dynamic meeting gestures	6	LSTM+RNN+HMH/CV	96.3	Start/End of gestures are explicitly marked.
Corradini, [20]	Dynamic/Static	6	HMM+SOM/CV	84.70	Low performance over on small dataset and isolated recognition
Corradini, [20]	Dynamic/Static	6	HMM+RBF/CV	88	Low performance over on small dataset and isolated recognition
Corradini, [20]	Dynamic/Static	6	DTW/CV	75.86	Low performance over on small dataset and isolated recognition
Craven, [22]	Static alphabet handwriting	26	Rule+MLP/CV	92.5	Off-line and Static recognition

Table 3.2: Gesture recognition algorithm in literature with deployed recognition algorithm (HMM, Elman, Jordan, DTW, SOM, RNN, RBF,MLP, TDNN), data acquisition method (Sensor: Glove, mouse, Computer Vision (CV)), dataset type (static, dynamic gestures), vocabulary size, recognition rate and some remarks.

Chapter 4

Gesture Analysis & Modeling

This chapter focuses on two main topics: construction of class models and intra/inter class model similarity and some discussion of the complexity of datasets with examples.

Class model construction is an off-line process comprising data acquisition, data pre-processing, feature selection/extraction, and construction of class models. Class models are summary representations of the training examples. For construction of class model, a template-based modelling is preferred. Template-based modelling represents the trajectory of classes in a form of features with summary and compact statistical parameters. Features can be directly or indirectly related to either spatial or temporal characteristics of classes. Features are extracted from pre-processed data which is smoothed or transformed from raw data. In the case of FDO gestures, two different approaches have been used for data acquisition, the first of which is based on a tracker input device. The second approach, computer vision, deploys a desktop webcam to gather data. According to data acquisition methods, the FDO dataset is divided into two subgroups, FDO_PT and FDO_CV, for tracker and computer vision-based approaches respectively. These two datasets have common properties apart from the number of samples, data acquisition methods, and the number of users used to perform gestures. While FDO_PT deploys only one user (the author itself), FDO_CT deploys four users.

Channel construction is based on estimating the parameters which represent best the underlying statistical distribution of training data at each time point of the channels. In this study it is assumed that features are independent of each other and training data at a time index in a channel that follows normal statistical distribution. Therefore, the channel construction procedure is based on estimating the parameters of statistical mean μ and standard deviation σ to represent training data at each time index in the channels.

Having constructed the class models, we need to consider the intra/inter similarities between classes and their samples. Therefore, in the second half of the chapter, a comprehensive complexity and similarity analysis of class models are addressed. For this purpose, several well-established techniques, which are mostly developed for static classes, are adapted for temporal classes. In addition to these existing techniques, new, novel complexity and similarity techniques (for example intersection volume and temporal periodic and index of sub-event variance analysis) are also introduced.

Considered complexity and similarity techniques are as follows: entropy analysis for class models and inter similarity between classes models and samples in terms of information complexity; statistical techniques such as Chi-Square, skewness and kurtosis to analyse assumed underlying statistical distribution; Fisher linear discriminant analysis for *within* and *between* class distance ratio; Principal component analysis based EROS (*E*xtended F*ro*benious norm) method for the similarity between classes by utilising the Frobenious norm over eigenvectors and eigenvalues of covariance matrix of samples; and the intersection volume between class models. In addition, temporal variance among the length of samples and in position of sub-events is investigated.

Discussion of these techniques are supported and exemplified over an artificial dataset W_Test, before a detailed analysis on FDO datasets is carried out.

4.1 Modelling Temporal Classes

Throughout every moment of our daily lives, human beings interact with the surrounding environment. Although a human being is bombarded with enormous quantities of information, he/she has the ability to distinguish between useful and senseless information without making so much effort. The method developed here, is constructed by imitation of the human gesture recognition system even tough it has not been properly understood yet.

There are two basic theories (template-based, feature based) described by cognitive psychologists, that try to explain how humans recognize a pattern [31]. Although the success and flexibility of the human being's pattern recognition system is not driven by these theories, they do agree about the main idea behind the pattern recognition process: matching the perceived data with stored data to make predictions [42].

The first theory, template-based theory, goes back to Plato. According to template-based theory, known patterns are stored as a template or form in long-term memory. The pattern recognition process is basically based on matching the stored template with perceived data in order to estimate which template is the closest one.

With regards to the flexibility, the theory does not provide an answer as how to pattern recognition systems of human beings can recognize variances of a pattern with high flexibility. This theory implies that, for each variance of a pattern, there should be a template. But this is not realistic from the point of storing huge amounts of templates, which correspond to a pattern. That solution can be tackled by storing the associated probable areas within templates themselves. In another words, by using auxiliary variables, ideal templates can accommodate the variances of the template.

Whereas, the gist of the second theory, based on features, is that, a pattern is a set of specific features or attributes. For example, an alphanumeric pattern, A, is specified by two straight lines and a connecting cross-bar. Extraction of features from perceived data, construction of the feature set from extracted features, and matching/comparing the set with the sets stored in long-term memory are the phases of the pattern recognition process in feature-based theory.

In this study, template-based class modelling is implemented. The motivation behind this is that template-based models accommodate more information for sequential (incremental) and on-line temporal recognition. Because, each frame or data point is obtained incrementally from input devices in temporal patterns, recognition algorithms can utilise or process the acquired frame and template at each time unit for prediction and recognition as soon as data is acquired. Whereas, feature-based recognition systems work more in off-line mode. It needs more time to build the features. Since one of the aims of this study is to recognize or predict the gesture as soon as on-line (real-time) data is available, template-based approach is preferred.

Another reason behind template modelling is that template-based modelling techniques provide more specific feedback for training purposes. Unlike black box representations (such as neural networks), template-based representation provides detailed information about temporal patterns. Another advantage is that the generation of artificial patterns from templates is a trivial task, which can be used to create tutorials, videos, and animations for training purposes. In addition, template-based representation can also help to make an analogy between existing temporal classifications (HMM and DTW, for instance) and proposed algorithms in this thesis.

The model of temporal classes is constructed in a way to be compatible with five-tuple *(C, L,H, F, B)* notation described in the problem definition chapter 2.2. In other words, a temporal class model, (C_i) is defined in the form of templates, which comprise ϑ L_i-length channels $(H_{i,j})$, in F^{ϑ} feature space and time.

In this study, more precisely, channel $H_{i,j}$, which accommodates possible trajectories of the feature f_j over time L_i is referred to as a template. As explained above, the flexibility problem of template0based modelling is overcome by introducing a mean

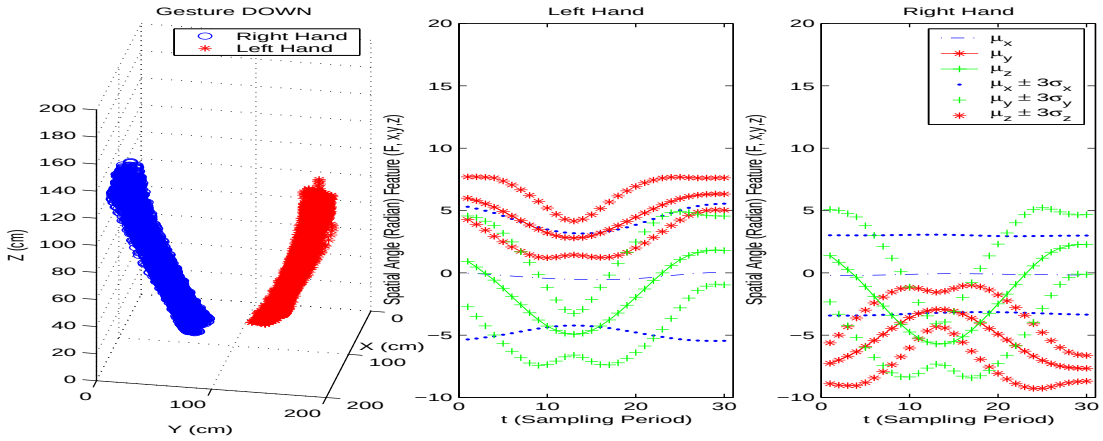


Figure 4.1: Trajectory of Down gesture and its corresponding spatial templates (x, y, z spatial channels). Templates have two components, mean μ and standard deviation σ. The figure shows the mean templates $\mu_{x,y,z}$ for three channels and $\mu_{x,y,z} \pm 3\sigma_{x,y,z}$ band width which corresponds to 95 % confidence interval.

(μ) template and its auxiliary variables, standard deviation (σ). Mean templates indicate the ideal templates, whereas standard deviation (σ) accommodates possible band width in which the variances of the ideal template could occur. Therefore, a channel can be considered as a two-component template ($\mu \mp \sigma$).

Figure 4.1 illustrates the trajectory of Down gestures and their associated spatial channels in the FDO dataset. The following convention is used to illustrate a channel or a template of a class in this thesis. The vertical axis (y) corresponds to the spatial or temporal feature of interest and the horizontal axis (x) corresponds to the sampling period (L) of the class of interest. For example, in figure 4.1, angular features for all the three channels, measured in radians are shown along the vertical axis. The x axis represents sampling period.

Class models are constructed out of raw training data with the following procedure: First, data are pre-processed which includes smoothing and transformation. Then, distinctive and concise features are determined from raw data. The main idea behind this construction of templates is to represent training samples at each time point with a statistical distribution (normal). Therefore, before estimating the statistical distribution parameters (mean, μ, standard deviation σ), samples are stretched and compressed to have identical length (class period, L_i). In addition, sub-events are aligned in order to obtain more a meaningful mean template and narrow band width (standard deviation).

In the following section, this class modelling procedure will be elaborated in more detail for the FDO dataset.

Figure 4.2: Input Devices for FDO gestures. Tracker-based Polhemus FasTrack (left) is used for acquiring for FDO_PT dataset. Whereas for computer vision-based data polling, an average desktop webcam (right)is used. Even though, Polhemus FasTrack provides four sensors, only two sensors are used, each of which is used for a hand.

4.2 Data Acquisition and Pre-Processing

Two different data acquisition schemes have been implemented for FDO gestures, the first of which is tracker-based and the second one is computer vision-based. Hitherto, due to their acquisition scheme, FDO gestures have been investigated under two different dataset names: FDO_PT (Tracker based) and FDO_CV (Computer Vision based). For tracker and computer vision-based approaches Polhemus FasTrak and an average desktop webcam are deployed as respective input devices. Figure 4.2 illustrates the devices. In the following subsection 4.2.1, the detailed discussion of input devices and acquisition schemes are covered.

Prior to discussion of these datasets separately, here are some appropriate remarks regarding their common and distinctive properties. As it was explained in the Problem Definition chapter of this thesis, 18 gestures of the Flight Deck officer were looked at. Appendix B illustrates all possible marshalling signals for the FDO and pilot. Figure 2.6 in the Problem Definition chapter also shows the subset of FDO gestures considered in this study. Please refer to the Problem Definition chapter for a comprehensive discussion about FDO datasets.

Another major difference between the FDO_PT and FDO_CV dataset is the number of users used to collect the data. For the FDO_PT dataset only the author performed the gestures, while for FDO_CV, four users were employed in order to analyse the performance of the proposed algorithm in case of multiple users. The users' physical appearance accommodates large variances, such as their heights [161cm, 169cm, 173cm, and 179cm].

FDO_PT is collected via a tracker device (Polhemus FasTrak) of which two sensors acquire the position of hands in a three dimensional coordinate system (x, y, z). Whereas, in computer vision, only position of hands (x, y) in two dimensional Cartesian space are utilised. FDO_PT and FDO_CV consist of approximately 150 and 75 samples for each class, respectively.

4.2.1 Data Collection Using Tracker-Based FDO_PT Dataset

4.2.1.1 Input Device (Polhemus Fastrak) for FDO_PT Dataset

As the input device, the tracking-based Polhemus Fastrak device is used [50]. The device includes a base unit, a transmitter and up to four receivers. The device works based on an electro magnetic field which is created by transmitter and its force at the point of the receivers is used to calculate the coordinates and the orientation of the receivers. The device has four sensors, for each hand, and only one of them is used. The receivers indicate the coordinates of hands.

The device provides six different data, the first three of which are Cartesian coordinates *(x,y,z)* and the last three are orientations(azimuth, elevation, and roll). Although six data are provided by the system, just Cartesian coordinates *(x,y,z)* are utilised for the recognition system. The update rate of the device (documented in specification of the device), is 120 samples (data point) per second is divided by the number of receivers. In the system, two receivers are used, therefore, the update rate is 60. But in practice, it is observed that, the sample rate is around 20 due to high noise in the environment.

The device is programmed by using C programming language and RS232 serial communication protocol. The Maverik [48] virtual environment tool is used to build an interface to collect and arrange the data. For more information about how data is collected using the interface, please refer to the Appendix relating to Polhemus FasTrak or previous internal technical reports [117] and [118].

4.2.1.2 Issues About The Tracking Device

Even though, the Polhemus Fastrak tracking device is appropriate for a real-time recognition system, it has some issues such as magnetic field and coverage. At present these devices are tethered and as such limits the user's movement and his/her locations. Since the device is based on a magnetic field, the environment in which the data is collected must be isolated from other magnetic fields as much as possible. Isolation

Figure 4.3: Virtual Environment in which FDO_PT data is captured. Virtual environment is created by openGL based Maverik toolkit.

of the environment from other magnetic fields is non-trivial, consequently, raw data includes some noise. In order to reduce the noise, the device's filter option is calibrated to high. In addition, transmitter and receivers should not be located near metallic objects.

The second issue about the device is coverage limitation. The system gives best performance when receivers are within 30 inches of the transmitter. Because some of the arm-based gestures are performed outside of this volume, the performance of the system is reduced. Furthermore, even though the device supports being 10 feet within the standard transmitter; special attention has to be paid to checking the boundary. If the receivers go out of a 10x10x10-feet-volume, the device does not return robust data. For example, it is observed that some samples of Wave Off, Up and Engage gestures go beyond the boundaries,thereby reducing the performance of the recognition.

The Appendix relating to the Polhemus FasTrak illustrates the error table C.2, which contains symptoms and possible solutions in some of the problem.

4.2.1.3 Data Collection Process in FDO_PT dataset

In order to collect the data FDO_PT dataset, a virtual environment interface is created using the openGL based MAVERIK (the MAnchester Virtual EnviRonment Interface Kernel) [48]. From the perspective of this thesis and data collection, the following observations are worth emphasizing here: the virtual environment serves as an interface to capture raw data from the Polhemus Fastrak and to store them in files, to be analysed

#	G	x	y	z	SE
⋮	⋮	⋮	⋮	⋮	⋮
410	3	70.92	172.57	72.70	1
411	1	56.90	11.79	77.86	0
412	3	69.80	173.08	79.03	0
413	1	54.14	12.74	86.73	1
414	3	65.70	172.15	88.05	1
415	1	50.14	17.96	104.26	1
⋮	⋮	⋮	⋮	⋮	⋮
438	3	71.58	168.96	94.79	1
439	1	53.02	15.09	96.31	1
440	3	67.35	174.14	84.44	1
441	1	55.04	12.63	84.87	0
442	3	69.06	174.02	78.70	0
443	1	55.13	11.62	78.90	1
⋮	⋮	⋮	⋮	⋮	⋮

Figure 4.4: Training Data File (TDF) Format. TDF accommodates many samples (cycles) of a gesture class (G) and it is self descriptive, namely consists of all required information, such as origin(G_c^{h1}), boundaries (G_c^{h2}) and start/end of point of samples, in order to compute channels automatically.

and processed later. Actually, this interface provides a prototype of a whole gesture recognition system, in which raw data is retrieved, pre-processed, and passed to a Matlab-based gesture recognition module, and its on-line results, namely recognised gesture (C_R) is fed to helicopter simulation to implement the recognized gesture C_R directive (for example, if it is *Left* gesture, a helicopter moves to left). Figure 4.3 illustrates a screen shot from the virtual environment. Please refer to [117] for a comprehensive discussion on virtual environment interfaces.

Data from the FDO_PT dataset is collected in a systematic way by the author through the virtual environment interface. The main outlines of the data collection process are as follows: First the user indicates some reference points (new origin and boundaries of Cartesian space) for further channel computation. Then, the user repetitively performs the cycles (samples of the gesture) serially by specifying the start/end point of each cycle.

The raw data (x, y, z) of the gestures are saved into ASCII text files, named as training data file (TDF) in the thesis, with a predefined format. TDF consists of three parts, two header sections and a body. The header sections hold information for further channel information, which is special to its TDF file. The body part contains several cycles (samples) of gestures. Figure 4.4, depicts the content of an example of a TDF

file.

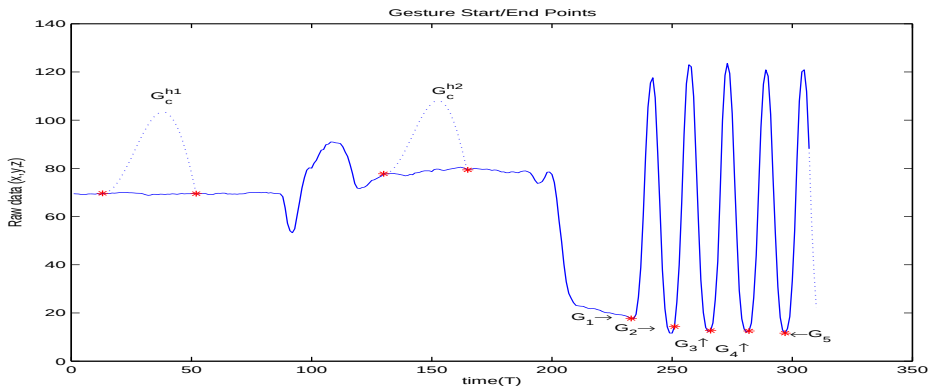


Figure 4.5: Structure of a TDF on x raw data. G_c^{h1} and G_c^{h2} are the header information and indicate auxiliary information (origin point and boundaries) for channel construction. TDF consists of more than 50 samples of a gesture serially. Start/End frames of samples are explicitly marked by the mouse during the data collection. This data corresponds to *Back* gesture.

A TDF contains six columns, the first of which (#) indicates the order of data, the second one *(R)* is the receiver number (1 is the left hand, 3 is the right hand), from the third to fifth columns contain the coordinates of receivers *(x, y, z)* respectively and the last column *(SE)* represents the starting point of a cycle (sample) data gesture. The last column is used to construct gesture templates. 0 in the last column represents a starting point of a gesture, whereas 1 indicates the other points of a gesture. The starting points are indicated by the user explicitly pressing the left button of the mouse. It should be born in mind that the last column is only valid when training data is collected. In test cases, the last column is not used. Figure 4.5 illustrates the structure of the TDF file over the x raw data for a hand (third column of TDF file). Note that in figure 4.5, G actually corresponds to a channel ($H_{i,x}$). TDF has the following properties:

- A TDF accommodates several samples (cycles) of a given gesture class (G).
- A TDF is self descriptive, namely consists of all required data to serve to construct models of the gesture C.
- A TDF consists of several gesture cycles (samples) (average 50). The length of cycles are different and the start/end points are marked in the last column, Since all cycles are performed sequentially, while 0 in the last column (SE) indicate the starting of cycle *i* G_c^i, it can also indicates the end point of the previous cycle *i-1*, G_c^{i-1}.

- The start/end points of each cycle is specified explicitly by pressing the left button of the mouse when the gesture G is performed in the Maverik virtual environment.
- The first two cycles are known as header cycles (G_c^{h1} new origin, G_c^{h2} arm length) and are used for feature construction, which will be explained in the next section.
- All cycles except header cycles are performed sequentially without a break.
- G_c^{h2} is the length of arm or the boundary of Cartesian space.
- Average of G_c^{h1} and G_c^{h2} indicates the origin point(O) and boundaries of Cartesian space for all samples of gesture in TDF.
- Since TDFs are independent and self descriptive from each other, a new TDF can be added to the system automatically.
- Data (x, y, z)in TDF is raw data and has to be passed to the data pre-processing component before channel construction.

The following procedure is carried out in order for the Maverik interface (Figure 4.3) to construct a TDF for gesture G.

1. Specify the origin point O (G_c^{h1}): In this initial phase, origin is specified for the TDF file as follows (Note that this origin is used for channel construction later): First, both sensors (hands) are located at a preferred origin point O (bottom of the neck for example). Then, after explicitly marking the beginning of the origin data (G_c^{h1}), the data is started to be collected for about 6-7 seconds. The acquisition of the origin header is terminated again explicitly by marking its end by pressing the left button of the mouse. Note that the explicit marking of start/end gestures is carried out by pressing the left button of the mouse in the virtual environment interface, because the Polhemus FasTrak does not accommodate any explicit marking.
2. Specify the boundary of the performing volume (G_c^{h2}): Arms are stretched out to the initial position of the *Up* or *Down* gestures, in order to measure the length of the arm, consequently the boundary of performing volume. Subsequent explicit marking by the mouse in this study, boundary header data G_c^{h2} is collected for 3-4 seconds. Then, once again, the collection is terminated by an explicit marking.
3. After that, hands are stationed at the initial position of the Gesture G of interest.

4. Just before performing the cycles the start point of the first cycle is explicitly marked.

5. Then immediately, a gesture cycle is performed (G_c^i).

6. At the end of the cycle (G_c^i), the end point of the cycle is marked again explicitly. Note that this end point is also the starting point of the next cycle.

7. Repeat the steps 5 - 6 to add more cycles to the TDF if desired or go to last step.

8. Stop the Maverik Interface collecting the training data.

TDFs contain only raw data (x, y, z). Figure 4.6 and 4.7 show the trajectory of raw data which is stored in TDF for the FDO_PT dataset. FDO_PT and FDO_CV datasets are available on-line from the following web address with a descriptive article, and MATLAB scripts to extract data from the samples file.

http://personal.rmcs.cranfield.ac.uk/~turand

As explained above, the Polhemus Fastrak has some issues (noise, limited coverage volume), which causes unreliable, noisy data. For example, some samples of Wave Off gestures include sudden, discontinuous large jumps in the trajectories, because of the limited coverage of the Polhemus Fastrak. The smoothing operation does not eliminate this type of noise in the data. Therefore, these abnormal samples have to be extracted from the FDO_PT dataset manually. Please note that the FDO_CV dataset also includes some abnormal data (figures 4.9, 4.10, 4.11) because of user's mistakes. The abnormal samples in Polhemus Fastrak are highly discontinuous and unnatural, which are largely introduced by the device. Whereas, abnormal samples in FDO_CV are more plausible. Therefore, these abnormal samples in FDO_CV are retained to validate the proposed recognition algorithm.

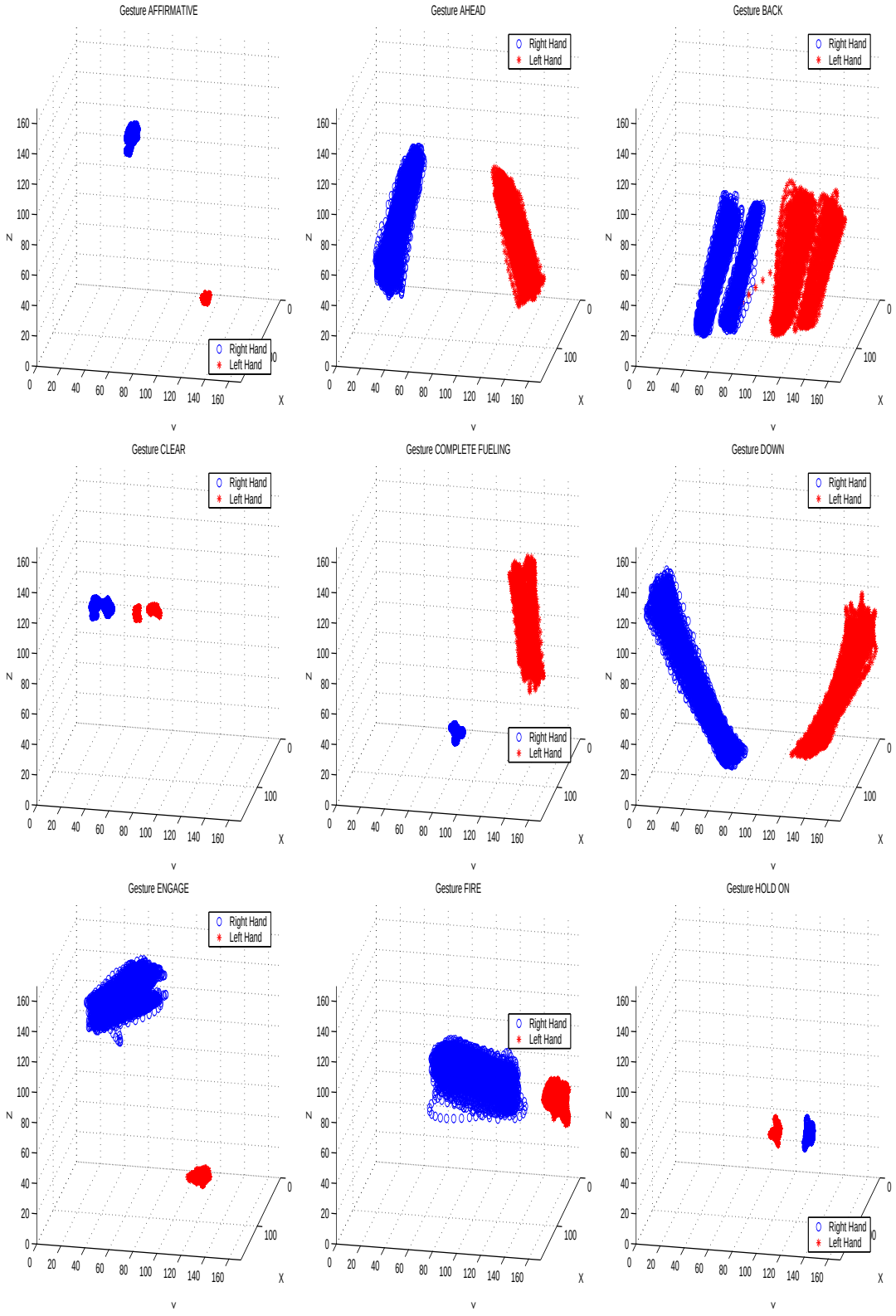


Figure 4.6: FDO_PT Gesture Trajectories 1-9

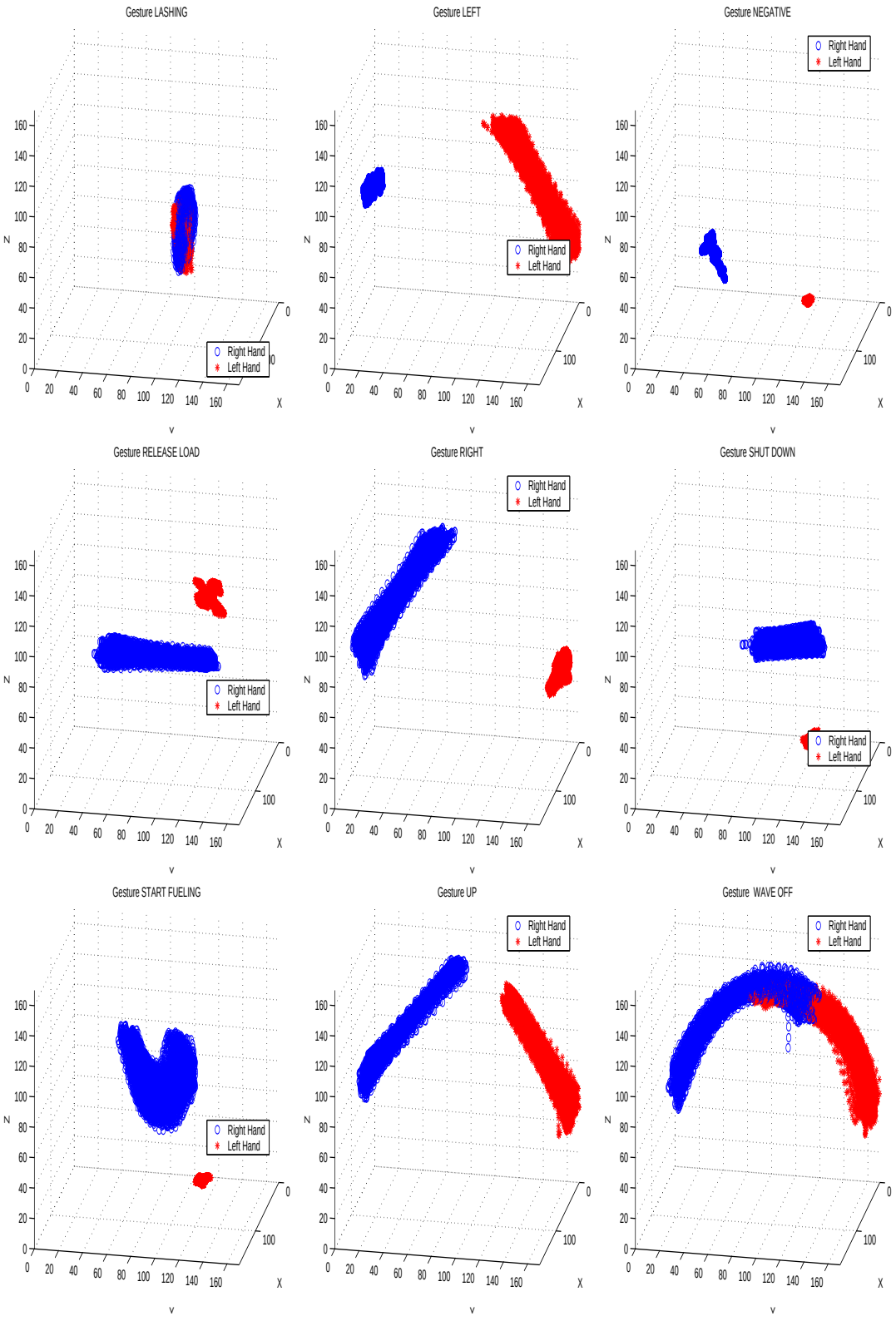


Figure 4.7: FDO_PT Gesture Trajectories 10-18

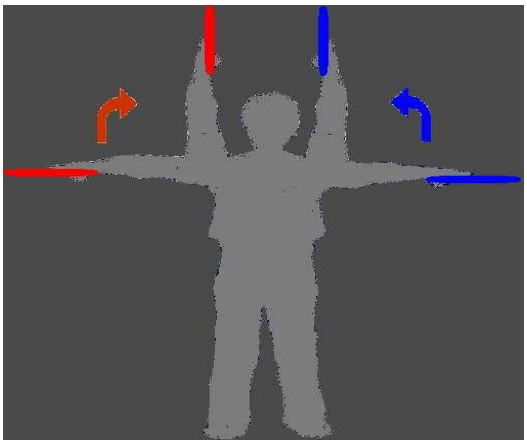


Figure 4.8: Data collection setup for the FDO_CV dataset. Red and Blue light stick in a dark room is used to perform gestures, in order to simulate night FDO gestures. The user is centred in the middle of the image to stretch his arms fully in the image size. In this figure, the user's image and light sticks are highlighted to show the setup in a dark room.

4.2.2 Data Collection in Computer Vision-based FDO_CV Dataset

The FDO_CV dataset consists of the same subset of 18 gestures as FDO_PT dataset. Four different users are employed for the data collection.

For data acquisition of the FDO_CV dataset, an average desktop web cam (Logitech QuickCam Express) is deployed. The webcam is set to 480×640 pixels with 12 fps for coloured video capturing. Figure 4.8 illustrates the setup for the FDO_CV data collection. For this dataset, it is assumed that FDO is performing the gestures in a night/foggy scenario. Therefore, the gestures are performed in a dark room with the user holding coloured light sticks (red and blue for right and left hand respectively) in each hand. In addition, as figure 4.8 shows, the user is positioned at the centre of the image, in order to cover the image width when stretching his arms either horizontally or vertically.

Bear in mind, in addition to implementing a night scenario of FDO, the data collection setup also aims to tackle the following two major issues, hand segmentation and handling physical variance. Since different coloured light sticks are used in the dark room, the segmentation of hand positions in the videos will be more trivial. Positioning users at the centre of the image to cover the window length serve to tackle physical variance among the users. In other words, the user is bounded into the window.

Video acquired from the webcam is pre-processed to compute the x and y positions of the middle points the light sticks for each sample of gestures. Unlike FDO_PT, each

sample of FDO_CV data is stored in a separate file.

Figures 4.9, 4.10 and 4.11 show the trajectories of FDO_CV dataset. Each gesture has approximately 75 samples. Only three users' trajectories are displayed for the sake of clarity. But it can be noted that the fourth user's trajectories have similar properties to the first user. The figures also illustrate some abnormal samples, for example in the *Right* gesture of the second user. These are due to the user's faults. However, these samples are kept to verify the performance of the recognition algorithms.

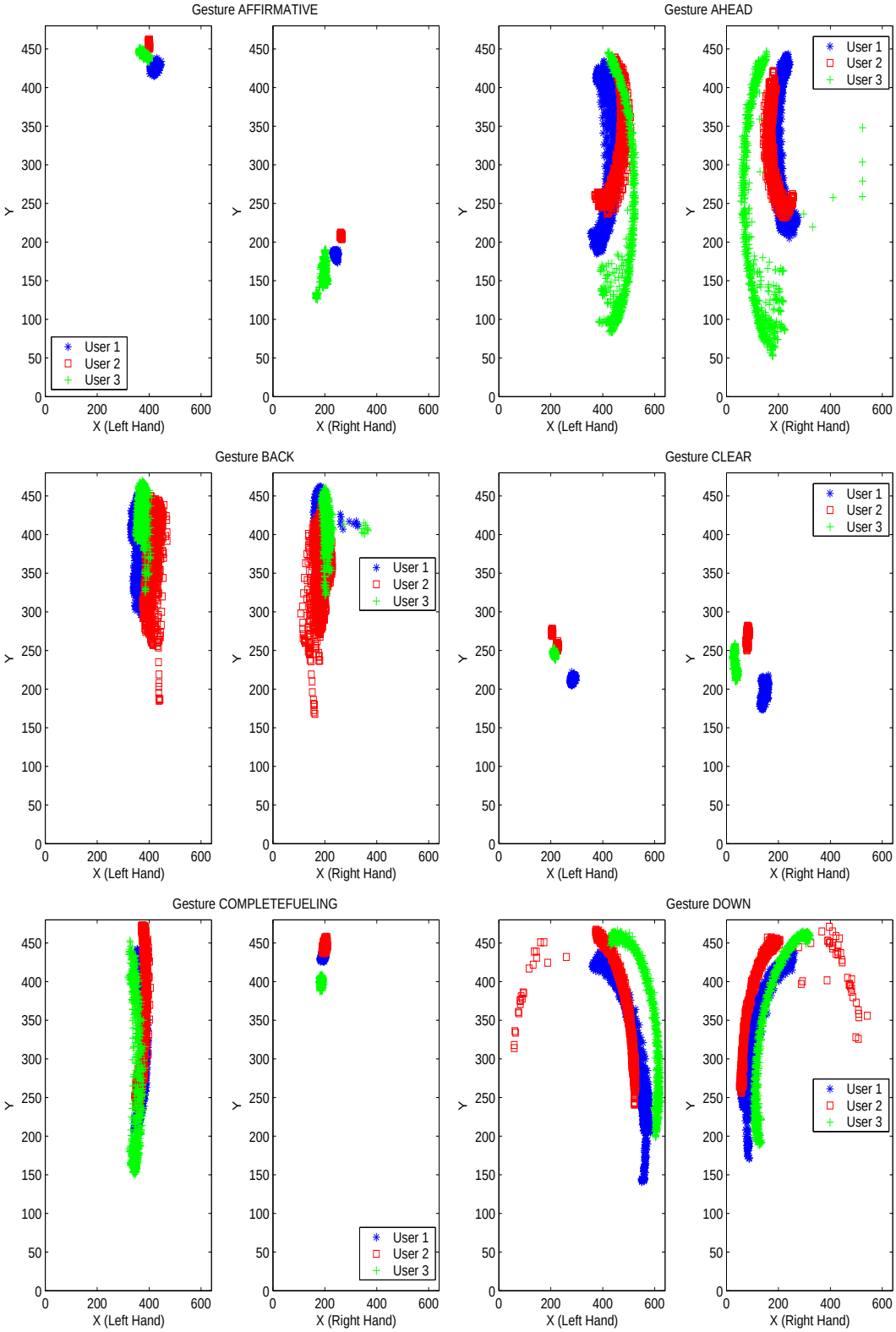


Figure 4.9: FDO_CV Trajectories 1-6

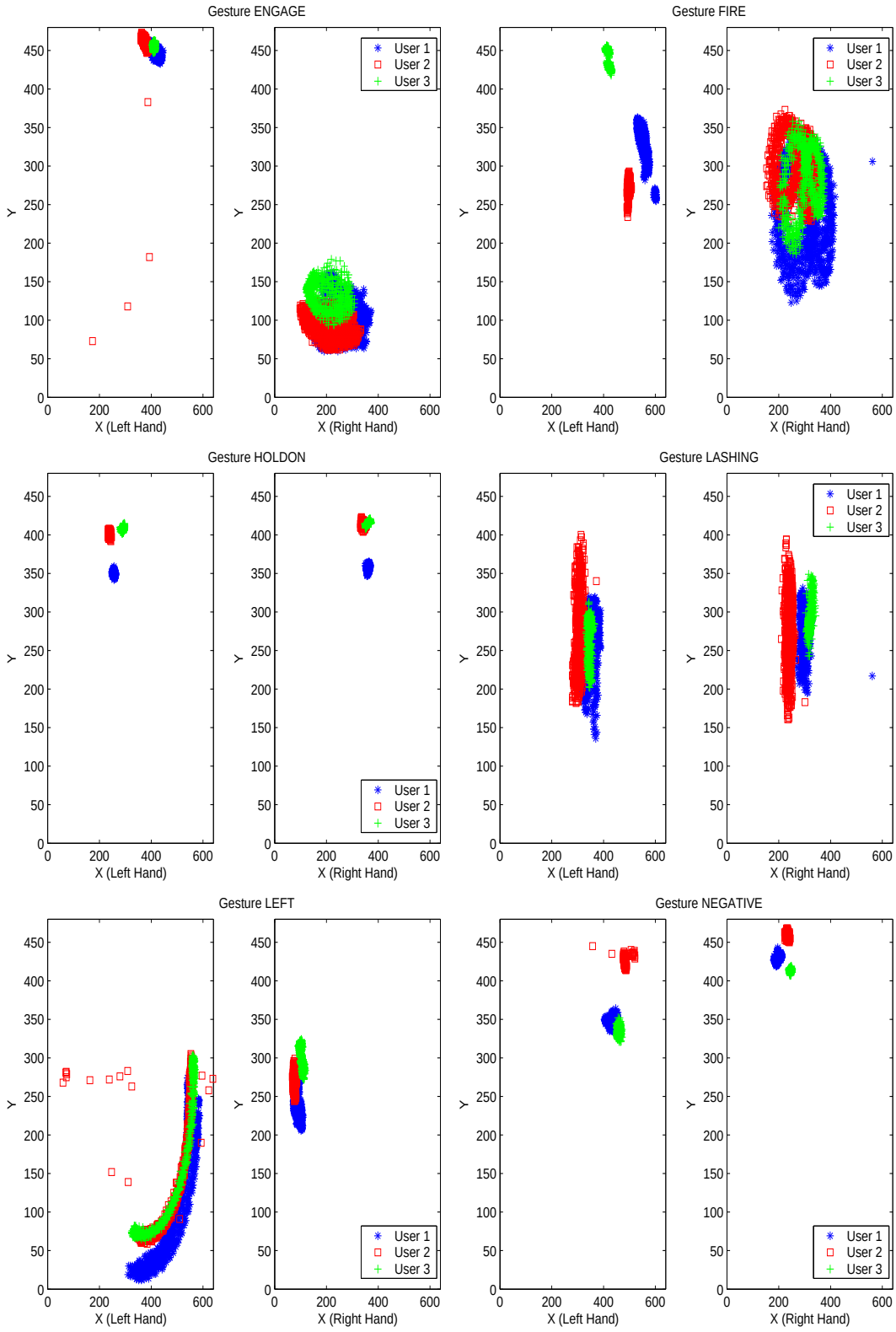


Figure 4.10: FDO_CV Trajectories 7-12

4.2.3 Pre-processing: Data Smoothing

As was explained earlier, input devices, such as Polhemus Fastrack, accommodate huge amounts of noise although filter settings are used and necessary pre-cautions are taken. Therefore, before channels are constructions, the raw data is smoothed in order to reduce the noise.

Since one of the aims of this project is to recognize gestures in an on-line mode, for smoothing operations, only a small portion (window width) of incremental data is utilised. The smoothing operation is simply based on taking the weighted average of data around $x(t)$ within a certain window band width $(2 * N + 1)$. The data point $x(t)$ of a channel at the time of t is smoothed as follows:

$$x(t) = \frac{1}{(2N+1)\sum_{i=0}^{2N} w(i)} \sum_{i=0}^{2N} w(i)x(t-N+i) \quad N > 0 \tag{4.1}$$

w(i) shows the smoothing weight vector with length of $(2N + 1)$. Note that the choice of window width and smoothing weight vector (w) affects the magnitude of sharp and peaky sub-events significantly. For example, a long bandwidth can completely eliminate peaky sub-events. But on the hand, a narrow window width would not smooth the noisy data.

Based on trial-error experiments, in this study, the window width (N) and its smoothing weight vector (w) are taken as 3 and w=[1,1,2,4,2,1,1], respectively. Note that the weight of x(t), 4 in the smoothing weight vector has more weight compared to others, because it is aimed to address the current point x(t) more strongly.The smoothing operation is applied independently on each raw channel data.

4.3 Feature Selection & Extraction

Feature selection and extraction are the backbones of any classification system. The performance of the classifier depends greatly on the quality and quantity of the features [8]. Therefore, a good quality of time and consideration has to be paid to the feature analysis task. As is mentioned in the feature analysis section of the literature review chapter, there is a strong correlation between the number of features and sample size for an accurate estimation of the true parameters of models.

This issue is investigated thoroughly in statistical pattern recognition literature under the title of *Curse of Dimensionality* and *Peaking Phenomenon* [52]. Briefly, these topics states that if the number of features increases, in order to estimate the

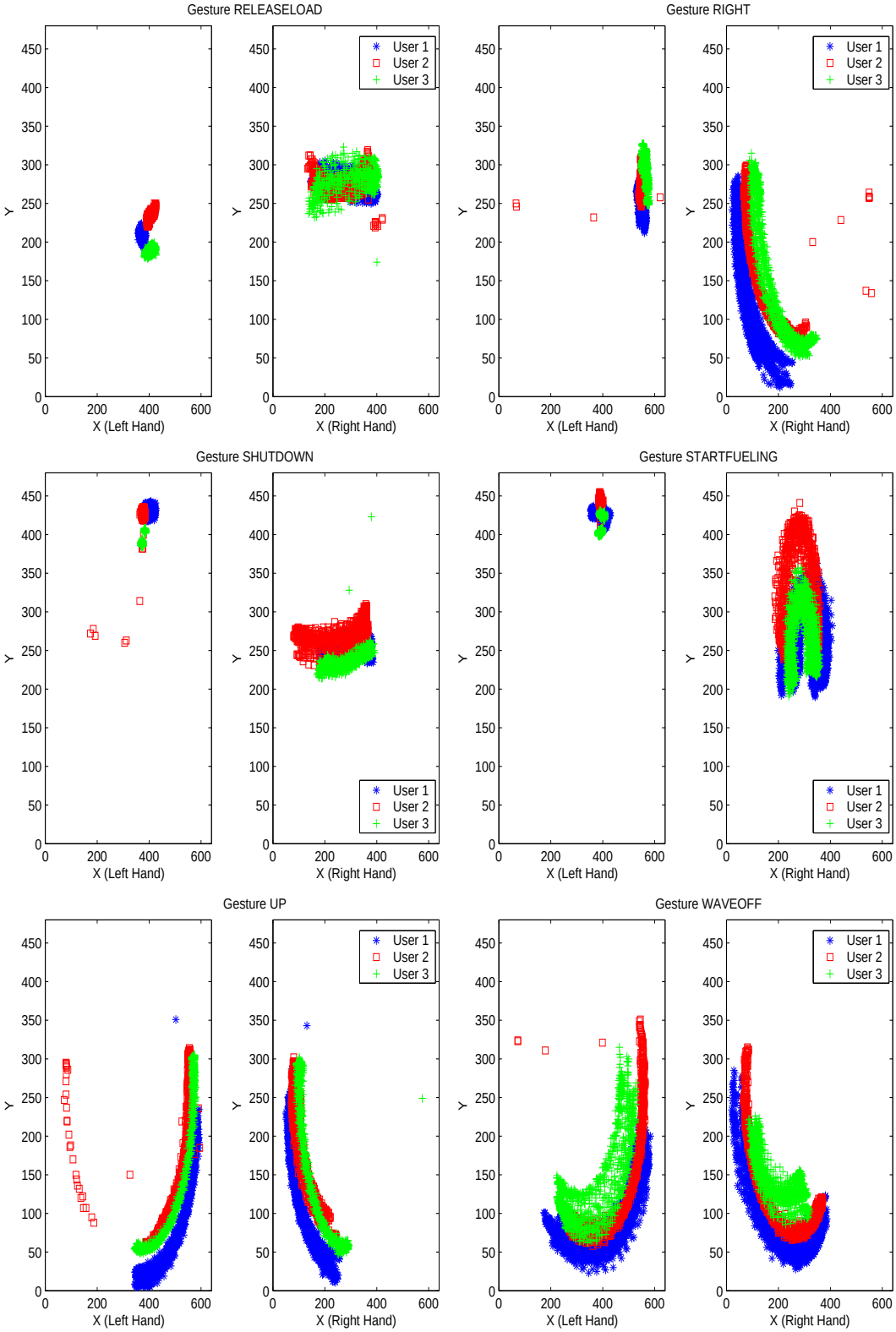


Figure 4.11: FDO_CV Trajectories 13-18

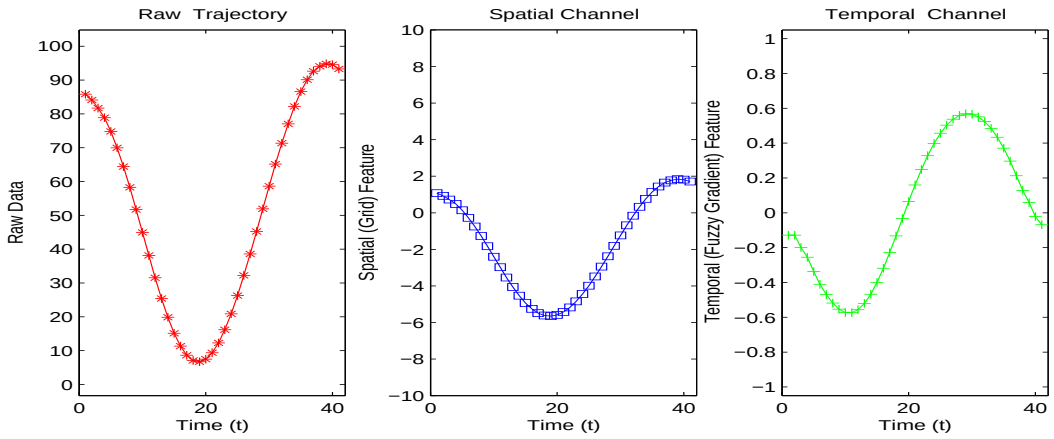


Figure 4.12: Examples of two major features: Spatial (grid based) and Temporal (fuzzy gradient of grid feature) of a raw data trajectory.

true parameters of models required, the sample size also increases exponentially. In terms of the quantity of the sample size, it is suggested to employ 10 times as much as the number of features in literature [52].

On the other hand, in terms of the quality of the feature set, there is no such rule of thumb. But it is agreed that the features must be distinctive, compact, expressive and robust to noise. The best way of verifying the quality of the feature sets is to test and compare them on the classifiers [52, 45]. Therefore, in this study, for the FDO_PT dataset, three different feature sets are employed and the best one is used as the feature set for the FDO_PT dataset thorought the rest of the thesis.

Features are domain dependent. Hence, in literature, a wide range of feature sets is used for temporal pattern recognitions. These feature sets can be discussed under two general titles: The first and most important one is spatial features which represent the locations of the data in ϑ dimensional feature space. More specifically, spatial features are in the form of statistical parameters (for example, in the case of normal distribution, mean μ and standard deviation σ) which indicate probable areas where the class may be in. The boundaries of probable areas are determined via statistical methods (mean and standard derivation).

The second major generic features are time dependent temporal features - gradient, velocity or acceleration. Temporal features indicate the dynamic behaviour of the spatial information (features) such as direction (fuzzy or piecewise gradients, (decrease, stable, increase -1,0,1)) or angular velocity. Figure 4.12 depicts the spatial and temporal channels of a raw channel based on spatial (grid) and temporal (fuzzy gradient) features.

Calculation of the spatial feature will be discussed in further detail at a later stage. But fuzzy gradient features (x') of a spatial feature (x) is discussed here. Fuzzy gradient features accommodate the direction of consecutive spatial features such as whether it is increasing (1), decreasing (-1) or between them [-1, 1] in spatial feature space. In some cases (especially at the beginning, end and transition from one gesture to another) gestures are performed either in a decreasing or increasing manner. These temporal behaviours are represented as fuzzy gradients between -1 and 1. Note that some experiments are conducted over a binary gradient (no fuzzy representation, either increasing (1) or decreasing (-1)). The result of these experiments are far away from satisfactory, hence this gradient representation is omitted. It is assumed that if the gradient Δ between two consecutive points in a spatial feature is bigger than a threshold (τ), it is assumed increasing (1). Similarly if the gradient is less than $-\tau$, it is taken as decreasing (-1). If the gradient is between $-\tau$ and τ, the gradient is mapped to between -1 and 1. These can be formulated as follow:

$$\Delta = x_{t+1} - x_t \tag{4.2}$$

$$x'_t = \begin{cases} 1 & \text{if } \Delta \geq \tau \\ -1 & \text{if } \Delta \leq -\tau \\ \frac{\Delta}{\tau} & \text{if } -\tau < \Delta < \tau \end{cases} \tag{4.3}$$

Threshold τ value is set empirically for each spatial feature, as there is no robust universal method to determine it in an unsupervised manner. Another point that should be noted here is that while statistical distribution properties of gradient values between $-\tau$ and τ are preserved, on the outside of threshold areas, since gradients are pruned at these points, the real underlying statistical distribution of the gradient is hampered. However, in this study, gradient features are considered as normal distribution and represented with a mean and standard deviation as will be explained further in the channel construction section.

4.3.1 Feature Analysis For FDO_PT

As was explained above, the best method for feature selection discussed in literature is to test the feature set on the classifier of the interest. Therefore, three different feature sets are investigated for the FDO_PT dataset: Raw spatial data, angular spatial and its temporal features and grid spatial and its temporal features. In the following section, these features will firstly be elaborated in detail. Later, a discussion will be held about

which feature set performs best and then will be used as the main feature set of the FDO_PT dataset.

4.3.1.1 Raw Data Features Analysis for FDO_PT

For this dataset, the raw data of the FDO_PT dataset is the used feature set. Recall that, the FDO_PT input device acquires three coordinate data (x, y, z) for each hand. So, the feature vector consists of six raw coordinate data (rR_x, rR_y, rR_z for the right hand and rL_x, rL_y, rL_z for the left hand).

$$F_{Raw} = [rR_x, rR_y, rR_z, rL_x, rL_y, rL_z]$$

Please remember that a smoothing operation is applied over raw the data, due to high noise caused by Polhemus FasTrak input devices.

4.3.1.2 Angular Feature Analysis For FDO_PT

To overcome the inter and intra personal variances, instead of using raw coordinate data *(x, y, z)*, the angles between hands and planes are proposed as the spatial feature set, with a new origin point O which is just below the neck at shoulder level. For example, in literature, studies [105], [69] and [153] use spatial angular feature sets for gesture recognition. The angles between hand and planes *(xy, yz, xz)* are α, β, γ respectively and are constrained as, $0 \leq \alpha, \beta, \gamma \leq \pi$. The crucial point is that the global coordinate system is mapped onto two different local coordinate systems, *(right, left)* for each hand. The point O, is the origin of both local coordinate systems. Figure 4.13 depicts the two local coordinate systems with the new origin point (O). The angles, α, β and γ are computed as following:

$$\begin{aligned} \vec{u} &= \frac{\vec{V}}{|V|} \\ \alpha &= \arccos(\vec{x}, \vec{u}) \\ \beta &= \arccos(\vec{y}, \vec{u}) \\ \gamma &= \arccos(\vec{z}, \vec{u}) \end{aligned} \tag{4.4}$$

where V is hand's coordinates, $|V|$ is the distance from the origin O to V and $\vec{x}$, $\vec{y}$, $\vec{z}$ are the x, y, z axis's respectively. As was explained in the data collection of the FDO_PT dataset, two header files in TDF record the origin point O and the length of the arm (V) for angular feature computation.

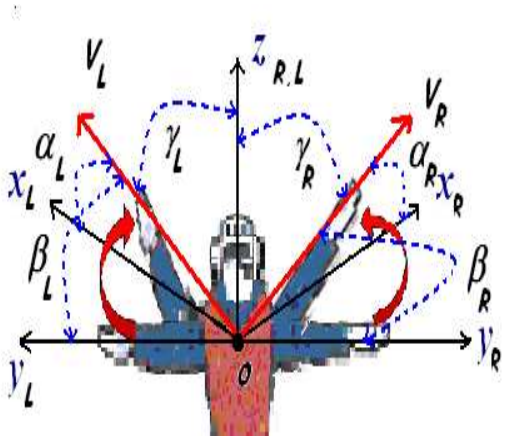


Figure 4.13: Angular features on a transformed local coordinate system. The origin O, is the same for both right and left hand angles. The angles (α, β, γ) between hand and planes *(xy, yz, xz)* are used as spatial features.

With temporal (fuzzy gradient) features of angles, an angular feature vector is chosen of the form :

$$F_{Angular} = [\alpha_R,\ \beta_R,\ \gamma_R,\ \alpha_L,\ \beta_L,\ \gamma_L, \alpha'_R,\ \beta'_R,\ \gamma'_R,\ \alpha'_L,\ \beta'_L,\ \gamma'_L]$$

4.3.1.3 Grid Feature Analysis for FDO_PT

The main idea behind this feature set is that the raw data is normalized according to the physical limitations of the user. For normalization, it is assumed that users are fitted into a fixed $15 \times 15 \times 15$ unit 3D grid cube such that, when the user stretches his arm horizontally or vertically upwards at the shoulder level, the user's hands touch the boundary of the grid cube on the top, left and right edge of the cube. Figure 4.14 illustrates this setup.

As was explained above in the data acquisition section of the FDO_PT dataset, the normalization operation utilises the second header information in TDF to determine the boundary of the grid cube for the user. In addition, the same origin point in the Angular Features is taken as new the origin point for the grid feature system too. According to that feature it is computed as follows:

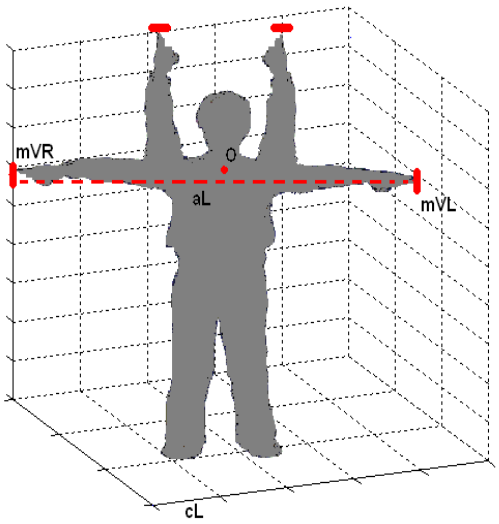


Figure 4.14: Boundaries specified by the user for determining the grid cube. Raw data is normalized by utilising origin point O and stretched arm length (aL) in order to estimate spatial grid features.

$$\begin{aligned} aL &= |mRV - mVL| \\ cL &= \frac{aL}{eL} \\ gF &= \frac{V - O}{cL} \end{aligned}$$

where mVR and mVL correspond to the boundary of the right and left hands' positions; therefore, aL is the length of the stretched arms from one end to the other end; eL is the length of the edges of the cubes, which is taken as 15 in the study; cL is the length of the cell for the user; O is the origin of the new system; and finally gF is the estimated grid features for right and left hand positions indicated by V. Note that all the variables are three dimensional vectors which correspond to a position in Cartesian space.

With temporal (gradient) features of spatial grid features, the grid based feature vector is :

$$F_{Grid} = [R_x, R_y, R_z, L_x, L_y, L_z, R'_x, R'_y, R'_z, L'_x, L'_y, L'_z]$$

4.3.1.4 Discussion of Feature Analysis on FDO_PT

These feature vectors are tested over the proposed algorithm in order to find the efficient feature vector. Table 4.1 shows 10-Fold cross validation results for isolated recognition of FDO_PT dataset. Grid and angular based feature vectors obtain similar results in the case of isolated gesture recognition. But, it is observed that the score S of classes in case of angular feature set are so close to each other, compared to grid-based feature vectors. That could lead to misclassification in on-line continuous, unsegmented recognition, in the case of the angular feature vectors. In addition, representation of some gestures such as *Ahead, Lashing* would not be meaningful because the arm is bent. Also, since the FDO_PT dataset is collected from only the single user, in the case of multiple users, the performance of angle and raw feature vectors will decrease dramatically. Therefore, grid-based feature vector is preferred as the main feature vector for the rest of the study:

$$F_{FDO\_PT} = F_{Grid} = [R_x, R_y, R_z, L_x, L_y, L_z, R'_x, R'_y, R'_z, L'_x, L'_y, L'_z]$$

where $[R_x, R_y, R_z, L_x, L_y, L_z]$ corresponds the spatial grid features for the Right and Left hand and $[R'_x, R'_y, R'_z, L'_x, L'_y, L'_z]$ accommodates the fuzzy gradient temporal feature of the grid feature.

	F_{Raw}	$F_{Angular}$	F_{Grid}
Mean (μ)	4.21	0.15	0.09
Std (σ)	0.95	0.15	0.14

Table 4.1: Results of the isolated FDO_PT dataset over different feature vectors by the used recognition algorithm (RM). The grid-based feature set obtains a better performance. Therefore, the grid-based feature vector is preferred as the main feature vector for the rest of the study.

4.3.2 Feature Analysis For FDO_CV

The outcomes of the FDO_PT dataset feature vector analysis played a critical role for determining the feature vector for the FDO_CV dataset. The conclusion of the FDO_PT analysis, that grid-based feature vector is the best feature vector, is applied to the FDO_CV dataset. Therefore, as was explained in the data collection of the FDO_CV dataset, data collection environment is setup to collect raw data in a form suitable for grid features. This is achieved by assuming users are in 2D planes (images) and they are stationed at the centre of images and touching the top, left and right

boundaries of images when the user vertically and horizontally stretches his arms (figure 4.8). Therefore, smoothed raw data is used directly as the grid-based spatial feature vector. The feature set of the FDO_CV dataset is:

$$F_{FDO\_CV} = F_{Grid} = [R_x, R_y, L_x, L_y, R'_x, R'_y, L'_x, L'_y]$$

where $[R_x, R_y, L_x, L_y,]$ corresponds to spatial grid features for the *Right* and *Left* hand and $[R'_x, R'_y, L'_x, L'_y]$ accommodates the fuzzy gradient temporal feature of the grid feature.

4.4 Channel and Class Model Construction

Having decided upon the appropriate feature, in order to represent the training cycles in a compact and parametric form, the next step is to construct channels (class models) from training cycles which are in the form of features vector F. Five-tuples *(C, L,H, F, B)*, notation and definition introduced in the problem definition chapter, are used as base for channel construction. Recall that the channel $H_{i,j}$ of class model C_i accompanies the time series or stream of the feature f_j.

Channel construction is based on estimating the parameters which represent best the underlying statistical distribution of training data at each time point. Hence, the underlying statistical distribution at each time index of the training samples determines the construction procedure. Note that it is assumed that the statistical distribution in a channel is the same for all its time indices.

In this study it is assumed that features are independent of each other and training data at a time index in a channel that follows normal statistical distribution. Therefore, the channel construction procedure is based on estimating the parameters of statistical mean μ and standard deviation σ to represent the training data at each time index in the channels. Bear in mind that, since construction is based on statistical methods, a sufficient amount of samples (training cycles) for each temporal class has to be collected in order to carry out a robust statistical parameter estimation.

As the last two graphics in figure 4.15 illustrates, a spatial channel includes a mean trajectory (μ) and standard deviation (σ) which accompanies the boundary of probable areas with some confidence level. The mean (μ) and standard deviation ($\mu \mp 3 * \sigma$) trajectories are the statistical mean and standard deviation at each time step, and index, respectively. The milestones of construction of the channel $H_{i,j}$ from the training cycles represented in the form of feature f_j can be described as follows:

- Feature Vector Analysing (f_i): Deciding upon comprehensive and distinctive feature vector F to represent spatial and temporal characteristic of the training cycles. In case of the spatial feature, in some domains, the raw data may be kept without any change, while in others, it may be transformed into other forms. For example, as discussed above in the case of the FDO_PT and FDO_CV datasets, the F_{Grid} grid-based spatial and temporal features are used.

- Pre-Processing and creating feature-based training samples: Raw training cycles are pre-processed (smoothed, transformed) before they are converted into feature space.

- Period Estimation (L_i): The length of the training cycles (periods) can vary because of noise and inter/intra user variations. Hence, a common period has to be computed (L_i) for the channels $H_{i,j}$. The common period is the average length of all training cycles. Recall that the period L_i is identical for all channels in class model C_i. For example in the case of the FDO_PT dataset, the minimum period is 20 (*Affirmative*) and maximum is 39 (*Wave Off*).

- Alignment: Because of the temporal variance, the length of training cycles can be less or greater than common periods. In addition, a specific event in a channel can be performed in the different indices. Therefore, the sample training cycles are stretched or compressed to common period length (L_i). Apart from that, some sub-event alignment operations are done to organize the same sub events to occur at the same indices. The main idea behind alignment, and the stretching and compression operation is to keep the statistical representation meaningful at the indices which have common characteristics (such as same sub-even). Alignment, and the stretching and compressing operation will be elaborated upon in the next section.

- Statistical Representation: Having aligned all the training cycles, and statistical parameters representing the underlying distribution at time indices can be estimated by using conventional statistical procedure. In this study, it is generally

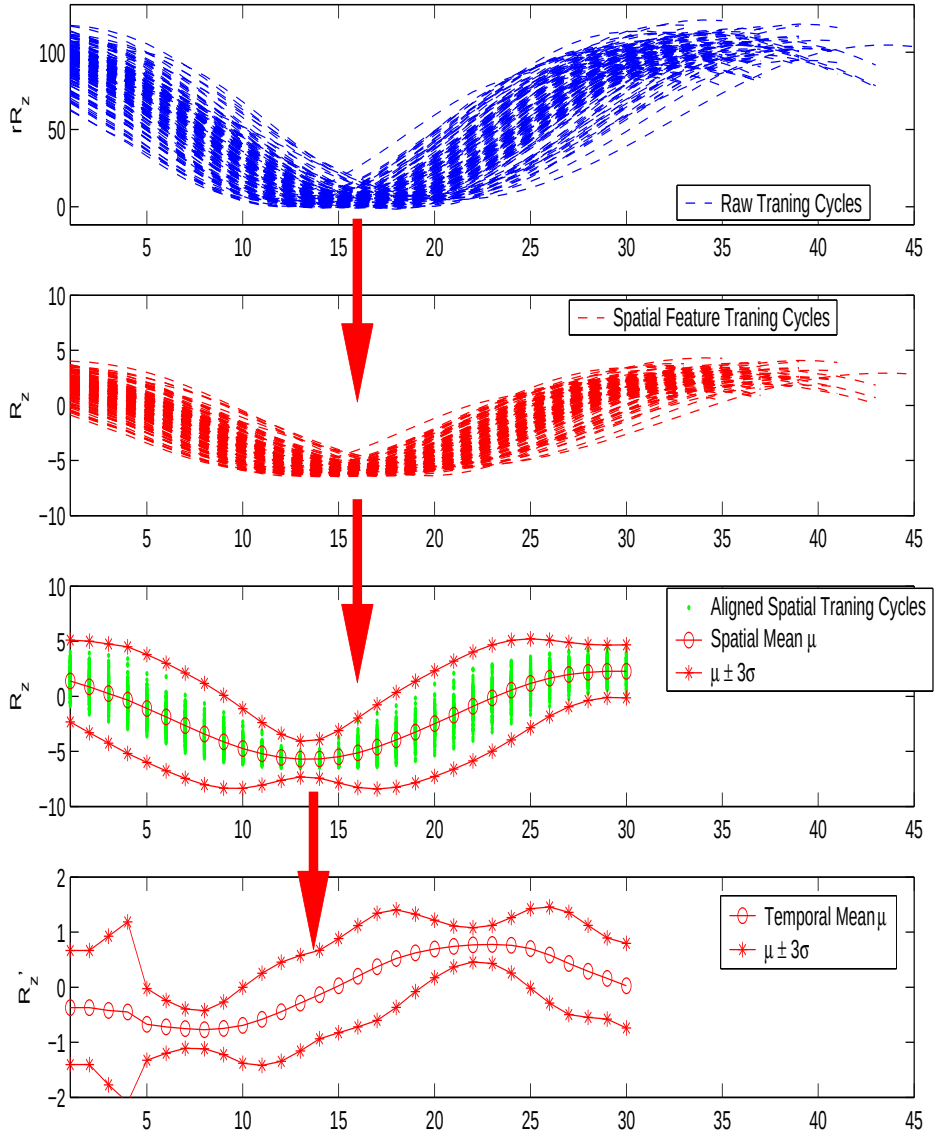


Figure 4.15: Constructing a spatial grid (R_z) and temporal fuzzy gradient (R''_z) channel for *down* gesture of FDO_PT dataset. First, raw pre-processed training cycles (top figure) are converted into spatial grid feature (second top figure, $rR_z \rightarrow R_z$). Then, a common period (L_{down}) is estimated for the class due to high variance among the samples length. Later, training samples are stretched, compressed and aligned to have the length of L_{Down} (top third figure). Temporal fuzzy gradient cycles are estimated from aligned spatial grid cycles. Finally, statistical Gaussian mean and standard deviation of aligned spatial and temporal cycles are calculated for each index.

assumed that at each time indices, the training cycles are normally distributed. Therefore, as the parameter statistical mean and standard deviation of training cycles are estimated. Please note that the alignment operation is vital to preserve the underlying statistical distribution at each index.

Figures 4.15 illustrate spatial (R_z) and temporal (R'_z) channel construction procedure for the *down* gesture of the FDO_PT dataset from raw training data (rRz). Firstly, the raw training cycles (top graph) are transformed into the spatial grid feature (second top graph, $rR_z \rightarrow R_z$). As the figure shows, the training cycles have different lengths. Therefore a common period (L_{down}) is estimated for the class. Later, training cycles are stretched, compressed and aligned to give the same length to L_{Down} (third graph). Temporal fuzzy gradient cycles are estimated from the aligned spatial grid cycles. Finally, the statistical Gaussian mean and standard deviation of aligned spatial and temporal cycles are calculated for each index. Due to the high number of channels and classes, the illustration of classes' models of FDO_PT and FDO_CV are omitted. But in the following sections, a detailed analysis of class models will be presented.

4.4.1 Alignment, Stretching and Compressing Operation*

Training cycles, due to intra/inter personal variance, noise and temporal variances, accommodate high variance in their lengths and location of sub-events. Therefore, before statistical parameters are estimated, training cycles have to be organized in a way to reduce the affect of temporal variances, or in another words to maximize common properties of training data at each time index. Therefore, stretching, compressing and sub-event alignment is applied over channels before statistical parameter estimation. These operations are vital for obtaining meaningful and robust statistical parameter estimation. For example, if the alignment operation is omitted on peaky sub-events, in the case of high temporal variance, these sub-events would not occur on the same indices. This could lead to unreliable statistical parameter estimations for these indices. In the following subsection, this phenomenon will be discussed in detail.

4.4.2 Stretching and Compressing

As the top graph of figure 4.15 illustrates, training data accommodates different variances. Hence, after converting raw data to feature space, the first operation is to compress or stretch to a common length, period L_i. Compression or stretching operation are performed for a given channel. The stretching operation is applied by uniform

linear interpolation. In another words, new interpolated data points are uniformly inserted into channels. The value of the new data points are the average value of the adjusting data points. Similarly, the compression operation is done uniformly. Instead of interpolating new data, two data points are merged to one with their average value.

4.4.3 Sub-Event Alignment

Sub-events are short phenomena in channels which are distinctive compared to the rest of the channel. Sub-events are mostly observed on temporal channels due to the sudden change of speed, direction, and acceleration. But it can be observed in spatial channels as peak points, global or/and local maximum/minimum points. The feature set F_{Grid} for the FDO_PT dataset does not accommodate distinctive sub-events. Even though, in case of a new feature set consisting of sub-events, those phenomena would be critical. Therefore, an alignment operation will be covered on the artificial data (W_Test). Figure 4.16 top left shows four sub-events (K=4), at indices $I_{se} = \{25, 49, 51, 75\}$) in the channel ($A_\beta$) of the artificial dataset (W_Test).

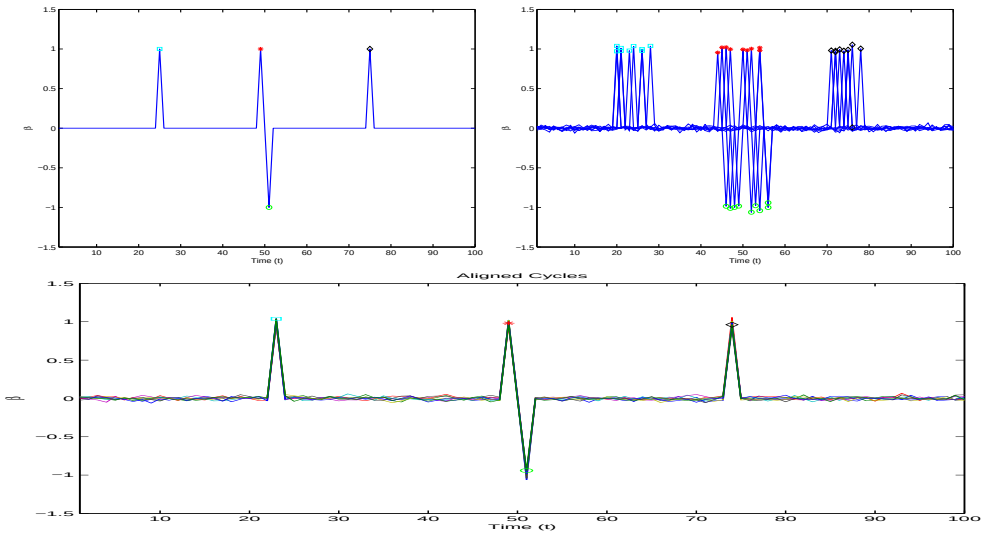


Figure 4.16: Sub-Events in β channel of class A in the parametrised artificial dataset W_Test (Top-Left). A_β channel has four sub-events at the ideal index ($I_{se} = \{25, 49, 51, 75\}$). Top-right figure shows the sample A_β channel in which sub-events are scattered. Bottom figure shows the aligned sub-events of scattered channels.

Since, the proposed recognition algorithm is based on an index prediction of given incremental data, sub-events could be extremely useful for the prediction process. Therefore, class models should preserve the sub-events properties, during model construction. Statistical representation could eliminate or reduce sub-events in the class

model, if sub-events are not aligned to a common index. For example, figure 4.16 (top and right) shows some samples of A_β channel. As can be seen, sub-events occur at different locations, indices I_{pse} in each sample. If the statistical parameters of these cycles are estimated without alignment, at ideal sub-event indices (I_{se}), robust and meaningful values are not obtained. Inadequate mean μ and large standard deviation *sigma* statistical parameters are obtained at ideal sub-event indices (I_{se}). But on the other hand, if sub-events in samples are aligned such as in figure 4.16 (bottom), a more robust parameter estimation can be achieved.

There is no universal sub-event detection algorithm, since sub-events have a wide range of uncommon properties from channel to channel and dataset to dataset. Therefore all the sub-events positions I_{pse} in channels are extracted with ad-hoc, supervised methods. Properties of sub-events, for example, the number of expected sub-events in a channel, their possible locations, their amplitude, spatial and temporal characteristics, are utilised in order to detect the possible index of sub-events at each cycle. But, once sub-events in cycles have been found, the following procedure is applied for all the channel and datasets. Since the number of sub-events (K) are specified manually for each channel, first, an ideal sub-events index I_{se_k} is estimated. The index of *k.th* ideal sub-event I_{se_k} is the average of estimated index of *k.th* sub-event I_{pse_k}.

$$I_{se_k} = \frac{1}{\#S} \sum_{k=1}^{K} I_{pse_k} \tag{4.5}$$

where #S is the number of samples. Having found the ideal sub-event I_{se_k}, sub-events are aligned to this index either partially compressing or stretching the cycles as explained in the previous section. But the important difference, here, is that these operations are applied only to some parts of cycles, and length of cycles are not changed.

Please note that the summary statistical knowledge of the sub-events' indices (I_{se_k} and I_{pse_k}) are stored, and utilised in later the proposed recognition algorithm.

4.5 Characterization of Dataset

Hitherto, class model construction has been covered. Complexity of the dataset or in other words, the inter similarity between class models, affects greatly the recognition algorithm. Therefore, in this section, an analysis on dataset complexities and class model similarities will be discussed. Note that throughout the thesis, the terms,

"complexity" and "similarity" of sequential data are used interchangeably. But this assumption is not realistic from the point of view of the pattern recognition, because the similarity between sequential data is more important than the complexity of pattern.

4.5.1 Basic Measures

The following properties of datasets are the basic measures used to evaluate the complexity of datasets at first sight. [79]

- Number of Features/Channels: Total number of channels used to construct class models in the dataset (ϑ).

- Number of Samples: Total number of training samples in a dataset. The number of observations is important to make a robust statistical representation. Insufficient number of samples can cause over or under fitting. As explained in the literature review chapter and the above feature selection section, *Curse of Dimensionality* and *Peaking Phenomenon* concepts shed a light on this issue. In literature, it is suggested that the number of samples for a class should be 10 times greater than the number of features (10ϑ) [52].

- Number of Classes (Vocabulary Size, Perplexity): Total number of the classes in dataset (ϖ). All classification tasks are based on estimating class boundaries (linear or non-linear) in feature space. Therefore, in a finite feature space, the huge number of classes increases the probability of overlapping or intersection among class' segments. The smaller the vocabulary, the more accurate and the faster the recognition. In addition, the complexity of grammar and length of vocabularies are inversely proportional to the complexity of class models and datasets.

These criteria are not comprehensive enoight for describing the complexity of a dataset. Therefore, in the following subsection more advanced techniques will be investigated. These techniques are entropy analysis; Chi-Square, skewness and kurtosis analysis for fitness of statistical parameter of class models; Fisher linear discriminant analysis; principal component-based *EROS* analysis; intersection volume analysis and temporal analysis for variance in samples length and position of sub-events. These techniques focus on different aspects of class models such as channel complexities, verifying assumed statistical properties, inter class similarities, similarity between samples and their associated and other classes' models, and principal feature analysis.

These techniques are exemplified over a parametrised artificial dataset W_Test. The reason behind preferring the W_Test dataset is due to its control parameters, which enable us to verify the complexity analysis with its parameters. In order to avoid repeating ourselves, the description of the W_Test is omitted here. A detailed description of the W_Test dataset is given in the Experiments and Results section 6.2.1. Refer to the results chapter for a detailed description of the W_Test before continuing to the next sections. After elaborating upon these techniques with the W_Test dataset, the complexity of the FDO_PT and FDO_CV datasets will be analysed.

4.5.2 Entropy

Entropy is an interchangeable term which actually originated in the area of physics. In physics, entropy corresponds to an irreversible degraded and unavailable-for-work energy, and therefore has a with different meaning, involving complexity, noise, disorder, uncertainty, probability, and random mixtures in various communities. In the context of machine learning, entropy is a technique along with statistical methods to measure information content and the disorder of a dataset.

In this text, entropy is used with the same meaning as in information theory, which corresponds to the uncertainty of the underlying stochastic process and information content of a system. In other words, the probability distribution of variables determines the uncertainty or randomness of the system. Therefore, the complexity of a system can be called as its uncertainty, or namely the entropy. For example, if the probability of a variable is concentrated around a point, the complexity (uncertainty or disorder) of that variable will be low. (Low entropy, figure 4.17-a). On the other hand, if the probability of the variable is uniformly distributed, the complexity (uncertainty), namely, the entropy of the system, will be high (Figure 4.17-b).

The probability distribution of systems can be described in a histogram such as the one in figure 4.17, which paves the way for computing entropy. Density of the bins in a histogram is inversely proportional to the entropy of the system. To be precise, the probabilities of a random observation are correlated to the density of the bin into where the observation falls [8]. Figure 4.17 illustrates the entropy of two different systems containing the same number of observations and bins.

Formally, the entropy of a discrete random variable $X = \{x_1, x_2, x_3 \cdots x_n\}$ with probability $P_X = \{p_1, p_2, p_3 \cdots p_n\}$ is defined as follows:

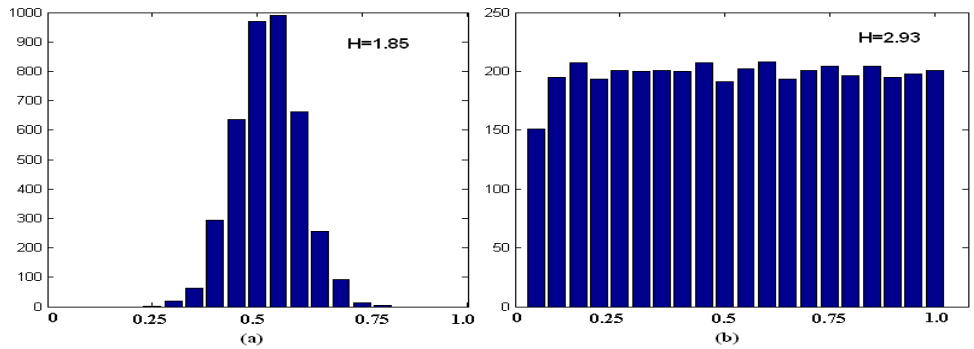


Figure 4.17: Histogram-based entropies. Histograms are the main tools for calculating the entropy. a) Histograms, in which data is concentrated around some the bins, have low entropy. b) The entropy would be higher if the data is distributed uniformly among bins. While, symmetric distributions, such as Gaussian, tend to have low entropies, uniform distributions, have high entropies.

$$H(X) \quad = \quad -\sum_{i}^{n} p_i \; log(p_i) \tag{4.6}$$

where the logarithm is generally base 2, for measuring the entropy in units of a bit. Note that, in this section, entropy is represented by H which should not be confused with the channel abbreviation H_i, j which always has the channel index j as sub-index. The entropy of a continuous distribution is analogous to a discrete distribution, in which the sum operator is replaced by an integral.

For calculating an entropy, a simple but reliable histogram-based approach is used [80]. The method is based on the classical histogram scheme, in which the density of the bins determines the probabilities. The method is sensitive to the width of the bins and the number of bins. Therefore, in our study, while the number of bins (k) is kept fixed, the width of the bins (W) is adjusted with regards to the number of bins and maximum/minimum values of features in the dataset ($W = (maxVal - minVal)/k$).

In a closed system, the entropy cannot exceed a maximum level (principle of maximum entropy). For a discrete distribution,the maximum entropy is reached, when all observations are distributed uniformly. In that case, the maximum entropy is $log(k)$ where k is the number of bins in the histogram. Whereas, in continuous distributions, the maximum entropy is related to the variance (σ) and has the value of $0.5\ log(2\pi\sigma e)$.

The calculation of the entropy of temporal data is analogous to non-temporal data. But in the case of temporal data, two kinds of entropy should be considered. The first

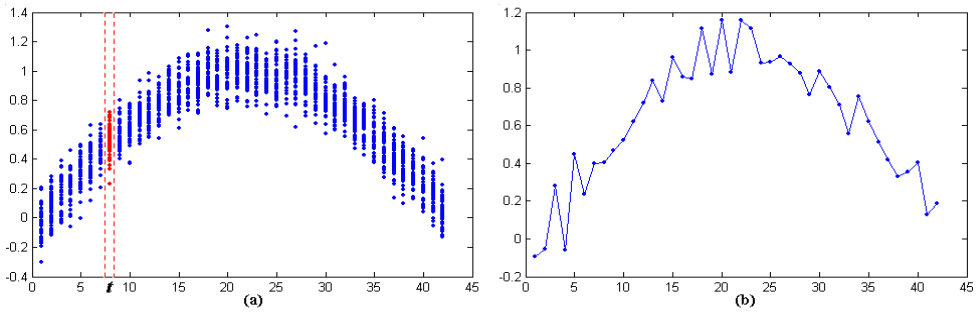


Figure 4.18: Data for frame (vertical) and channel (horizontal) entropy of temporal dataset. a) Frame (vertical) entropy corresponds to the spatial variance of whole samples in a frame (t) over all samples of the class. b) Channel(horizontal) entropy deals with inter spatial variance of a channel in a sample along time index.

one is horizontal or channel (hHX) entropy which corresponds to inter spatial variance (entropy) in a sample along the time axis. The second one is called vertical or frame entropy (hVX) and accommodates spatial entropy of whole samples at a certain time (t) or in a frame. Figure 4.18 depicts the sort of data used for horizontal and vertical entropy. Note that channel and frame entropy is used substitutely as horizontal and vertical entropy in the rest of the study.

In the pattern recognition literature, entropy is used in the following two ways: The first one is related to the content of information used for discrimination of classes, and the second one is complexity or uncertainty of a dataset. For the sake of classification, if a variable does not vary over time at all, its entropy would be zero, namely, it contains no or less information for the classification task [79]. Furthermore, if a variable contains so much uncertainty because of noise or irrelevant information, it would be a challenge. While we analyse the characteristics of data, we will focus on the following two criteria:

- Information content of features and classes
- Uncertainty or complexity of features and classes

Actually, both of these criteria can be summarized under the heading noise-to-signal ratio (NSR). Noise-to-signal ratio is the proportion of non-useful information (noise) to all the information. It is, in a way, similar to information gain. In addition, since a realistic dataset consists of multiple features and classes, it would be more appropriate to consider the average, maximum and minimum values of these criteria. In a compact manner, figure 4.24 depicts these calculations in a pseudo MATLAB source code.

4.5.3 Complexity of Features, Frames and Classes

Features underpin the recognition systems. Hence, complexity and uncertainty of features is vital. As has been mentioned, the horizontal and vertical entropies of features and frames have to be taken into account in temporal data analysis. While horizontal entropy is based on a channel, vertical entropy is based on frames.

The horizontal entropy of the s^{th} sample of the channel j of class C_i is estimated as follows (line 19, figure 4.24):

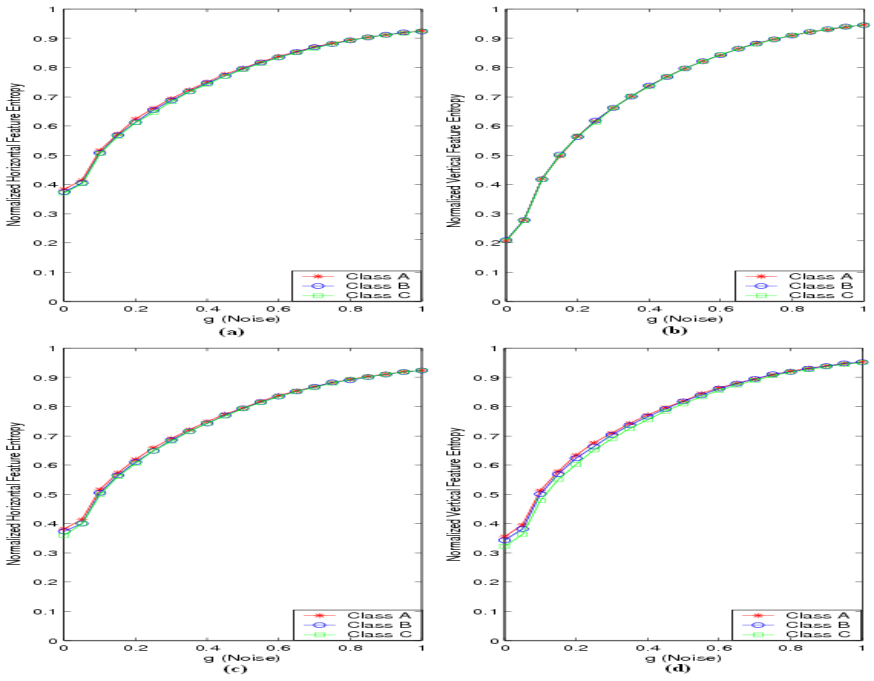


Figure 4.19: Normalized Average Vertical and Horizontal Feature Entropy of W_Test dataset with different gaussian noise (g) and c=h=d= {0, 0.2} and *irrel on*. Vertical and Horizontal entropies are proportional to the noise. Although high horizontal entropy is useful for the proposed algorithm, high vertical entropies degrade recognition.

$$hH(X_{i,j,s}) \quad = \quad \frac{1}{L_i}\sum_{t=1}^{L_i} p_{i,j,t,s}\ log(p_{i,j,t,s}) \quad 0 < j \leq \vartheta \ and \ 0 < s \leq \#S_i$$

where $\#S_i$ is the number of observations, L_i is the period and ϑ is the number of features for the class C_i. Consequently, the average horizontal feature entropy of class

C_i is computed as follows (lines 16-21 and 50 figure 4.24):

$$\bar{hH}(C_i) = \frac{1}{\vartheta * \#S * log(q)} \sum_{j=1}^{\vartheta} \sum_{s=1}^{\#S} hH(X_{i,j,s})$$

where q is the number of bins used in the histogram-based entropy estimation.

As pointed out before, since a histogram-based entropy has a maximum entropy limit, horizontal feature entropies can be normalized to interval [0,1] via maximum entropy. In this way, comparing various datasets is more comprehensible.

Horizontal feature entropy accommodates the internal spatial variance in a sample along the time axis. A high horizontal entropy indicates more scattered data points in time order. Thus, it provides more information for determining the time index of a data point, which is the main intuition behind the proposed algorithm.

Another important point to be addressed in temporal data is the entropies between samples. As mentioned earlier, vertical entropy (hVX) is used for analysing the uncertainty between samples at a certain time t. In a nutshell, vertical entropy accumulates spatial variance. Vertical entropy at the time t for the channel j of class C_i is estimated as follows (the lines 35-42, figure 4.24):

$$hV(X_{i,j,t}) = \frac{-1}{\#S_i} \sum_{s=1}^{\#S_i} p_{i,j,t,s}\ log(p_{i,j,t,s}) \quad 0 < j \leq \vartheta \ and\ 0 < t \leq L_i$$

Note that, $p_{i,j,t,s}$ for vertical and horizontal entropy estimation is not the same. While, for vertical entropy estimation, data points at time index t are considered, for horizontal entropy estimation, data in a sample is considered. Figure 4.18 illustrates the data used for vertical and horizontal entropy estimation.

Once again, for clarity and comprehensibility, vertical entropies for class C_i can be normalized as follows:

$$\bar{hV}(C_i) = \frac{-1}{\vartheta L_i log(q)} \sum_{j}^{\vartheta} \sum_{t}^{L_i} hV(X_{i,j,t})$$

where q is the number of bins used in the histogram.

High vertical entropy indicates that the dataset is scattered in feature space. In this case, the dataset is represented by a wider band width, which augments the intersection between classes. Consequently, it degrades the recognition results.

Figure 4.19 illustrates the average horizontal and vertical feature entropies of the artificial dataset W_Test with different Gaussian noise (in the a and b, c=h=d=0; in the c and d, c=h=d= 0.2 and *irrel on*). As the figures depicts, horizontal and vertical entropies are directly related to the noise level. Although, noise provides high horizontal entropy which is important for the next index prediction in the proposed algorithm, it also usually paves the way for high vertical entropy, which dramatically degrades the recognition.

So far, complexity of features and frames have been discussed. In addition, the complexities of classes as a whole can be estimated based on either prior class frequency probability or averaged channel (horizontal) entropy as explained above. Total class entropy $H(C)$ based on prior frequency probability can be estimated as follows:

$$H(C) \;=\; -\sum_{i}^{\varpi} \pi_i \; log(\pi_i)$$

where ϖ is the number of classes and π_i is the prior probability of the class i. This scheme is useful when prior probability information is available. In fact, in terms of the recognition task, that complexity measurement does not accommodate so much information. Therefore, the average channel entropy is used for class entropy estimation as follows (lines 1-7, figure 4.24):

$$H(C_i) \;=\; \frac{-1}{\vartheta L_i} \sum_{j}^{\vartheta} \sum_{t}^{L_i} p_{i,j,t} \; log(p_{i,j,t}) \tag{4.7}$$

In our study, templates (channels, $H_{i,j}$) in class models C_i are considered for class entropy estimation ($H(C_i)$, Equation 4.7). The ideal templates are assumed to be the mean or most characteristic trajectory that a class will follow in feature space. For example, in the artificial dataset W_Test, ideal templates are those with all control parameters set to zero (g=c=d=h=0, *irrel off*). But, in the case of a real dataset 4.4, as explained in the model construction section, templates are constructed from several samples.

Figure 4.20 depicts the average horizontal class (template based) the entropies of dataset W_Test under various Gaussian noise. Because of Gaussian noise and not applying any alignment scheme during the construction of templates, the mean template does not accommodate the effect of noise over the entropy in the figure. Therefore, in this example, template-based horizontal class entropy tends towards the stationary

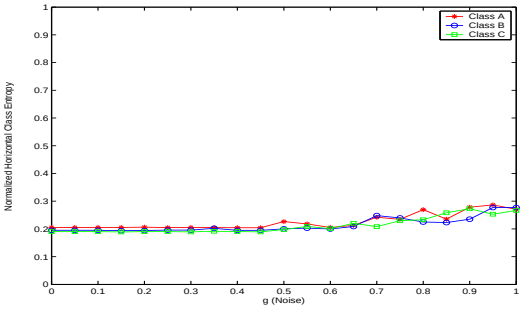


Figure 4.20: Normalized Average Horizontal Class Entropy (template-based) of W_Test dataset with different noise (g) and c=h=d=0 and *irrel on*. Since, during template construction, no alignment is used and Gaussian noise, the entropy tends to be stationary, unlike horizontal and vertical feature entropy.

entropy, unlike feature entropies.

The calculation of entropies based on ideal templates paves the way for analysing the common information (mutual entropy) between samples and classes.

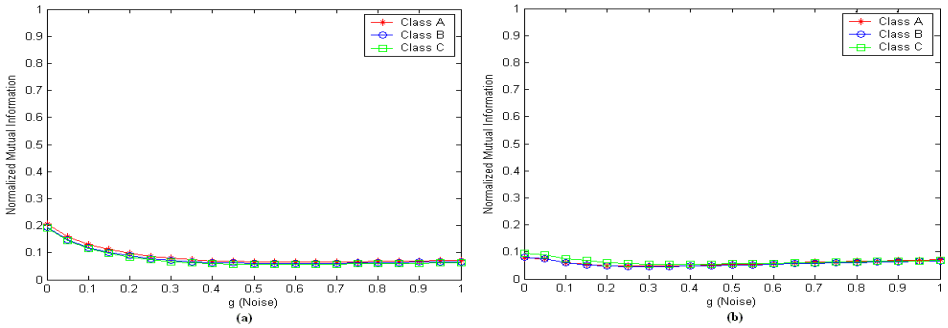


Figure 4.21: Normalized Average Horizontal Mutual Information of W_Test dataset with different noise (g) and c=h=d=0 and *irrel* is on.

Cross Mutual Entropy

Mutual information between two variables I(X,Y) indicates how much information one random variable (X) tells about another variable (Y). In other words, it is the shared information between the two variables. It also expresses dependency between random variables. For example, if two variables are independent, the mutual information is zero, because random variables do not accommodate any information about each other. Mutual information is estimated as follows in this study:

$$
\begin{aligned}
I(X,Y) &= \sum_{x}\sum_{y} p_{xy} log(\frac{p_{xy}}{p_x p_y}) \\
I(X,Y) &= H(X) + H(Y) - H(X,Y); \\
H(X,Y) &= -\sum_{xy} p_{xy}\ log(p_{xy}) \\
\bar{I}(X,Y) &= \frac{1}{\vartheta}\sum_{i}^{\vartheta} I(X_i, Y_i)
\end{aligned}
$$

where p_{xy} is joint probability, p_x and p_y are the marginal probabilities of random variable X and Y; H(X,Y) is the joint probability entropy which corresponds to the total entropy of the combined variables. Minimum mutual entropy is zero and maximum mutual information is $min(H(X), H(Y))$. $\bar{I}(X,Y)$ is the average mutual information between two multi-variate classes.

In our study, mutual information between the samples and their associated class models are investigated (I(X,C)). Figure 4.21 illustrates the mutual information in the dataset W_Test. Even though, horizontal feature entropies increase proportionally with noise (figure 4.19-a,c), since, template-based horizontal class entropies tend towards stationary behaviour (figure 4.20), the mutual information between feature ($h\bar{H}X$) and class ($\bar{HC}$) also approaches stationary behaviour.

Besides that, in our study, *cross* mutual information between samples (hHX) and all classes model (C) is analysed to find out the cross shared information. Lines 22-30 in figure 4.24 depict how cross mutual information between ideal class templates ($H_{i,j}$) and samples (hHX) are calculated.

It is expected that cross mutual information between a sample X_i and its associated class model (C_i), will be higher than those to which X_i does not belong. Therefore, a scheme can be applied upon all cross mutual information to utilise an average disparity measurement in the dataset. The scheme is called *mutual information precision* (MIP) in the study and implemented as lines 31-32 in figure 4.24 illustrates.

MIP is the ratio of the number of samples, of which mutual information with their associated classes are the greatest compared to other templates, and to the total number of samples. In fact, this is not more than a nearest neighbourhood implementation in which, in a way, precision boosts when the entropy of a sample is the nearest to the entropy of its associated template.

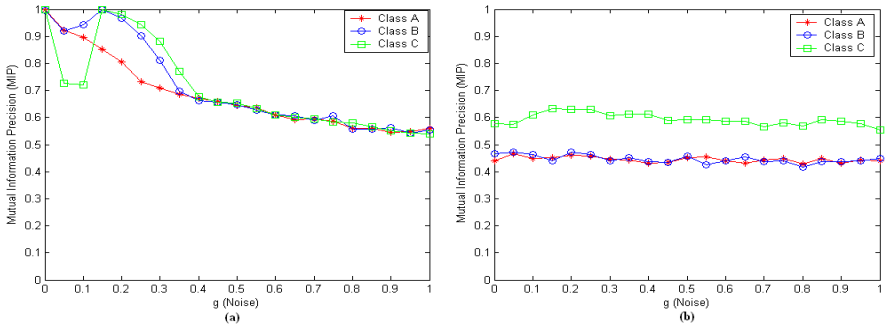


Figure 4.22: Mutual Information Precision (MIP) of W_Test dataset with different noise (g) and c=h=d={0 (a), 0.2 (b)} and *irrel* is on.

$$MIP = \frac{\#nearestI}{\#S}$$

$$\#nearestI = \sum_{i}^{\varpi}\sum_{s=1}^{\#S_i}\delta(\underset{a}{argmax}(I(hHX_{i,s}, H_{a,j})), i) \quad 0 < j \leq \vartheta; 0 < a \leq \varpi$$

where $\delta(x, y)$ is the Kronecker delta function which returns 1 if x equals y and 0 otherwise. MIP ranges in the interval of $[0, 1]$. In the worst case, MIP will be zero which will be an indicator of either misrepresentation of templates or noisy samples. On the other hand, the higher the MIS, the greater the similarity between class models and their samples. Figure 4.22 illustrates the MIP for the W_Test with Gaussian noise and various parameters.

Noisiness of Channels (Noise-Signal-Ratio)

Besides the mutual information precision (MIP), noise ratio would be a good factor to characterize datasets. If we assume noise as the non useful information in a system, we can estimate the noise ratio of a class by taking the proportion of non useful information (noise) to all information [79]. If the average feature entropy $\bar{H}(X)$ and $\bar{H}(X) - I(X, C)$ is taken as all information and non useful data respectively, the noise signal ratio (NSR) will be :

$$NSR = \frac{\bar{H}(X) - \bar{I}(X, C)}{\bar{H}(X)}$$

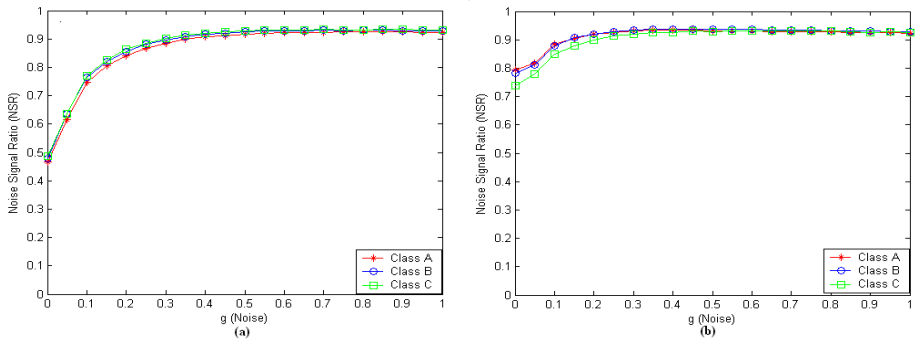


Figure 4.23: Noise Signal Ratio (NSR) of W_Test dataset with different noise (g) and c=h=d={0 (a), 0.2 (b) }.

where entropies are horizontal. Noise Signal Ratio implies that the dataset contains irrelevant information or natural random variation. In other words, it corresponds to the average dissimilarity between samples and their associated templates in a class. Actually, $1 - NSR$ corresponds to *information gain* . A higher noise signal ratio signifies more irrelevant information and vice versa. NSR ranges in the interval of [0,1]. While zero indicates a complete correlation between signals and their associated templates 1 indicates independence or dissimilarity in the set.

Figure 4.23 shows the NSR of the dataset W_Test with different Gaussian noise and parameters. Since mutual information (figure 4.21) is constant while the horizontal entropy of features increases, consequently, the NSR will consequently increase.

4.5.4 Summary of Entropy

Entropy is a general scheme for analysing information content and complexity of a dataset. Analyses are class-based, in other words, operation is carried out on either individual classes (HC), or samples ($h\bar{H}X$, $h\bar{V}X$), or samples and their associated classes (MIP,NSR). So far, the theory and its application on temporal data (W_Test dataset) have been explained in detail. For the sake of clarity, it would be appropriate, in a compact way, to conceptualise the entropy measurements for characterization of a dataset. Once again, for that purpose, the dataset W_Test will be exemplified.

- Horizontal (Channel) Feature Entropy ($h\bar{H}X$): Corresponds to average entropy in a channel along the time axis. Normalized horizontal entropy, ranges from 0 to 1. High horizontal entropy indicates scattering of data along the time axis and is useful for predicting the time order of frames incrementally in the proposed algorithm. Lines 16-21 in figure 4.24 correspond to the estimation of horizontal

```
% Class Entropies(hC)
for i=1:NC                                  % NC Number of Class
    for k=1:NF                              % NF Number of Feature
        x=CLASS_TEMPLATE{i}(k,:);
        hC(i,k)=entropy(x);                 % Class Entropies(hC)
    end
end
%Horizontal, Vertical Feature (hHX, hVX )
%and Cross Mutual Entropies (hI=H(hC,hHX))
for i=1:NC
    per=size(CLASS_TEMPLATE{i},2);          % Period of Class i
    for j=1:numOfsamples(i);
        data=load(sampleFile{i,j});         % Loading jth sample of Class i
        sData=makeSameLen(data,per);        % Make data length same as template
        %Total Horizontal Feature Entropy (hVX )
        for k=1:NF
            allVData{k}(j,:)=sData(:,k);    % Rearrange data for Vertical Entropy
            x=data(:,k);
            hHX(k)=entropy(x);              % Horizontal Entropy
            totHHX(i,k)=totHHX(i,k)+hHX(k); % Total Horizontal Entropy
        end
        % Total Mutual Cross Information between samples and class templates
        for ii=1:NC
            for k=1:NF
                y=data(:,k);x=CLASS_TEMPLATE{i}(k,:)
                e=entropyJoint(x,y);        % joint entropy
                mI(ii,k)=hC(ii,k)+hHX(k)-e; % Mutual Entropy, information
                totHI{i}(ii,k)=totHI{i}(ii,k)+mI(ii,k);
            end
        end
        nearest=findNearestHC(mI,hC,1);     % Find nearest hC to mI
        MIP(i)=MIP(i)+sum(i==nearest)/NF;   % Mutual Information Precision
    end
    %Vertical Feature Entropy (hVX )
    for k=1:NF
        allVX=allVData{k};
        for iii=1:per
            x=allVX(:,iii);
            e(iii)=entropyMy(x);
        end
        hVX(i,k)=mean(e);                   % Vertical Feature Entropy (hVX )
    end
    totHI{i}=totHI{i}./numOfsamples(i);     % Normalize Total Mutual Informat4ion
    HI(i,:)=totHI{i}(i,1:NF);               % Mutaul information of current class
    avgHI(i)=mean(HI(i,:),2);               % Average mutual information of features
end
% Normalizing
MIP=MIP./numOfsamples;
totHX=totHX./numOfsamples;
avgHHX=mean(totHX,2);
avgHVX=mean(hVX,2);
avgNoise2SignalRatio=mean(((avgHHX-avgHI)./avgHHX),2)
```

Figure 4.24: Calculation Horizontal, Vertical, Cross Mutual Entropy and Noise Signal Ratio (information gain) in a pseudo MATLAB source code.

feature entropy. Horizontal channel entropies of static classes are approximately 0, therefore they do not convey any information for frame index prediction in the proposed algorithm.

Table 4.2 shows normalized horizontal entropy of the W_Test dataset when parameters equal 0 and 0.2. As expected, the horizontal entropy increases proportionally with parameters and table 4.2 reflects this property. For example, there is a remarkable similarity between the channels A_β, B_β and C_γ apart from a couple of peaks. Slightly different entropy values between these channels reflect this similarity in the channels. Apart from this, the table shows relatively higher entropy for the more scattered α channels when the parameter is set to g=c=d=h=0 and *irrel on*.

	α	β	γ	α	β	γ
Class A	0.55	0.06	0.54	0.66	0.53	0.67
Class B	0.55	0.03	0.54	0.66	0.50	0.67
Class C	0.55	0.53	0.1	0.66	0.67	0.49
	g=d=c=h=0			g=d=c=h=0.2		

Table 4.2: Normalized Horizontal Entropy of W_Test dataset with parameter *irrel on* and g, c, h,d = {0, 0.2}.

- Vertical (Frame) Entropy ($h\bar{V}X$): Accommodates spatial variance in a frame among samples for a class. It is normalized to interval of [0,1]. High vertical entropy means wider band width which can degrade recognition rate. Lines 35-42 in figure 4.24 illustrate the estimation of vertical entropy of classes. Minimum, average and maximum estimates will be shown for characterizations for more complicated datasets.

  Vertical feature entropies (hVX) of the dataset W_Test with parameters 0 and 0.2 are demonstrated in table 4.3. Vertical feature entropies (hVX) of channels, which are independent the parameter *irrel on*, are zero when all parameters are set to zero. Furthermore, the table shows that, spatial complexity of the dataset increases proportionally when the parameters are increased.

- Horizontal Class Entropies (HC): Represents the horizontal entropy in the model of classes (C). It is used for estimation of mutual information between the signal and the templates, MIP and NSR. All the criteria for horizontal channel entropies are valid upon the class entropies.

  Table 4.4 shows the horizontal class entropies. Bear in mind that the ideal case of templates (all parameters are 0 and *irrel off*) are used for the estimations. For

	α	β	γ	α	β	γ
Class A	0	0	0.66	0.63	0.54	0.73
Class B	0	0	0.67	0.63	0.51	0.72
Class C	0	0.66	0	0.58	0.73	0.50
	g=d=c=h=0			g=d=c=h=0.2		

Table 4.3: Normalized Vertical Feature Entropy of W_Test dataset with parameter *irrel on* and g, c, h,d = {0, 0.2}.

example, since the templates of A_γ, B_γ and C_β are zero in the ideal case, their entropies are zero, as the table illustrates.

	α	β	γ
Class A	0.55	0.06	0
Class B	0.55	0.03	0
Class C	0.55	0	0.2

Table 4.4: Normalized Horizontal Class Entropy of the W_Test dataset.

- Cross Mutual Matrix (CMM): Accommodates minimum, average and maximum shared information between samples and classes. It is based on horizontal entropies and normalized to interval of [0,1]. The higher mutual information points out more shared information between samples and classes. It is expected that mutual information between a class and its samples are high and vice versa.

  Tables 4.5 and 4.6 represent cross mutual information for parameters 0 and 0.2 respectively. The property showing that the channels A_α and B_β are similar, can be seen in the tables as well. Besides that, since HC of the channels A_γ, B_γ and C_β are zero, their cross mutual information is also zero. Table 4.6 illustrates possible shared information is lost between samples and their associated templates, when the parameters are increased. Note that the cross mutual entropy tables are column-based, namely each column, indicates cross mutual information between the given samples of the class in the row and the classes in the columns.

- Mutual Information Precision (MIP): Provides a brief, comprehensive and accurate description of a cross mutual table. It is the rate of the number of samples, of which mutual entropies are the nearest to their associated templates among others, to the total number of samples. MIP takes values from the interval of [0,1]. The highest MIP illustrates the most distinctive class.

- Noise Signal Ratio (NSR): Signifies the content of information in a class. It is the

	Class A			Class B			Class C		
	α	β	γ	α	β	γ	α	β	γ
Class A	0.55	0.06	0	0.55	0.02	0	0.36	0	0.02
Class B	0.55	0.03	0	0.55	0.03	0	0.36	0	0.01
Class C	0.36	0.01	0	0.36	0.01	0	0.55	0	0.02

Table 4.5: Normalized Cross Mutual Entropies of the W_Test dataset with parameter *irrel on* and g=c=h=d=0.

	Class A			Class B			Class C		
	α	β	γ	α	β	γ	α	β	γ
Class A	0.14	0	0.01	0.14	0	0.01	0.14	0.01	0
Class B	0.14	0	0.01	0.14	0	0.01	0.14	0.01	0
Class C	0.15	0.02	0	0.15	0.02	0	0.17	0.00	0.01

Table 4.6: Normalized Cross Mutual Entropies of W_Test dataset with parameter *irrel* is on and g=c=h=d=0.2.

ratio of non useful information (noise) to useful information. Here, information is described in terms of average mutual information of a class and its samples. It is not normalized and the higher values imply an irrelevance level of data among samples.

These criteria, indicated only by the mean$\mp\sigma$ (standard deviation), minimum and maximum of values are shown in tables 4.7 and 4.9. Since the dataset W_Test is not large enough to have a meaningful statistical standard deviation representation, the standard deviation (σ) is omitted from the tables.

	$\bar{hC}$	$\bar{hHX}$	$\bar{hVX}$	$I(hHX, hC)$	$\bar{MIP}$	$\bar{NSR}$
Mean	0.20	0.38	0.21	0.20	0.99	0.48
Min	0.19	0.37	0.21	0.19	0.99	0.46
Max	0.21	0.38	0.21	0.20	1	0.48

Table 4.7: Characteristics of the dataset W_Test with parameters *irrel on* and g=c=h=d=0.

4.5.5 Chi-Squared Test (χ^2)

It is assumed throughout our study that the underlying distribution of the samples at a given time is a Gaussian distribution. The estimations of probabilities or membership degrees are pivoted on this assumption. Therefore, we have to see whether the underlying distributions are normal or not. For this purpose, Chi-Squared, Goodness of Fit Test, (χ^2) can be used.

	$\bar{hC}$	$\bar{hHX}$	$\bar{hVX}$	$I(hHX, hC)$	$\bar{MIP}$	$\bar{NSR}$
Mean	0.25	0.78	0.52	0.12	0.34	0.84
Std	0.31	0.16	0.18	0.01	0.25	0.01
Min	0.00	0.46	0.37	0.00	0.19	0.84
Max	0.80	0.89	0.83	0.55	0.63	0.85

Table 4.8: Characteristics of the dataset W_Test1 with parameters *irrel on* and g=0.1, h=d=0.2,c=0.1.

	$\bar{hC}$	$\bar{hHX}$	$\bar{hVX}$	$I(hHX, hC)$	$\bar{MIP}$	$\bar{NSR}$
Mean	0.24	0.70	0.49	0.10	0.34	0.86
Std	0.24	0.07	0.25	0.00	0.36	0.00
Min	0.00	0.58	0.26	0.00	0.05	0.86
Max	0.67	0.81	0.82	0.45	0.74	0.86

Table 4.9: Characteristics of the dataset W_Test2 with parameters *irrel on* and g=h=d=0.2, c=0.1

Chi-Squared test is a generic statistical technique used to validate the fitness of assumed underlying distribution in terms of representing the population. Suppose that, the samples are observed either from an assumed distribution (D_0) or an unknown distribution (D_1). The interest of the problem is to find a confidence level of the initial hypothesis which implies that the distribution (D_0) generates the samples. While the initial hypothesis is called *null hypothesis* (H_0), the second one, which indicates an unknown distribution, is called *alternative hypothesis* (H_1). Based on the cumulative sum of distances between observed samples and expected values, the *null hypothesis* is either rejected or accepted with some degree of confidence. The Chi-Squared test can be estimated as follows:

$$\chi^2_{obs} = \sum_{i}^{k} \frac{(E_i - O_i)^2}{E_i} \approx \chi^2_{k-p-1} \quad if\ H_0\ is\ true.$$

where k is the number of segments dividing the samples range; E_i (calculated assuming H_0 true) and O_i are the expected and observed number in the segment i respectively. Bear in mind that each segment has to consist of more than 4 observations due to a Chi-Squared constraint. The larger the value of χ^2_{obs}, the more likely it is that the null hypothesis is false or underlying distribution is not fitted. The fitness can be assessed with some degree of confidence (expressed in probability) via Chi-squared distribution.

It can be shown that the test statistic has a Chi-Squared distribution (χ^2_{DoF}) which is continuous and defined for the positive values. Chi-Squared distribution is only

dependent on one parameter called the degree of freedom (DoF). DoF is merely the number of parameters that can vary independently. For instance, in the case of testing a normal distribution, $DoF = k-p-1 = k-2-1$, where p is the number of parameters (mean, (μ) and standard deviation (σ)).

Once, χ^2_{obs} is estimated, it can be assessed in the following way: The χ^2_{obs} is compared to χ^2_{DoF}, against the corresponding value of DoF in the critical-values table of Chi-Squared with some confidence levels. These values of 0.05 or 0.01 are the frequently used confidence levels. If χ^2 is less than the critical value, the null hypothesis is accepted by the chosen confidence level.

Since an artificial dataset is generally created by adding noise that is normally distributed, the Chi-Squared test is unnecessary to control whether the underlying distribution is normal or not. Therefore, the Chi-Squared test should be considered primarily for real datasets. Actually, the Chi-Squared test has the capability of measuring the fitness of any system or association of two or more variables, given the observations and expectation.

4.5.6 Skewness and Kurtosis

Skewness is used to measure the asymmetry of the population around the sample mean. Negative and positive values of skewness indicate left or right directed scattering from the sample mean, respectively (Figure 4.25-c,e). Skewness of a symmetric distribution, such as normal distribution, around the sample mean is zero.

Skewness is the ratio of the third moment of mean data to the third power of the standard deviation of data. Skewness of a population is as follows:

$$\gamma = \frac{E(X-\mu)^3}{\sigma^3} \tag{4.8}$$

Kurtosis is another measurement of univariate distribution. Kurtosis is related to flatness of the distribution, or in other words, size of its tails. If a distribution has a unique peak around the mean and a small flat area around the tails, it is called leptokurtic (high) kurtosis (Figure 4.25-b). Distributions with high kurtosis (leptokurtic) are similar to normal distribution. On the other hand, if a distribution, such as in figure 4.25-a), does not have a distinct peak around the mean, it is called low kurtosis or platykurtic. Distribution with low kurtosis tends towards uniform distributions. Kurtosis is the ratio of the fourth moment to the fourth power of standard deviation and it is calculated as follows for a distribution with mean (μ) and standard deviation σ:

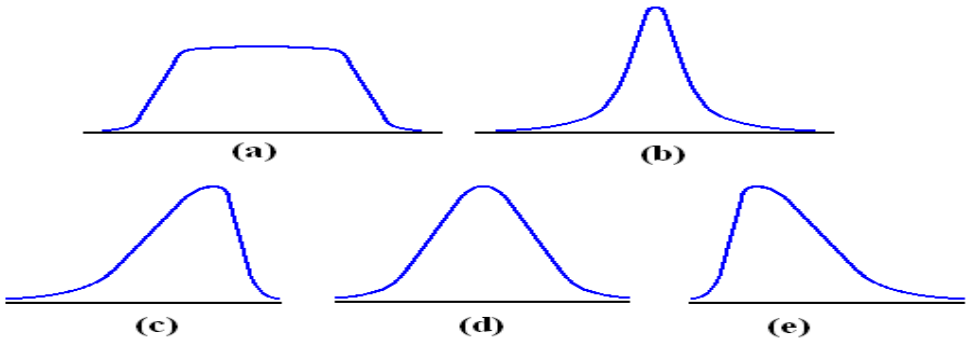


Figure 4.25: Kurtosis and Skewness of a univariate distribution. a) Platykurtic (low) Kurtosis b) Leptokurtic (high) Kurtosis c) Negative Skewness d) Symetric (Not skewed) e)Positive Skewness

$$k = \frac{E(X-\mu)^4}{\sigma^4} \tag{4.9}$$

Kurtosis of a normal distribution is 3. If kurtosis of a distribution is less than three, that would mean a flatter, more uniform distribution. Kurtosis and skewness of a dataset validate the fitness of the distribution selected for modelling the classes at each frame index.

Skewness and Kurtosis are proposed for univariate data. For matrix or multivariate data, the mean of absolute values of skewness and kurtosis of all variables (rows) is preferred. In our study, in the case of multivariate data, that scheme has been carried out.

4.5.7 Fisher Linear Discriminant

In pattern recognition literature, besides *Fisher linear discriminant* (FLD), several adequate techniques are proposed to discriminate classes linearly. Hence, prior to using more complicated non-linear techniques, it would be useful to analyse the underlying decision boundaries in terms of linear discriminant theory, which will also pave the way to a comprehensive analysis of the correlation between classes.

Roughly speaking, discriminant analysis investigates the disparities between two or more classes with respect to multiple variables in order to expose the discriminating variables and factors. Linear discriminant analysis (LDA) is a type of discriminant analysis (DA) which seeks linear decision boundaries between classes.

Linear discriminant analysis reduces high dimensional data into one dimensional

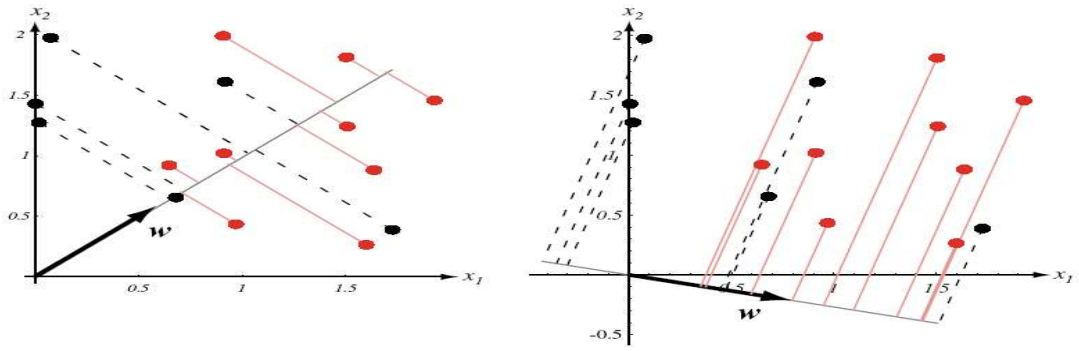


Figure 4.26: Discriminant analysis seeks an optimum vector w on which, while the scatter between projected points in the same class is minimized, the scatter between projected class' means are maximized (two different classes red and black). The vector w in the right figure is more discriminatory than in the left [27].

data, namely, a vector (line) w by projecting points onto the vector ($y = w^T x$). The vector is a unit length vector and its direction is important for discrimination. The core of Fisher linear discriminant analysis is to find an optimum vector w, on which, the projected points are clustered according to their class. Figure 4.26 illustrates the projected points clusters for two different vectors w.

In Fisher linear discriminant analysis, the optimum vector, w, maximizes the criteria function $J(w)$, which is the ratio of the *between class scatter*, S_B and *within class scatter* S_W as follows:

$$J(w) = \frac{w^{\intercal} S_B w}{w^{\intercal} S_W w} \tag{4.10}$$

where

$$\begin{aligned} m_i &= \frac{1}{n_i} \sum_{x \in D_i} x \\ S_i &= \sum_{x \in D_i} (x - m_i)(x - m_i)^{\intercal} \\ S_W &= S_1 + S_2 \\ S_B &= (m_1 - m_2)(m_1 - m_2)^{\intercal} \\ w &= S_W^{-1}(m_1 - m_2) \end{aligned}$$

D_i consists of the data for class i.

Although, Fisher linear discriminant analysis is used for two classes, it can be

extended for multiple classes - multiple discriminant analysis (MDA). For a further discussion, please refer to [27].

In our study, Fisher linear discriminant analysis is used to obtain the cross-scatter between classes. The value of the criteria function is used as a discrimination metric. Since J(w) cannot be normalized, for each dataset, it can have a different range. It has to be interpreted individually for each dataset.

In addition, the Fisher linear discriminant analysis has to be modified in the following way for temporal data:

$$J(w,t) = \frac{w(t)^{\mathtt{T}} S_B(t) w(t)}{w(t)^{\mathtt{T}} S_W(t) w(t)} \tag{4.11}$$

where

$$\begin{aligned}
m_i(t) &= \frac{1}{n_i} \sum_{x \in D_{i,t}} x \\
S_i(t) &= \sum_{x \in D_{i,t}} (x - m_i(t))(x - m_i(t))^{\mathtt{T}} \\
S_W(t) &= S_1(t) + S_2(t) \\
S_B(t) &= (m_1 - m_2)(m_1 - m_2)^{\mathtt{T}} \\
w(t) &= S_W^{-1}(t)(m_1(t) - m_2(t)) \\
0 < \quad t &< min(L_1, L_2) \;\; L \; is \; the \; period \; of \; classes.
\end{aligned}$$

where the data, $D_{i,t}$ corresponds to all the data t frame of samples in the class i. In other words, Fisher linear discriminant analysis is applied to the training data points which have same time indices. This leads to a frame-based temporal discriminant analysis in which the inter-class discriminancy of frames is addressed. This analysis is useful for proposed algorithm since the algorithm utilises the degree of scattering of frames to predict the fitness (membership degree) and time order of the test data. For instance, figure 4.27 illustrates the Fisher linear discriminant criteria function metrics for class A and class B, and C for the W_Test dataset. In figure 4.27-a, for example, Fisher linear discriminant reveals the key discriminant frames (50,51) between class A-B.

Figure 4.28 envisions the Fisher linear discriminant analysis over an artificial dataset W_Test with various noise levels g=c=d=h=0 and *irrel* on. The metrics are the mean, max and min of the criteria function J(w,t). Since the *irrel* parameter is based on normal distribution (0,σ) at each time index for a large number of data points, it does not cause any discrimination factors. As the figure indicates, for the high value of noise,

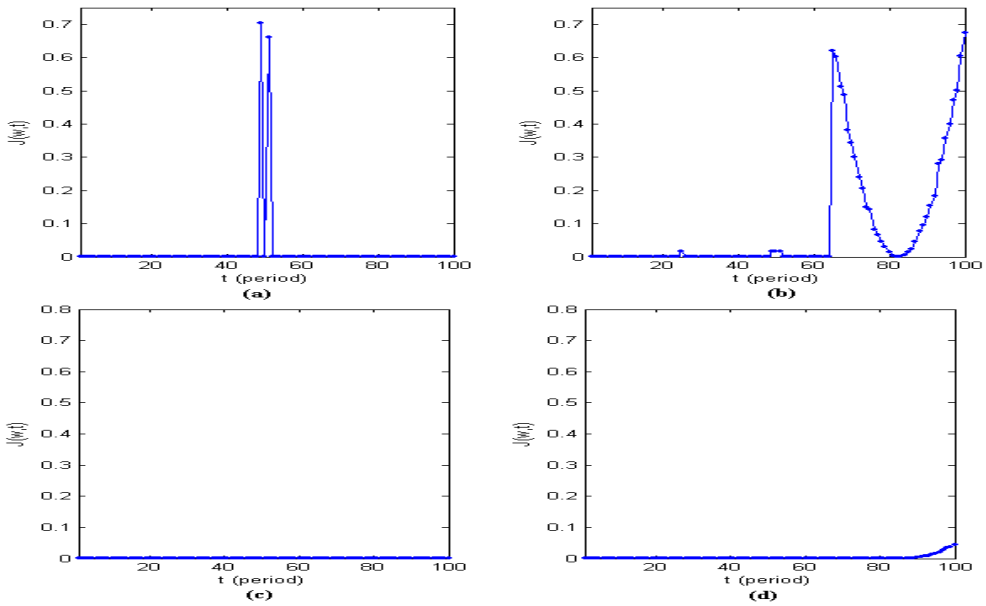


Figure 4.27: Linear Discriminant Analysis Criterion Function for the dataset W_Test Between Class A and B, C when *irrel* is on, g=0.2,c=h=d=0 (top figures) and g=d=h=0.2 and c=0.1

	A $\mu \mp \sigma; min/max$	B $\mu \mp \sigma; min/max$	C $\mu \mp \sigma; min/max$
A	$0.00 \mp 0.00; 0.00/0.00$	$0.01 \mp 0.10; 0.00/0.70$	$0.09 \mp 0.17; 0.00/0.67$
B	$0.01 \mp 0.10; 0.00/0.70$	$0.00 \mp 0.00; 0.00/0.00$	$0.09 \mp 0.17; 0.00/0.72$
C	$0.09 \mp 0.17; 0.00/0.67$	$0.09 \mp 0.17; 0.00/0.72$	$0.00 \mp 0.00; 0.00/0.00$

Table 4.10: Cross Table of J(w,t) for the dataset W_Test when g=c=d=h=0 and *irrel* is on.

Fisher linear discriminant analysis suggests that it is unlikely that there are linear decision boundaries. Furthermore, the situation gets worse when other parameters are set to non-zeros values. For instance, as can be seen from figures 4.27-c and d, scatter discrimination between classes A, B, and C almost disappears, when the parameters are set to g=c=d=h=0.2 and *irrel* is on.

In order to characterize a dataset in terms of the Fisher linear discriminant analysis, mean $\mp$ standard deviation, max/min values ($\mu \mp \sigma; min/max$) of all cross discriminant $J(w,t)$ are summarized in a table (cross Fisher linear discriminant table) shown in tables 4.10 and 4.11. Furthermore, this table can be aggregated (averaged of μ and σ ; maximum and minimum of min/max) in order to obtain a more compact representation. For example, the dataset W_Test, has the following values $0.06 \mp 0.15; 0.00/0.72$ and $0.002 \mp 0.01; 0.00/0.05$ for the parameters g=c=d=h=0 and g=c=d=h=0.2 (*irrel*

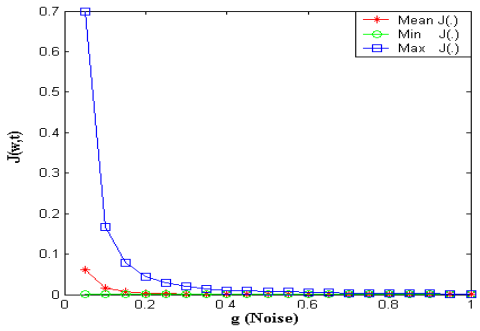


Figure 4.28: Linear discriminant analysis criterion function metrics (maximum, minimum and mean) for the dataset W_Test with various noise levels g, and *irrel* on, c=d=h=0.

	A $\mu \mp \sigma; min/max$	B $\mu \mp \sigma; min/max$	C $\mu \mp \sigma; min/max$
A	$0.00 \mp 0.00; 0.00/0.00$	$0.00 \mp 0.00; 0.00/0.00$	$0.00 \mp 0.01; 0.00/0.04$
B	$0.00 \mp 0.00; 0.00/0.00$	$0.00 \mp 0.00; 0.00/0.00$	$0.00 \mp 0.01; 0.00/0.05$
C	$0.00 \mp 0.01; 0.00/0.04$	$0.00 \mp 0.01; 0.00/0.05$	$0.00 \mp 0.00; 0.00/0.00$

Table 4.11: Cross Table of J(w,t) for the dataset W_Test when g=c=d=h=0.2 and *irrel* is on.

on) respectively. The tables 4.10 and 4.11 illustrate the cross Fisher linear discriminant analysis J(w,t) table for the dataset W_Test. The tables reflect the significant similarity between class A and B.

4.5.8 PCA-Based Similarity Measure (EROS- (*E*xtended F*ro*benius*s* norm))

PCA based similarity measurements are one of the most commonly used schemes for characterization of datasets. The intuition behind the schemes are based on the angle ($cos\theta$) between the principal components of two given matrices. Recently, a new improved PCA-based scheme *Eros* (*E*xtended F*ro*benius norm) was developed for disparity characterization[155]. In our study, Eros will be the main PCA-based technique to analyse the datasets. In the rest of the section, the results of the scheme upon the artificial dataset W_Test will be elaborated upon, following by a brief discussion on Eros. For a more detailed discussion on Eros, please refer to[155].

Eros uses weighted Frobenious (also known as Euclidean)norms to the eigenvector and eigenvalues of principal components which are obtained from the covariance of multivariate time series represented in matrices as follows:

Algorithm 1 Computing a weight vector w based on the distribution of raw eigenvalues

```
1: function computeWeightRaw(S)
Require: an n × N matrix S, where n is the number of variables for the dataset and N is the number of MTS items in the dataset. Each column vector s_i in S represents all the eigenvalues for ith MTS item in the dataset. s_ij is a value at column i and row j in S. s_*i is ith row in S. s_i* is ith column, i.e, s_i.
2: for i=1 to n do
3:    w_i ← f(s_*i);
4: end for
5: for i=1 to n do
6:    w_i ← w_i / Σ_{j=1}^{n} w_j;
7: end for
```

Algorithm 2 Computing a weight vector w based on the distribution of normalized eigenvalues

```
1: function computeWeightRatio(S)
Require: the same as Algorithm 1.
2: for i=1 to N do
3:    s_i ← s_i / Σ_{j=1}^{n} s_ij;
4: end for
5: computeWeightRaw(S);
```

Algorithm 3 Preprocessing Algorithm of *Eros*

```
Require: the number of all the MTS items in the dataset N.
1: for i=1 to N do
2:    A ← the ith MTS item in the database;
3:    B ← covariance matrix of A;
4:    [C, D, E] ← SVD(B);
5:    s_i ← the eigenvalues in D;
6:    store E as the ith right eigenvector matrix;
7: end for
8: Compute the weight vector w using Algorithm 1 or 2.
```

Algorithm 4 kNN Search Algorithm of *Eros* (Two-phase Sequential Scan)

```
Require: Given an MTS user query Q and a weight vector w
1: for i=1 to k do
2:    nndist[i] ← ∞;
3: end for
4: for i=1 to N do
5:    P ← the eigenvector matrix for the ith MTS item in the database;
6:    res ← D_min(P, Q, w);
7:    if res ≤ nndist[k] then
8:       res ← D_Eros(P, Q, w);
9:       if res ≤ nndist[k] then
10:         nndist[k] ← res;
11:         nnid[k] ← i;
12:         update nndist, nnid;
13:      end if
14:   end if
15: end for
```

Algorithm 5 Modified Leave-One-Out k Nearest Neighbor Search for Recall-Precision graph

```
Require: the number of MTS items in the dataset, N, k, the maximum number of relevant items, maxr;
1: for i=1 to 10 do
2:    precision[i] ← 0;
3: end for
4: for i=1 to N do
5:    Q ← ith item in the dataset;
6:    k ← 1;
7:    r ← 1;
8:    repeat
9:       Perform kNN search for Q;
10:      c ← the number of the same label items as Q in the k items retrieved;
11:      if c = r then
12:         precision[r] ← precision[r] + c / k;
13:         r ← r + 1;
14:      end if
15:      k ← k + 1;
16:   until r ≥ maxr;
17: end for
18: for i=1 to 10 do
19:    precision[i] ← precision[i] / N;
20: end for
```

Figure 4.29: Pseudo source code of Eros[155].

$$
\begin{aligned}
Eros(A, B, w) &= \sum_{i=1}^{n} w_i | < a_i, b_i > | \\
&= \sum_{i=1}^{n} w_i |cos\theta_i|
\end{aligned}
\tag{4.12}
$$

where a_i and b_i are column orthonormal vectors (eigenvectors) of size n (number of channels in this study ϑ),, $< . >$ is inner product operator w is a weight vector which is proportional to eigenvalues of datasets $\sum_{i=1}^{n} w_i = 1$, and $cos\theta_i$ is the angle between a_i and b_i. Eros ranges in the interval of [0,1] where 0 is less similar. Eros employs a weighted Frobenious norm to estimate the similarity or distance between two matrices

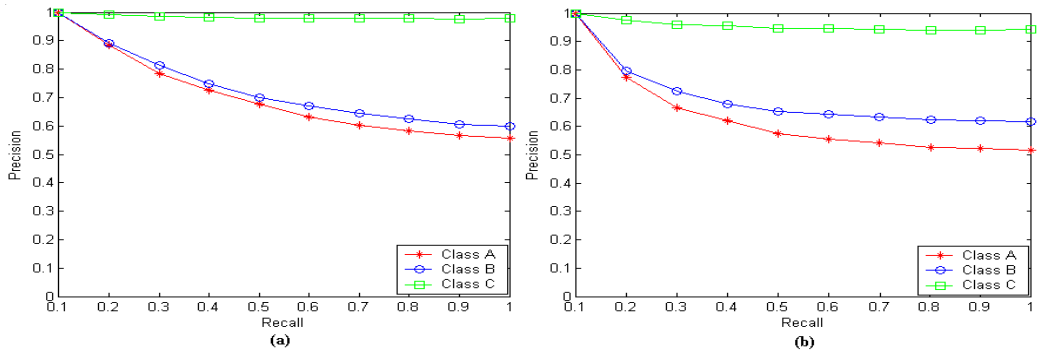


Figure 4.30: Recall/Precision of Eros for dataset W_Test with (a) g=c=h=d=0, (b) g=c=h=d=0.2 and *irrel on*.

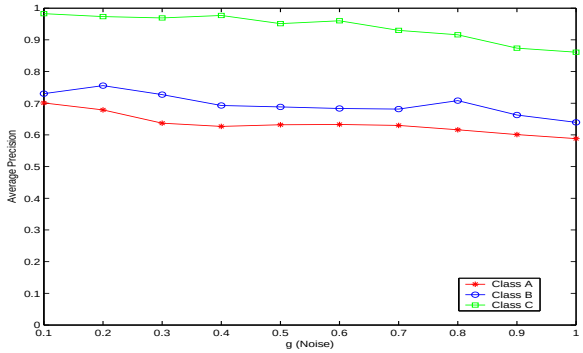


Figure 4.31: Average Precision of Eros (10 kNN) for dataset W_Test with different noise level (g) when c=h=d=0 and *irrel on*.

as follows:

$$
\begin{aligned}
D_{Eros}(A,B,w) &= \sqrt{2-2\sum_{i=1}^{n} w_i |< a_i, b_i >|} \\
&= \sqrt{2-2\sum_{i=1}^{n} w_i |\sum_{j=1}^{n} a_{ij} \times b_{ij}|}
\end{aligned}
\tag{4.13}
$$

The outline of the Eros is as follows: In the first phase, pre-processing, covariance matrices (two sample from different classes) of given two matrices are calculated. Eigenvectors and eigenvalues are then calculated based on the covariance matrices by using singular value decomposition (SVD). In further phases, only the right eigenvectors of the decomposition are used. The left eigenvectors are not used at all. In the second phase, the weight vector w is generated from the eigenvalues. In the final phase,

the similarity between corresponding eigenvectors and weights are estimated in terms of recall/precision rates for a given query from the class of interest and each elements of the dataset by a k nearest neighbour (k-NN) algorithm. The k-NN algorithm checks to see, in order to recall rth gesture of interest, how many gestures from database k (neighbourhood) should be retrieved. Hence, precision is defined as $p = \frac{r}{k}$. Figure 4.29 illustrates the pseudo algorithm, (phases) of EROS. In our study, as aggregation function $f(s_{*i})$ in weight estimator phase, the *mean* function is used.

As mentioned above, the right eigenvector of singular value decomposition is only used in Eros. That provides dimension reduction (from $m \times n$ to $n \times n$; $(\vartheta \times \vartheta)$) in computation and a common size eigenvector matrix - right eigenvector matrix is always $n \times n$ through the dataset.

Eros is applied to the artificial dataset W_Test. The figure 4.30 illustrates the recall/precision results for two different scenarios - when the parameters are g=c=h=d=0 and g=c=h=d=0.2 (*irrel on)*. In both scenarios, the similarity between class A and B is clearly demonstrated. Figure 4.31 demonstrates average precision results for various noise levels (0 *leqg* ≤ 1), c=h=d=0 and *irrel on*. Note the similarity between class A and B is preserved with various noise levels again.

4.5.9 Intersection

The gist of a classification task is to divide feature space into segments in a way (linear or non-linear) that each segment corresponds to a class. Intersection areas between segments play an important role in recognition. The shared area or hyper volume between the class' templates indicates a similarity metric. Hence, in this section, a novel technique is proposed to analyse the intersection area/volume in terms of the similarity between class' templates/channels.

Let us assume that C_1 and C_2 are two classes with ϑ dimension/channels. $C_{i,j}$ is the j^{th} channel of class i with period of L_i. Please note that a channel $C_{i,j}$ is defined by two series of discrete components with mean and standard deviation $(\mu_{i,j,t}, \sigma_{i,j,t})$. Furthermore, it is assumed that, at a certain time (t) in a channel, data is distributed normally $D_i(\mu_{i,j,t}, \sigma_{i,j,t})$ and channels are assumed independent of each other. Under these assumptions, at the given time t, the similarity $\zeta_{i,k,j,t}$ between the channel j of two classes (C_i, C_k) and overall similarity $\zeta_{i,k}$ can be estimated by taking the intersection of two distributions at the time t as follows:

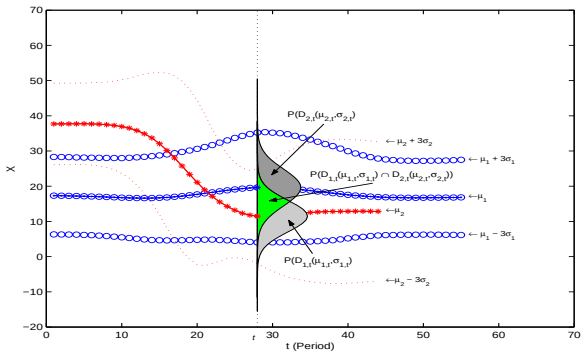


Figure 4.32: Intersection probability/area ($\zeta_{1,2,j,t}$) of two gaussian distributions at a certain time t of one dimensional discrete channels for imaginary class C_1 and C_2.

$$
\begin{aligned}
\zeta_{i,k,j,t} &= P(D_i(\mu_{i,j,t}, \sigma_{i,j,t}) \cap D_k(\mu_{k,j,t}, \sigma_{k,j,t})) \\
\zeta_{i,k,t} &= \sqrt[\vartheta]{\prod_{j=1}^{\vartheta} \zeta_{i,k,j,t}} \\
\zeta_{i,k} &= \frac{1}{L_{min}} \sum_{t=1}^{L_{min}} \zeta_{i,k,t} \qquad L_{min} = min([L_i, L_k]) \;\; 0 < t \leq L_{min}
\end{aligned}
\tag{4.14}
$$

Figure 4.32 illustrates the intuition behind the scheme at a certain time t on a channel $H_{1,j}$ and H_2, j for the classes C_1 and C_2. Although the intersected area can be estimated theoretically, in our study an empirical approach is applied as follows. Since the necessary parameters of the underlying distributions at a certain time ($\mu_{i,j,t}$ and $\sigma_{i,j,t}$) are known, at the first phase, sample observation populations $O_{*,j,t}$ for all distribution $D_{*,j,t}$ are created. Based on these sample populations, the intersection of two distributions ($\zeta_{i,k}$) of $D_{i,j,t}$ and $D_{k,j,t}$ are estimated as follows:

$$
\zeta_{i,k,j,t} = \frac{\#\ of\ samples\ which\ P(D_{i,j,t}(O_{k,j,t})) > 0}{total\ population\ size\ O_{i,j,t}} \tag{4.15}
$$

where robustness of the empirical estimation is strongly depended on large population size. This definition yields unsymmetrical similarity metrics. In other words, $\zeta_{i,k}$ is not necessary equal to $\zeta_{j,k}$. In fact, the definition of $\zeta_{i,k}$ can also be expressed as conditional probability as follows:

$$\begin{aligned}\zeta_{i,k,j,t} &= P(D_{k,j,t}(O_{i,j,t})|D_{i,j,t}(O_{i,j,t})),\\ &= \frac{P(D_{k,j,t}(O_{i,j,t}) \cap P(D_{i,j,t}(O_{k,j,t})))}{P(D_{i,j,t}(O_{i,j,t}))}\end{aligned}$$

Tables 4.12 and 4.13 illustrate cross similarity $\zeta_{i,k}$ among classes. The scheme confirms the high similarity between class A and B of W_Test dataset.

	A $\mu \mp \sigma; min/max$	B $\mu \mp \sigma; min/max$	C $\mu \mp \sigma; min/max$
A	$1.00 \mp 0.00; 1.00/1.00$	$0.98 \mp 0.14; 0.00/1.00$	$0.08 \mp 0.06; 0.00/0.16$
B	$0.98 \mp 0.14; 0.00/1.00$	$1.00 \mp 0.00; 1.00/1.00$	$0.08 \mp 0.06; 0.00/0.17$
C	$0.08 \mp 0.06; 0.00/0.16$	$0.08 \mp 0.06; 0.00/0.16$	$1.00 \mp 0.00; 1.00/1.00$

Table 4.12: Cross similarity (ζ) for artificial dataset W_Test when g=c=h=d=0 and *irrel on*.

	A $\mu \mp \sigma; min/max$	B $\mu \mp \sigma; min/max$	C $\mu \mp \sigma; min/max$
A	$1.00 \mp 0.00; 1.00/1.00$	$0.99 \mp 0.10; 0.00/1.00$	$0.98 \mp 0.10; 0.00/1.00$
B	$0.99 \mp 0.10; 0.00/1.00$	$0.99 \mp 0.10; 0.00/1.00$	$0.98 \mp 0.10; 0.00/1.00$
C	$0.98 \mp 0.10; 0.00/1.00$	$0.98 \mp 0.10; 0.00/1.00$	$0.99 \mp 0.10; 0.00/1.00$

Table 4.13: Cross similarity (ζ) for artificial dataset W_Test when g=c=h=d=0.2 and *irrel on*.

These tables can be shown in the form of a scaled image graph as in figure 4.33 and 4.34 respectively. A scaled image graph maps the elements of a matrix to a specific colour, using a colour map. For a given matrix of size $m \times n$, a rectangular patch is divided into mn sub-patches. An element of the matrix is then mapped to a specific colour. The sub-patch at location i, j is rendered using this colour. The colour bar shown on the right of the image graph shows the range of each sub-patch. Normally, the column names are shown on the axes. For example, figure 4.33 is used to illustrate the average values (μ) from table 4.12 which shows the intersection similarities in the format of $\mu \mp \sigma$. The average intersection similarity value between class A and C is 0.08 ignoring the variance (See table 4.12 shows). This corresponds to blue-like colour in the colour bar. In order to show average values for the columns and rows, an additional row and column is added to the image graph. In further chapters, scaled image graphs representation will also be used to illustrate intersection similarity matrices as well as other analyses, for the sake of clarity. The axes labels change according to the analysis. Generally in these analyses, the Y axes consist of the classes and the X axes consist of

the features. But in the case of inter cross similarity and complexity analysis, both X and Y axes correspond to the class labels.

As figures 4.33 and 4.34 illustrate, similarity matrices are not symmetric. The last row and column of the matrix represents the average of the respective columns and rows. Each cell represents the ratio of intersection volume ($\zeta_{i,k}$) between row (C_i) and column (C_k) class. The similarity ratio can be also explained in terms of the degree of *encapsulation* and *subset power*, which will be more meaningful over other datasets than W_Test. Each cell indicate the encapsulate power of the row class (C_i) to the column class (C_k): $C_k \subset C_i$. In other words, how much does row class C_k cover column class C_i. On the other hand, subset power is the degree of C_k being encapsulated by C_i. Therefore, while the last column of the matrix indicate the average encapsulate power of the row class to other classes, $C_i \supset C$; the last row indicate, average subset power of the column class (C_k) to other classes $C_k \subset C$.

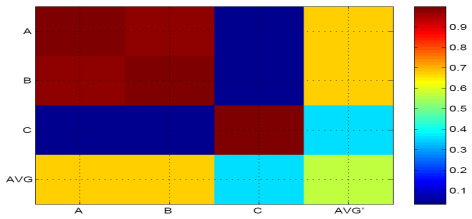


Figure 4.33: Cross similarity (ζ) W_Test dataset when g=c=h=d=0 and *irrel on* for table 4.12.

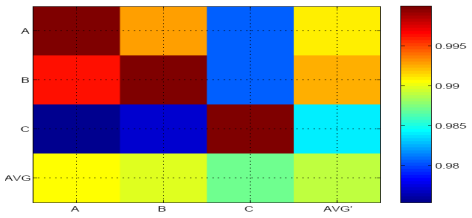


Figure 4.34: Cross similarity (ζ) W_Test dataset when g=h=d=0.2, c=0.1, and *irrel on* for the table 4.13.

For example, with regards to the last column and row in figures 4.33 and 4.34, class A and class B have more encapsulation and subset power than class C. In fact, due to the small of number classes and the high similarity between class A and B, the encapsulation and subset power on W_Test is not that expressive. But as will be elaborated in the next section, encapsulation and subset power in Intersection analysis on a FDO dataset articulate more information.

4.5.10 Periodical and Index Variance

So far, we have focused on the characteristics of data in R^{ϑ} dimensional feature space. In addition, for a class, the variance of the index of sub-events (SEVP) in samples and the variance of sample's periods (PPV) for each class can be analysed. In fact, these variances are mostly caused by temporal variance. Therefore, they can characterise temporal aspects of datasets.

4.5.10.1 Sub-Event Variance Percentage (SEVP)

As mentioned in the problem definition and gesture modelling sections, sub-events are characteristic events which occur around a certain index. Proportional to temporal variance, the location of sub-events can be shifted either side. Sub-Event Variance Percentage (SEVP) corresponds to the average variance in the sub-event's index to total period of class (L_i), such as the control parameter (c) of the artificial W_Test dataset.

Localization of sub-events is a non-trivial task which is domain and class dependent. This localization task, in particular, is more complex in the case of a real word dataset compared to artificial datasets. In most cases, such as c in the W_Test, a control parameter is assigned for sub-event variances.

As mentioned before, in the class modelling section, for each class, a supervised and ad-hoc scheme is applied to find k^{th} sub-event's index ($SE_{i,j,s,k}$) of channel $H_{i,j}$ in sample s of class C_i in order to align all k^th sub-events at same index $SE_{i,j,k}$ in the channel $H_{i,j}$. Once the sub-event's index ($SE_{i,j,s,k}$) is extracted, the sub-event variance percentage ($SEVP_i$) for class C_i is estimated as follows:

$$\begin{aligned} \delta_{i,j,k} &= \frac{1}{\#S_i}\sum_{s=1}^{\#S_i}|SE_{i,j,k} - SE_{i,j,s,k}| \\ SEVP_i &= \frac{max(\delta_{i,j,k})}{L_i} \quad 0 < k \leq K_j; 0 < j <\leq \vartheta \end{aligned} \tag{4.16}$$

where K_j is the total number of sub-events in the channel j; $\#S_i$ is number of samples in the training set and L_i is the period for the class C_i. Maximum mean variance $\max(\delta_{i,j,k})$ among all channels is considered for the SEVP estimation. Since the control parameter c corresponds to SEVP for the artificial W_Test dataset, the SEVP analysis on that dataset is omitted here.

4.5.10.2 Period Variance Percentage (PVP)

Similar to the sub-event variance percentage (SEVP), period variance (PVP) indicates the percentage duration change among the training datasets. PVP_i for class C_i is estimated as follows:

$$PVP_i = \frac{1}{L_i \# S_i} \sum_{s=1}^{\#S_i} |L_i - L_{i,s}| \tag{4.17}$$

where $L_{i,s}$ and L_i corresponds to the length of sample s in the training dataset (containing $\#S_i$ sample) and period for the class C_i. The control parameter d in the W_Test dataset corresponds to Period Variance Percentage (PVP) for the W_Test dataset.

4.6 Analysis of FDO Gestures

In the previous sections, general schemes for analysing a temporal dataset were presented. In this section, FDO gestures will be analysed in more detail and in contrast present the scheme with some related conventional techniques.

For collecting FDO gestures, two different acquisition schemes have been applied: Tracker (FDO_PT) and Computer Vision (FDO_CV). The main differences between these two dataset is that, for data acquisition, while FDO_PT uses only one person (the author), FDO_CV uses four different persons.

Both datasets consist of 18 gestures out of a total of 94 FDO gestures. These gestures accommodate all the challenges one would expect to come cross during FDO's gesture recognition. Each dataset consists of four static gestures (Affirmative, Clean, Hold On, Negative), six dynamic gestures (Ahead, Back, Wave Off, Down ...) and eight hybrid gestures (Left, Right, Fire ...). Please refer to the problem definition chapter and related appendix for more detailed information about the FDO gestures.

In the following subsection these two datasets will be analysed in more detail.

4.6.1 Tracker-based FDO Gestures (FDO_PT)

FDO_PT is collected via a tracker device (Polhemus FasTrak) of which two sensors acquire the position of hands in a three dimensional coordinate system (x, y, z). FDO_PT consists of about 150 samples for each class and these samples are collected only from a single person in different sessions. As explained in the modelling section, each raw gesture is represented by a stream of six coordinate data (x, y, z), a triplet for each

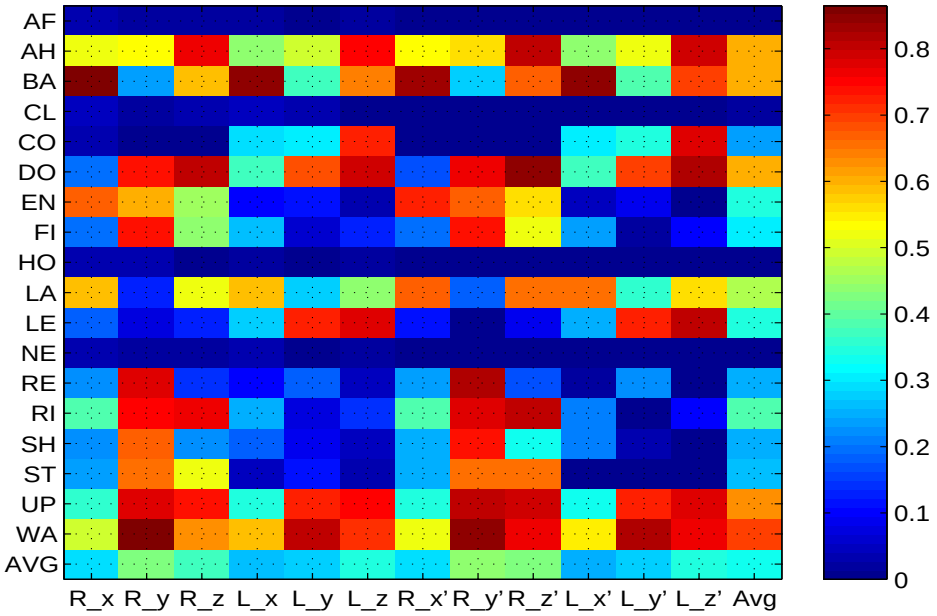


Figure 4.35: Normalized Channel Entropies of samples in FDO_PT dataset.

hand. The coordinates of the right and left hands (x, y, z) are then normalized to map to a fixed grid as was explained in the feature selection section, and the gesture data is represented in feature space as:

$$F_{FDO\_PT} = F_{Grid} = [R_x, R_y, R_z, L_x, L_y, L_z, R'_x, R'_y, R'_z, L'_x, L'_y, L'_z]$$

where $[R_x, R_y, R_z, L_x, L_y, L_z]$ corresponds to spatial grid features for the Right and Left hand and $[R'_x, R'_y, R'_z, L'_x, L'_y, L'_z]$ accommodates the fuzzy gradient temporal feature.

- Entropy
  - Horizontal Channel (Feature) and Class Entropy ($h\bar{H}X$): Figure 4.35 and 4.36 illustrate the normalized channel and class entropies for the FDO_PT dataset, respectively. Note that class entropies are estimated based on class models, whereas channel entropies utilise the samples of the class of interest. In fact, since the class model is constructed out of samples, it is expected that the channel and class entropies are in agreement, which is clearly illustrated in figures 4.35 and 4.36. The last row and columns of the figures indicate the average channel (features) entropies and classes for the FDO_PT dataset respectively.

    Another important observation about these figures is the entropy of static, hybrid and dynamic gestures. As it can be seen from the figures, while

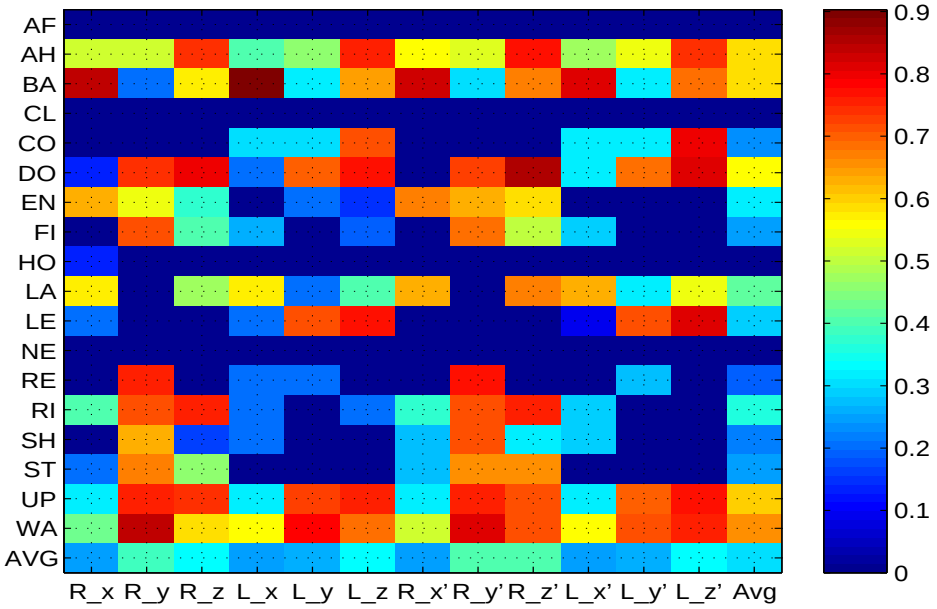


Figure 4.36: Horizontal Class Models Entropies for FDO_PT dataset.

static gestures (Affirmative, (AF), Clear (CL), Hold on, (HO) and Negative (NE)) have low channel/class entropy, hybrid gestures accommodate medium channel/class entropy, and dynamic gestures accommodate the highest channel/class entropies. Therefore, the recognition algorithm will be more successful with dynamic, hybrid and static gestures in that order, in terms of predicting the next frame index.

Channel and class entropy figures also show the dominant channels for each class and dataset. y and z channels for both hands (especially the right) and their gradients accommodates most of the entropies in the dataset. In case of hybrid gestures, for example, the left hand during the *Right* gesture, the static hand (left) accommodates slightly more entropy than either the static hand of a static gesture, due to the effect of dynamic hand to static hand in hybrid gestures.

– Vertical (Frame) Entropy ($h\bar{V}X$): Figure 4.37 illustrates the average normalized frame entropies of features and class for the FDO_PT dataset. Note that, similar to channel entropy estimation, frame entropy estimates are based on the samples of classes.

Unlike channel and class entropies, frame entropies accommodate high entropies apart from the temporal channel of static gestures. Even for static gestures, high entropies on spatial channel are observed. This phenomenon

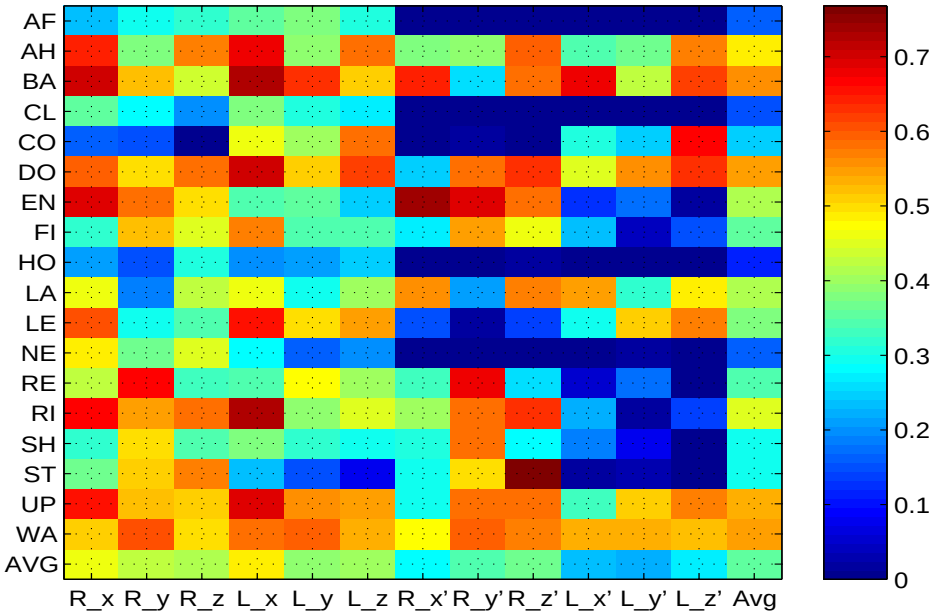


Figure 4.37: Normalized Frame Entropies of samples in FDO_PT dataset.

can be associated with large intra user variance such that when static gestures are performed, the user keeps his hands in various places at the beginning of the gesture. High frame entropy yields large band width to class modules (templates) which degrade discrimination power, namely the recognition results.

– Cross Mutual Matrix (CMT): Mutual information, in this thesis, is used to measure how much class model C accommodates or shares information with samples in the dataset. Normalized horizontal channel and class entropy schemes are used for cross mutual information estimation between samples and class models for the FDO_PT dataset which is illustrated in figure 4.38.

  Since the class entropy of static gestures is zero, consequently, mutual information between these gestures and with all other samples is zero. Note $0 \leq I(X,Y) \leq min(H(X), H(Y))$. In an ideal case, entropy values in diagonal cells have higher entropies on the same row, namely between samples and their associated classes. Unfortunately, the analysis indicates that some dynamic and hybrid gestures such as *Ahead*, *Down*, *Up* and *Wave Off* share large information not only with their own samples but also with other class samples. This indicates a higher inter similarity in the FDO_PT dataset.

– Mutual Information Precision (MIP) and Noise Signal Ratio (NSR): MIP provides a summary of cross a mutual matrix (figure 4.38). Note that, MIP

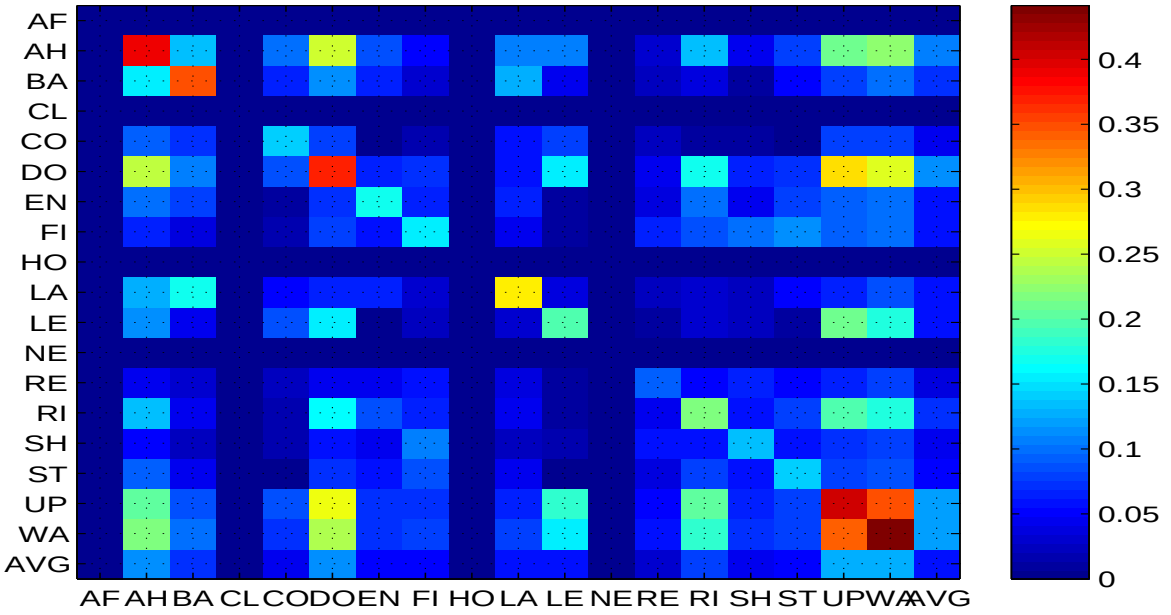


Figure 4.38: Cross mutual information between samples (Y axis) and class models (X axis) in the FDO_PT dataset.

ranges between [0,1] and the highest value (1) indicates a higher similarity between samples and their associated class in the dataset. The higher the MIP value, the more distinctive the class. Similarly, NSR indicates the content of useful information among a class sample compared to its class model. It is the ratio of non useful information (noise) to useful information. The higher values of NSR imply irrelevance levels of samples and their associated class model.

The FDO_PT dataset accommodates 0.42∓0.36;0/1 (mean ∓ standard deviation) and 0.55∓0.24;035/1 (minimum/maximum) MIP and NSR respectively. As these values show, the FDO_PT dataset consists of high inter similarity between samples and with other class models (MIP) and noise between samples and their associated class (NSR).

	$h\bar{C}$	$h\bar{H}X$	$h\bar{V}X$	$I(hHX, hC)$	$M\bar{I}P$	$N\bar{S}R$
Mean	0.31	0.33	0.36	0.19	0.42	0.55
Std	0.20	0.19	0.17	0.09	0.36	0.24
Min	0.00	0.00	0.00	0.00	0.00	0.35
Max	0.90	0.86	0.77	0.61	1.00	1.00

Table 4.14: Summary of normalized entropy analysis of the FDO_PT dataset. $h\bar{C}$, $h\bar{H}X$, $h\bar{V}X$, $I(hH\bar{X}, hC)$, $M\bar{I}P$, $N\bar{S}R$ correspond to normalized horizontal class, channel, frame entropies, normalized cross mutual information, MIP and NSR respectively.

Table 4.14 summarizes the entropy analysis of the FDO_PT dataset. The table shows the mean, the minimum and maximum normalized horizontal class entropy ($h\bar{C}$), channel entropy ($h\bar{H}X$), frame entropy ($h\bar{V}X$), cross mutual information ($I(hH\bar{X}, hC)$), Mutual Information Precision ($M\bar{I}P$) and Noise Signal Ratio ($N\bar{S}R$) for the dataset.

- Chi-Squared Test (χ^2) and Skewness/Kurtosis: Note that in this thesis, it is assumed that underlying the statistical distribution at each frame of samples of a class is Gaussian. The Chi-Squared test is carried out to verify this assumption. In other words,the Chi-squared test aims to verify the following null and alternative hypotheses (H_0, H_1):

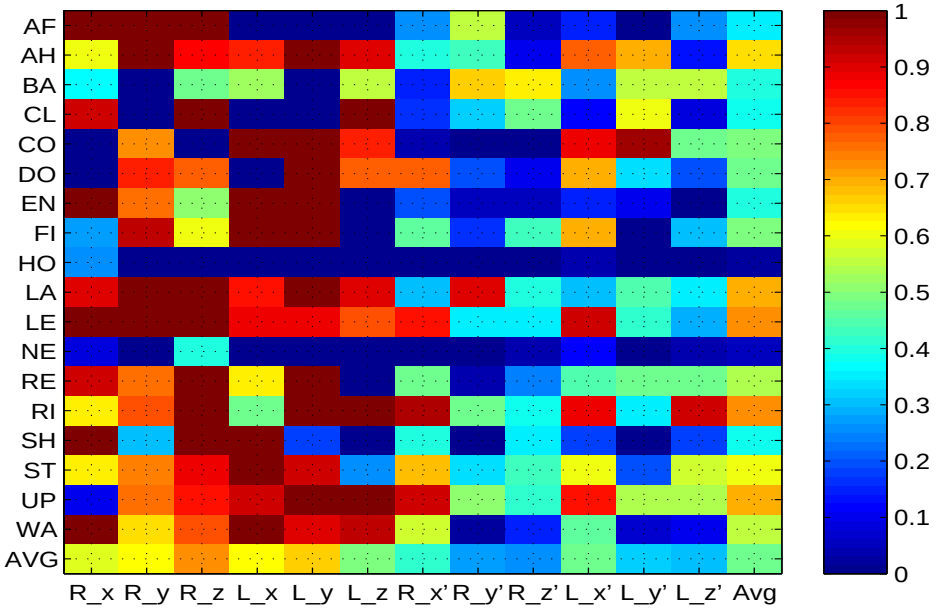


Figure 4.39: Chi2Test results for each channel and class for the FDO_PT dataset. Each cell indicates the ratio of frames in the channel of row class (C_i), which supports the null hypothesis (namely Gaussian distribution), to all the number of frames in the channel (L_i, or period of class, C_i). The highest ratio (1) indicates that all frames in the channel supports the null hypothesis. While the non constant channel of the spatial channel of the dynamic and hybrid channels mostly support the null hypothesis, due to used a model construction scheme, temporal channels do not support the null hypothesis (not completely Gaussian distribution).

$$H_0 \ : \ \text{Distribution at each frame is Gaussian}$$

$$H_1 \ : \ \text{Distribution at each frame is unknown}$$

The Chi-Square test is applied over the frames of all channels of the FDO_PT dataset. Figure 4.39 illustrates the ratio Chi-Square test results in each channel which supports the null hypothesis. For example, a value of 0 means none of the frames in the channel is Gaussian, 1 indicates all frames in the channels are Gaussian.

Two main observations are obtained from this analysis. The first of which is that unlike spatial channels ($R_x, L_x, R_y, L_y, R_z, L_z$) temporal channels do not show a complete Gaussian distribution behaviour(H_1). This can be attributed to the fact that, during construction, temporal channels are bound to within the range of [-1,1], by mapping the gradients, which is bigger than 1 and less than -1, to 1 and -1 respectively.

The second observation from figure 4.39 is that static gestures and static channels of dynamic gestures (for example x channel of *Up* and y of *Back* gestures) do not support the null hypothesis. This can be explained simply that in an ideal case, the static channel of either static or dynamic gestures at a frame is distributed around a certain point. Therefore, normal distribution may not occur in these channels. But as the figure shows (affirmative R_x, R_y, R_z, clear R_x, engage L_x,L_y and L_z and Wave off R_x), due to noise, intra variance and inefficiency of the input device (Polhemus FastTrak), a fake Gaussian distribution emerges in some channels.

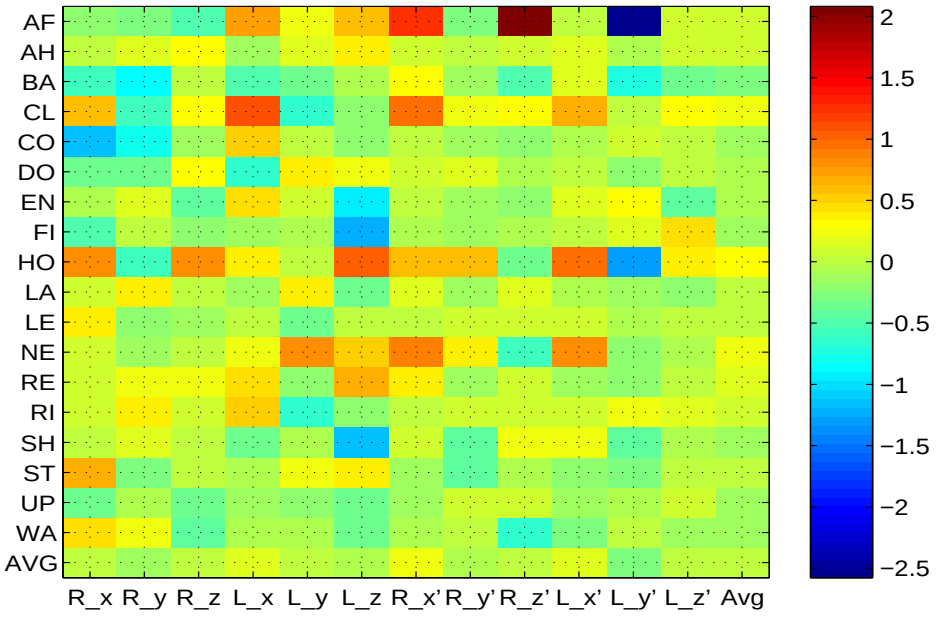


Figure 4.40: Skewness results for FDO_PT dataset.

Figure 4.40 and 4.41 illustrate the skewness and kurtosis of FDO_PT data for the channels. Note that, for Gaussian distribution, skewness should be around zero (symmetric) and kurtosis is three. A value less than three for kurtosis indicates a uniform, flatter distribution. While skewness indicates the direction of scattering, kurtosis indicates the shape of distribution (uniform, bell).

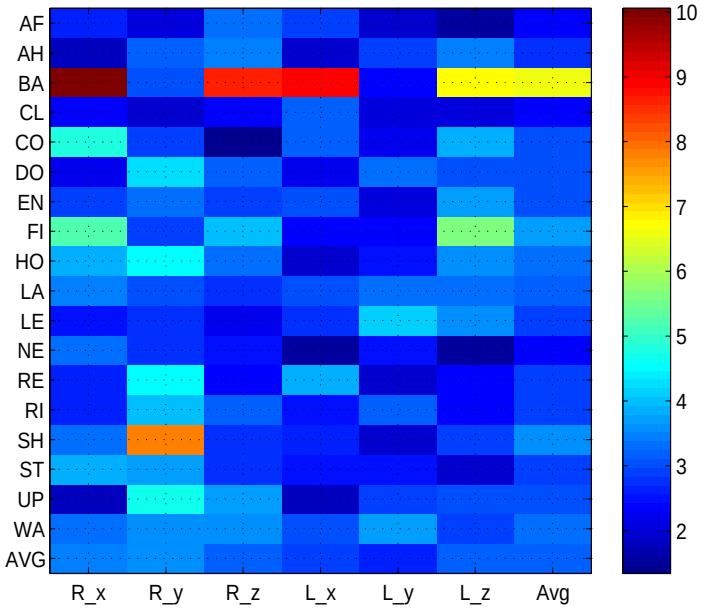


Figure 4.41: Kurtosis results for FDO_PT dataset.

Similar to the Chi-Square test, skewness and kurtosis are estimated at each frame and the average of the channel is shown in figures 4.40 and 4.41. For kurtosis analysis, temporal channels are omitted due to the fact that temporal channels have large kurtosis (around 70) which overshadows the spatial channels. Large kurtosis emerges especially in temporal channels of static gestures (for example *Hold on*) because, as was shown in the Chi-Square test analysis, temporal channels do not show a complete Gaussian distribution behaviour, due to its channel construction scheme.

Skewness and kurtosis tables are in agreement with the Chi-Square test results in terms of temporal/spatial channel and static channels outputs. As figure 4.40 shows, the skewness of dynamic gestures is around 0, in other words, distribution is symmetrical, whereas the static gestures and static channel of dynamic and hybrid gestures are scattered around a value of 0. On the other hand, kurtosis analysis (figure 4.41) points out a uniform, flatter distribution for the static spatial channel of FDO_PT gestures. Whereas, dynamic gestures have high kurtosis,

suggesting a bell shape distribution (Gaussian).

- Fisher Linear Discriminant Analysis : This analysis aims to exploit a cross linear discriminant degree between class samples. Figure 4.42 illustrates the cross class mean Fisher linear discriminant analysis (J(w,t)) for the FDO_PT dataset. Note that, J(w,t) indicates the ratio of the *between class scatter*, S_B and *within class scatter* S_W and it is not normalized. The higher the value of J(w,t), the higher the discriminant degree between two classes.

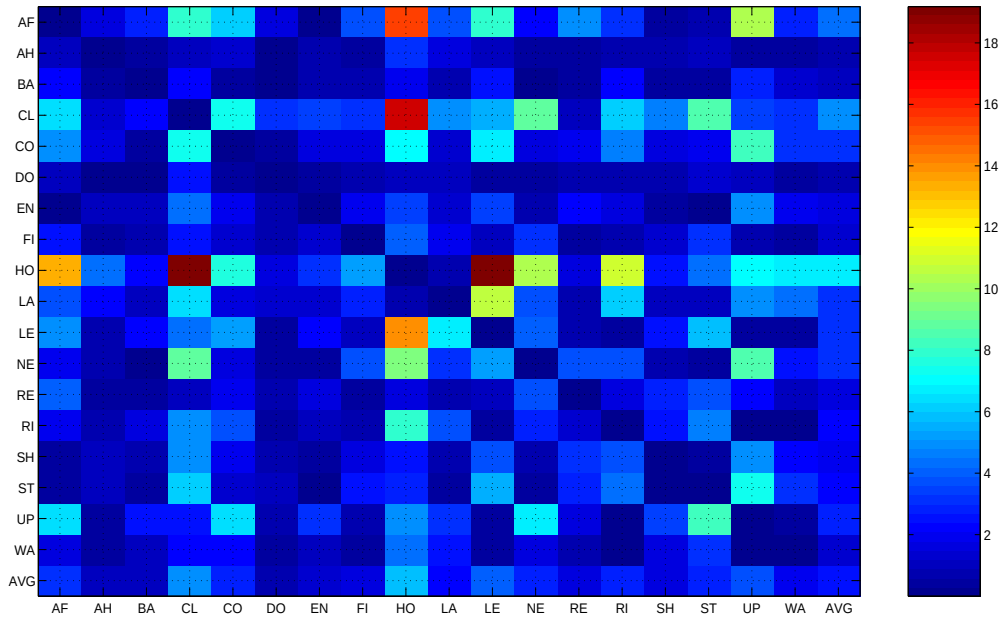


Figure 4.42: Fisher Linear Discriminant Analysis J(w,t) results for the FDO_PT dataset.

The analysis (figure 4.42) shows the minor discriminant degree between the gestures apart from static gestures. The figure shows that among all other static gestures, the *Hold On* gesture has in particular, the largest discriminative power in the FDO_PT dataset. This can be attributed to the fact that, as in normalized frame (vertical) entropy analysis discussed above (figure 4.37), static gestures have low vertical entropy, in other words, static gestures do not accommodate high inter-scattering. In addition, temporal channels of static gestures play an important role for this discrimination among static gestures. But on the other hand, dynamic and hybrid gestures, unfortunately, accommodate less discrimination in the FDO_PT dataset.

- PCA-Based Similarity Measure (EROS) Eros (*E*xtended F*ro*benius*s* norm) employs a weighted Frobenious norm to the eigenvector and eigenvalues of principal

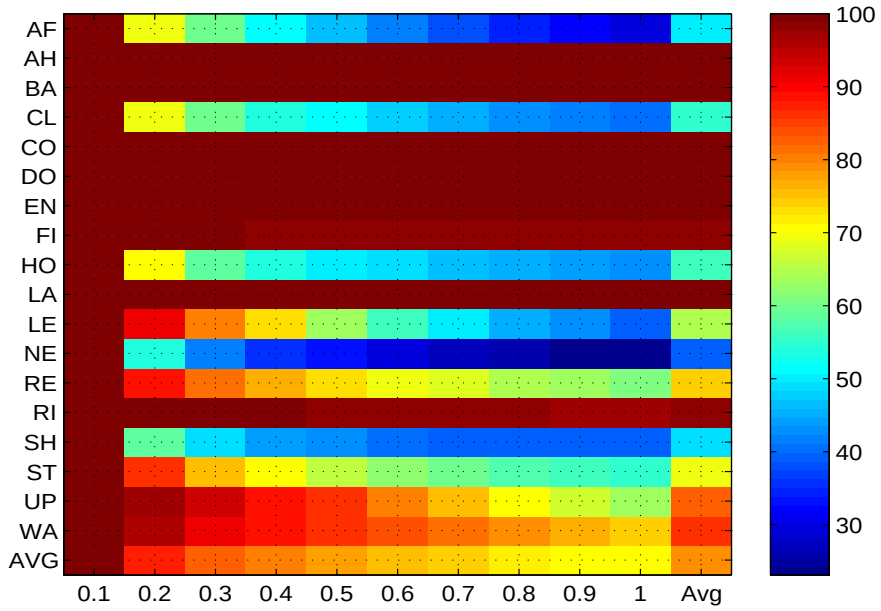


Figure 4.43: Recall/precision rate for the FDO_PT dataset by PCA-based similarity measurement (EROS). X axis corresponds to recall (r) value which is the number of gestures of interest that are retrieved in the neighbourhood of k. Precision rates ($p = r/k$) of static gestures are low, due to similar behaviour of covariance matrix of their samples on which EROS is based. Figure 4.44 illustrates EROS-based cross similarity among static gestures.

components which are obtained from covariance of processed samples (channels) of gestures which are converted to features and represented as a matrix (column vector for each channel). For similarity measurement, the k-nearest neighbourhood (k-NN) scheme is used to illustrate the similarity in recall/precision rates. Note that the k-NN algorithm checks, in order to recall *r-th* gesture of interest, how many gestures from database k (neighbourhood) are retrieved. Consequently the precision rate is $p = \frac{r}{k}$.

Figure 4.43 illustrates the recall/precision rates for the FDO_PT dataset. The X axis corresponds to the recall rate and each cell corresponds to the precision rate. Note that in order to be in agreement with the paper EROS is used [155], recall notation is kept the same. Actually, the recall value is in the range of [1,10], namely $r \times 10$ of the x axis in figure 4.43.

Unlike the Fisher linear discriminant analysis as explained above, the figure depicts a low precision rate ($p = r/k$) for static gestures. This can be attributed to the covariance matrices of samples on which EROS is established. Similarity

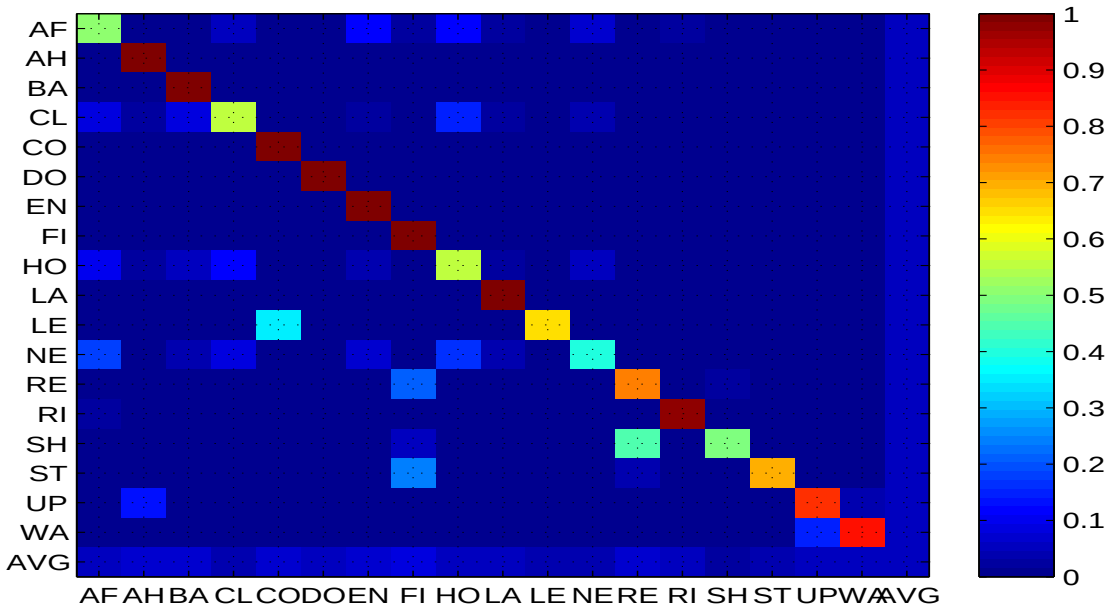


Figure 4.44: EROS-based cross gesture similarity for the FDO_PT dataset. Similarity among static gestures is high.

among covariance matrix, consequently eigenvector and eigenvalues of static gestures are high due to the similarity in spatial and especially temporal channels of static gestures. This similarity between static gestures reduces the precision rate dramatically.

In order to point out the similarity among static gestures, EROS-based cross gesture similarity precision is illustrated in figure 4.44. The EROS-based gesture similarity precision between row gesture C_i and column gestureC_j is illustrated with cells which is $\frac{r_j}{k}$ where r_j is the number of C_j gesture in the neighbourhood of k which consist of r gestures C_i. Recall value is in the range of [1,10] which can be seen in figure 4.43 and the average of all recall values are shown in figure 4.44. This corresponds to the density of class C_j in the neighbourhood of k. The matrix is in normalized form in which the sum of each row is 1.

The figure highlights the similarity between static and hybrid gestures. But on the other hand, dynamic gestures are quite distinctive. It can be concluded that, correlation or covariance between channels can be beneficial for only dynamic gesture recognition. The used algorithm must consider the spatial properties of static and hybrid gesture for a reliable recognition system. Bear in mind, unlike our on-line recognition algorithm, EROS is an off-line recognition scheme in which all channel data must be ready in advance before the estimations.

- Intersection Similarity: This scheme measures the shared area/or hyper volume among class models. Figure 4.45 depicts the cross class similarity of the FDO dataset by using an Intersection Similarity scheme. Note that the Intersection similarity matrix is not symmetric. The last row and column of the matrix represents the average of the respective columns and rows. Each cell represents the ratio of the intersection volume ($\zeta_{i,k}$) between row (C_i) and column (C_k) class. As explained before, this ratio can be also explained in terms of encapsulation and subset power. From this perspective, each cell indicates the encapsulate power of the row class (C_i) to the column class (C_k): $C_i \subset C_k$. In other words, how much row class C_k covers column class C_i. On the other hand, subset power is the degree of C_k being encapsulated by C_i. Therefore, while the last column of the matrix indicates average encapsulate power of the row class to other classes, $C_i \supset C$; the last row indicate, average subset power of the column class (C_k) to other classes $C_k \subset C$.

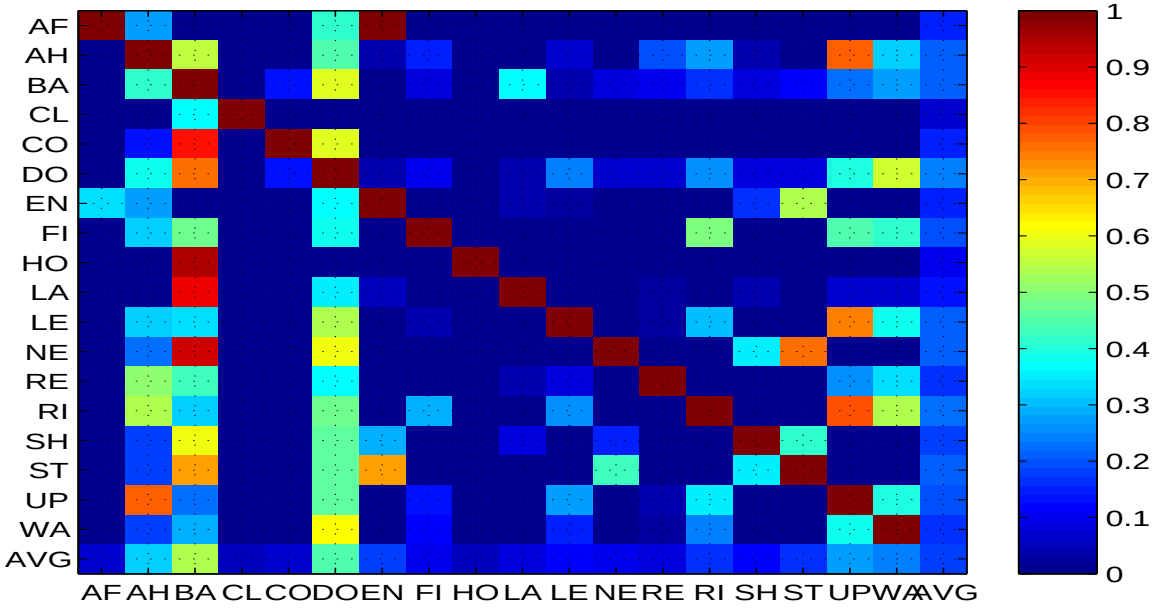


Figure 4.45: Cross class similarity of the FDO_PT dataset using the Intersection Similarity scheme.

Figure 4.45, for instance, pictures that *Ahead*, *Back*, *Down*, *Up* and *Wave Off* gestures have more encapsulation power over other gestures, especially static gestures. Another important observation is that, for example, the *Affirmative* gesture is a fully subset of the *Engage* gesture in feature space. But, bear in mind, the temporal aspect of the Engage gesture as was shown in the EROS analysis is a distinctive property of the recognition process.

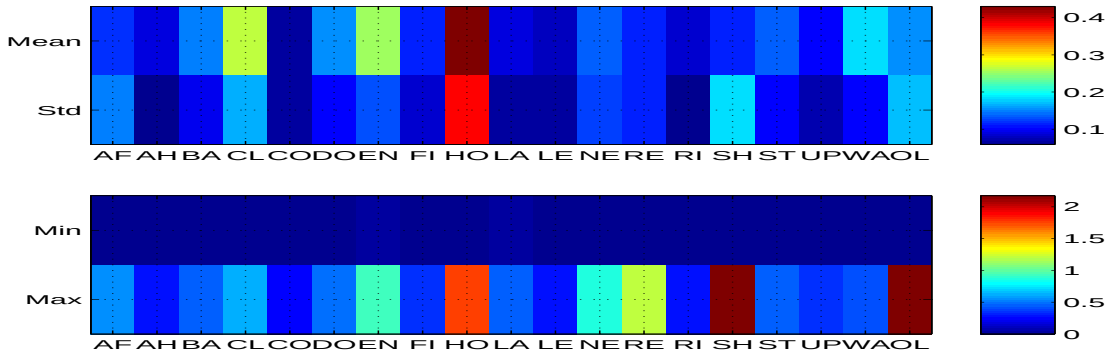


Figure 4.46: Period Variance Percentage (PVP) of the FDO_PT dataset. The last column of figures shows the respective overall (OL) aggregation of the respective rows.

- Sub-Event Variance Percentage (SEVP) and Period Variance Percentage (PVP): As elaborated in the gesture modelling section of this chapter, since channels in the FDO_PT dataset do not accommodate sub-events, sub-event variance percentage analysis (SEVP) is omitted and only period variance percentage analysis (PVP) is included here. Figure 4.46 illustrates the mean, standard deviation, maximum and minimum PVP of each class in the FDO_PT dataset. The last column of the figure corresponds to the overall (OL) aggregation of the respective rows, or more precisely the FDO_PT dataset. PVP analysis is shown in two subplots in figure 4.46, in order to prevent Max rows to overshadow other rows.

  The mean∓Std, Min/Max PVP for FDO_PT data is 1.56∓0.17,0/2.18. The dataset contains high temporal variance. Static gestures, especially, have high PVP. This is due to the data collection process, as was explained in the data collection section earlier in this chapter. Since static gestures do not have a cyclic movement to indicate the start/end points. Sometimes it is missed to indicate the end of static gestures on the decided average static gesture duration (4-6 seconds). Note the high value of PVP for the *Engage* gesture as well.

Summary of Tracker-based FDO Gestures (FDO_PT) Dataset Analysis

The first analysis applied over FDO_PT is entropy, in which class, channel, frame complexities are measured. Further, mutual information between samples and class models, (cross mutual information table), mutual information precision and noise signal ratio are investigated. As summarized in table 4.14, the FDO_PT dataset contains generally low channel and class entropy in static gestures. This can cause an issue in proposed recognition algorithm in terms of the prediction of the next frame index.

Moreover, the dataset has a high frame (vertical) entropy which directly implies a large bandwidth for modelling the class. Consequently, due to this large bandwidth, intersection similarity among classes increases, which leads to a low recognition performance. Cross mutual information analysis illustrates that dynamic gesture classes not only share high information with their samples, but also with static and hybrid gestures samples. The dataset also contains low mutual information precision and high noise to signal ratio, both of which are indicators of irrelevance between samples and their associated class model.

Two main observations are obtained from the Chi-Square test analysis. The first of which is that unlike spatial channels ($R_x, L_x, R_y, L_y, R_z, L_z$) temporal channels do not show a complete Gaussian distribution behaviour(H_1). This can be attributed to the fact that, temporal channels are forced to stay within the range of [-1,1] during their construction. The second observation is that static gestures and static channels of dynamic gestures (for example, x channel of Up and y of *Back* gestures) do not support fully the null hypothesis, normal distribution. The reason behind this is that in an ideal case, static channels of either static or dynamic gestures at a frame are distributed around a certain point. Therefore, normal distribution may not occur in these channels. This is caused due to noise, intra variance and inefficiency of the input device (Polhemus FastTrak) which creates a fake Gaussian distribution in some channels.

Skewness and kurtosis analysis are in agreement with the Chi-Square test results in terms of temporal/spatial channels and static channels. Skewness of dynamic gestures is around 0, in other words, they have symmetric behaviour, whereas static gestures and static channels of dynamic and hybrid gestures are scattered. On the other hand, kurtosis analysis points out a uniform, flatter distribution for the static channel of FDO_PT gestures. Whereas, dynamic gestures have high kurtosis, in other words, they have more of a bell shape (Guassian distribution).

In Fisher linear discriminant analysis, it has been shown that dynamic and hybrid gestures have a less discriminant degree (See Eq. 4.11) in the dataset. Static gestures contain a higher linear discriminant degree because of a lower frame (vertical) entropy or in other words, narrow bandwidth, as explained in the entropy analysis section. In addition, since static gestures have approximately the same spatial and temporal properties in every frame, each static gesture is more densely clustered, and hence their linear discriminations are relatively easier to estimate compared to the dynamic and hybrid gestures.

EROS analysis is a principal component analysis (PCA) scheme which applies a weighted Frobenious norm to eigenvectors and eigenvalues, which are obtained from

covariance matrix of the samples. Since EROS is based on a covariance matrix at the first stage, distinctive eigenvectors and eigenvalues do not emerge for static gestures. Hence, the EROS analysis points out this high similarity among static gestures in terms of PCA and covariance matrix analysis.

In intersection similarity analysis, it is shown that some gestures such as *Ahead*, *Back*, *Down*, *Up* and *Wave Off* have more encapsulation power than other gestures, especially static gestures. For example, *Engage* fully encapsulates the *Affirmative* static gesture in feature space.

Period variance percentage (PVP) analysis points out a high value of PVP (Mean$\mp$Std, Min/Max PVP 1.56$\mp$0.17,0/2.18) in the FDO_PT dataset, especially among static gestures. The *Engage* gesture, once again, emerges as a challenging gesture (in terms of temporal variance this time) for recognition purposes.

To summarize, the analysis of FDO_PT reveals the following: the dataset consists of high temporal variance (PVP) and dynamic gestures that mostly encapsulate the rest of the static and hybrid gestures. A PCA-based recognition algorithm, based on a covariance matrix of samples, obtains low performance, due to a high similarity in a covariance matrix of static gestures. On the other hand, the Fisher linear discriminant analysis-based algorithm, which relies on the distance between and within class distances is successful only on static gestures. It fails on dynamic and hybrid gestures. While spatial channels of dynamic gestures behave as normal distribution, on temporal channels they do not follow normal distribution because of the way the temporal channels are constructed. And entropy analysis shows that the dataset accommodates high frame entropy (large bandwidth), as was expected, low channel and class entropy for static gestures, low mutual information precision (MIP) and high noise signal ratio (NSR). High frame entropy increases intersection similarity, low class entropy, MIP and high NSR leads to a challenging task for index prediction in proposed recognition algorithm.

4.6.2 Computer Vision-based FDO Gestures (FDO_CV)

This dataset differs from FDO_PT dataset in two ways. Firstly, the mode of data acquisition; and secondly the number of users performing the gestures. Computer vision-based FDO gestures are collected via an average quality desktop webcam. Collected videos are pre-processed to extract the position of hands (x, y). Four different users performed the gestures. 18 out of over 40 FDO gestures are considered in the present work. The dataset includes over 70 samples of each gesture. Each raw gesture is represented by a stream of four coordinate data (x, y) for each hand. The coordinate

data x, y and their gradients are used as feature vectors.

$$F_{FDO\_CV} = F_{Grid} = [R_x, R_y, L_x, L_y, R'_x, R'_y, L'_x, L'_y]$$

where $[R_x, R_y, L_x, L_y,]$ corresponds to spatial grid features for the Right and Left hands and $[R'_x, R'_y, L'_x, L'_y]$ accommodates the fuzzy gradient temporal feature of the grid feature.

A detailed result of the FDO_CV dataset analysis is omitted here, due to its high similarity with the FDO_PT dataset's results which were discussed in detail in the previous section. A summary analysis of the FDO_CV dataset is presented here with supplementary figures and tables. The one important difference between the FDO_PT and FDO_CV datasets is that the FDO_CV contains more inter class and intra class similarity and more variance. This can be readily explained as follows: For the FDO_CV dataset four different users are employed for gesture performing. And these physical differences between the users are quite large. In addition, the input device of the FDO_CV dataset (webcam) only captures two dimensional coordinate data of gestures (x,y), unlike FDO_PT in which, tracker-based input device captures 3D coordinate data (x,y,z).

	$h\bar{C}$	$h\bar{H}X$	$h\bar{V}X$	$I(hH\bar{X}, hC)$	$M\bar{I}P$	$N\bar{S}R$
Mean	0.35	0.41	0.47	0.15	0.39	0.70
Std	0.23	0.18	0.09	0.07	0.34	0.19
Min	0.00	0.01	0.18	0.00	0.00	0.51
Max	0.93	0.81	0.76	0.46	0.92	1.00

Table 4.15: Summary of normalized entropy analysis of the FDO_CV dataset. $h\bar{C}$, $h\bar{H}X$, $h\bar{V}X$, $I(hH\bar{X}, hC)$, $M\bar{I}P$, $N\bar{S}R$ correspond to the normalized horizontal class, channel, frame, cross mutual information, MIP and NSR respectively.

Entropy analysis on the FDO_CV dataset reaches similar results to the FDO_PT dataset. Figure 4.48 illustrates the normalized channel, class and frame entropies for the FDO_CV dataset. Two further points worth noting are that spatial channels of static gestures have zero or low channel, class and frame entropies similar to FDO_PT. But unlike FDO_PT, temporal channels of static gestures in the FDO_CV dataset consist of higher entropies. The summary entropy table 4.15 of the FDO_CV dataset shows that the FDO_CV dataset accommodates a higher class, channel and frame entropy compared to the FDO_PT dataset. In addition, in the FDO_CV dataset, cross mutual information (figure 4.47) between classes and their samples is lesser than FDO_PV. Consequently, mutual information precision (MIP) is lower and noise signal ratio (NSR) is higher, both of which, as you recall, indicate irrelevance between samples and their associated classes.

Figure 4.47: Cross mutual information between samples (Y axis) and class models (X axis) in the FDO_CV dataset.

Unlike the FDO_PT dataset, the Chi-Squared, skewness and kurtosis (figure 4.49) reveal that the FDO_CV dataset, especially spatial channels, supports the null hypothesis, namely, it behaves more like a Guassian distribution. The reason is that more users lead to more intra class variance. In some cases, temporal channels support an alternative hypothesis (H_1), which implies an unknown, non Gaussian distribution. But, this is small compared to the FDO_PT dataset.

In figure 4.51 (top and middle), the Fisher linear discriminant and intersection similarity scheme clearly show the high inter classes similarity, because of the increased inter personal variance in the FDO_CV dataset. Fisher linear discriminant analysis once again shows discriminative power to extend among static gestures, as in FDO_PT. And inter class intersection similarity clearly shows a higher similarity between classes. Sub-event index analysis is omitted, because proposed channels do not accommodate any distinctive events. But on the other hand, period variance percentage (PVP) analysis reveals similar variance on length of samples. Figure 4.51 (bottom two) illustrates PVP analysis on FDO_CV.

Results of the PCA-based similarity, EROS, analysis are shown in figure 4.50. The figure reveals once again the higher inter class similarity. The important point here is that, not only static gestures but dynamic and hybrid gestures, unlike the FDO_PT dataset, contain high intra similarity, due to two reasons (inter person variance and lower channel numbers) as explained above.

In summary, these analyses point out that, the FDO_CV dataset contains more inter and intra complexity than the FDO_PT dataset because of the multiple users deployed and the reduced feature set (no z coordinate) in the FDO_CV dataset.

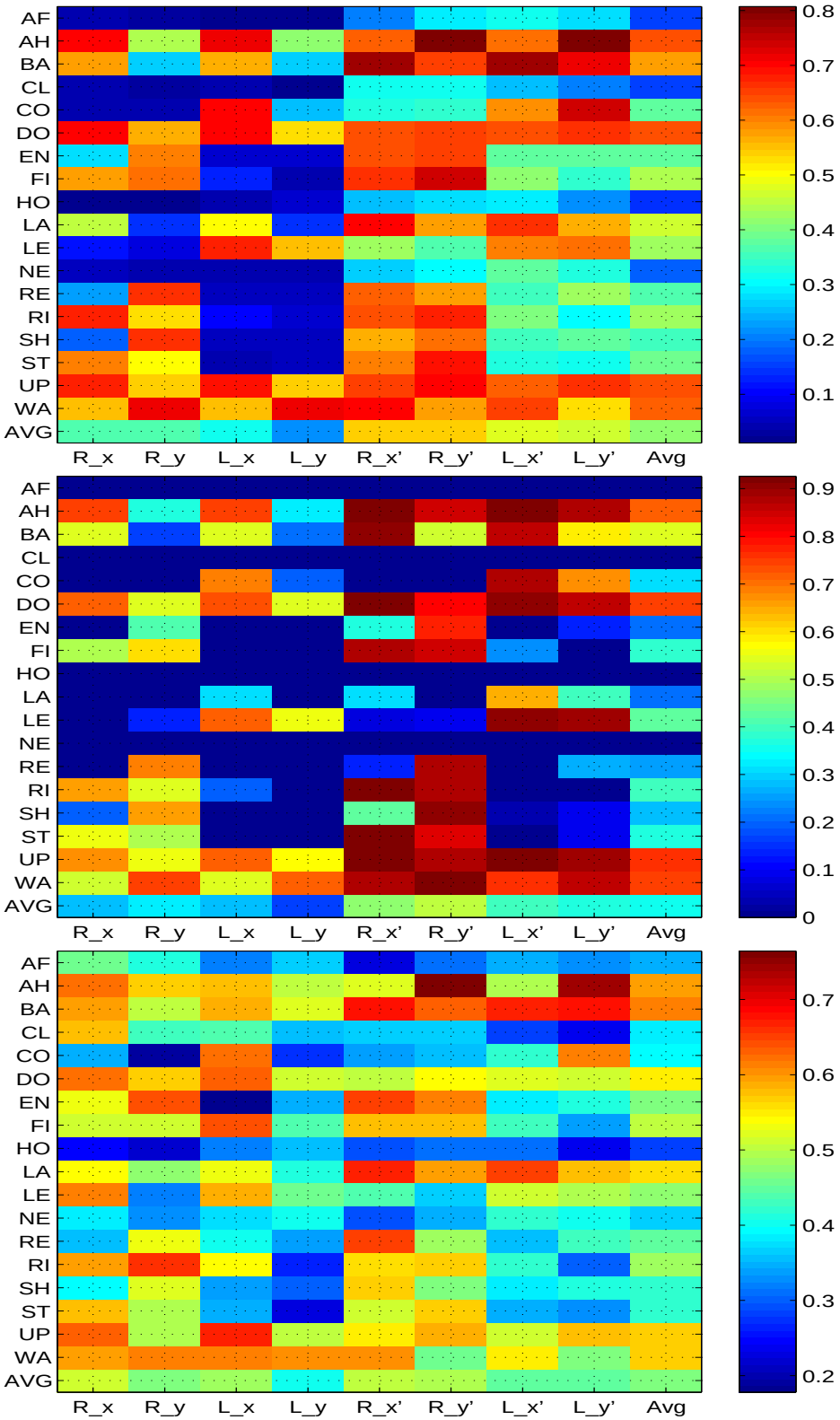


Figure 4.48: Normalized Channel (top), Class (Middle) and Frame Entropies of samples in the FDO_CV dataset. Spatial (R_x, R_y, L_x, L_y) and temporal (R'_x, R'_y, L'_x, L'_y) channels of right hand (R) left hand (L) are displayed for each class.

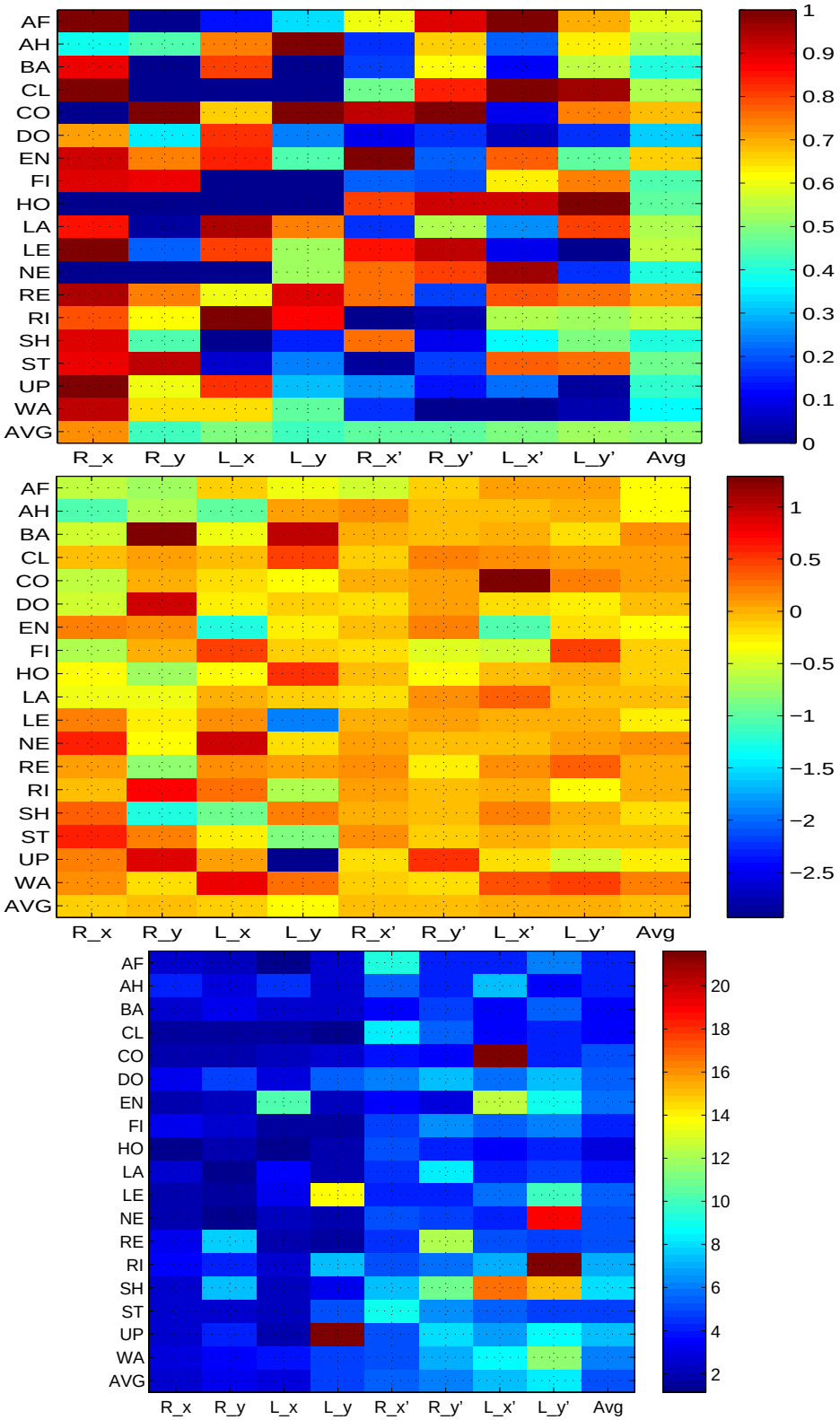


Figure 4.49: Chi2Square test ratio (top), Skewness (middle) and Kurtosis (bottom) results for the FDO_CV dataset. Similar to figure 4.39 for the FDO_PT dataset, each cell in the Chi2Square test indicates the ratio of frames accommodated in the Gaussian distribution.

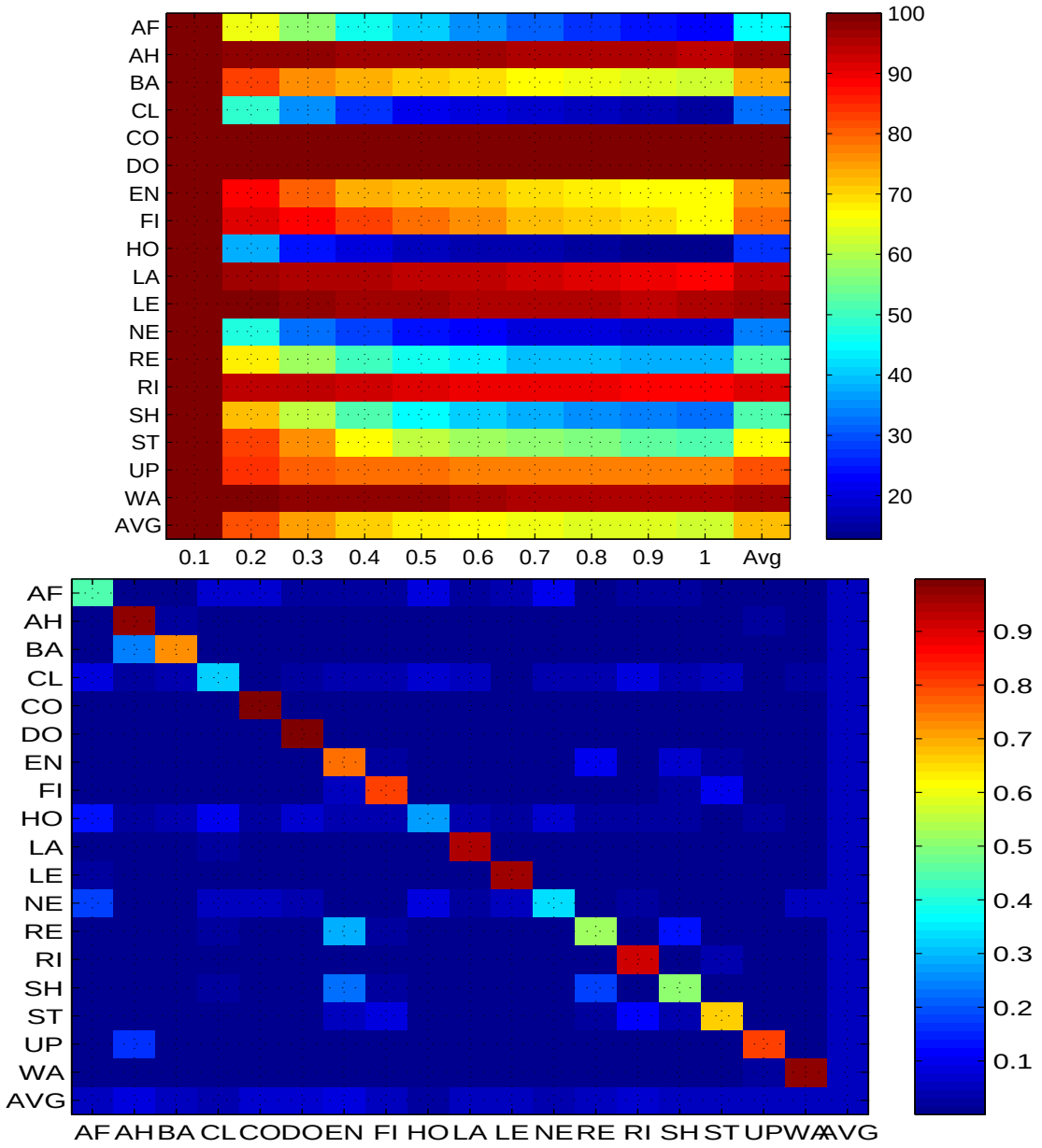


Figure 4.50: Recall/precision rate (top) and inter class cross similarity (bottom) for the FDO_CV dataset using the PCA-based similarity measurement EROS. X axis corresponds to the recall (r) value which is the number of gestures of interest retrieved in the neighbourhood of k. Precision rate ($p = r/k$) of static gestures, in particular is low, due to high similarity in covariance matrix of their samples on which EROS is based. In bottom figure the EROS-based cross similarity among static gestures can be seen.

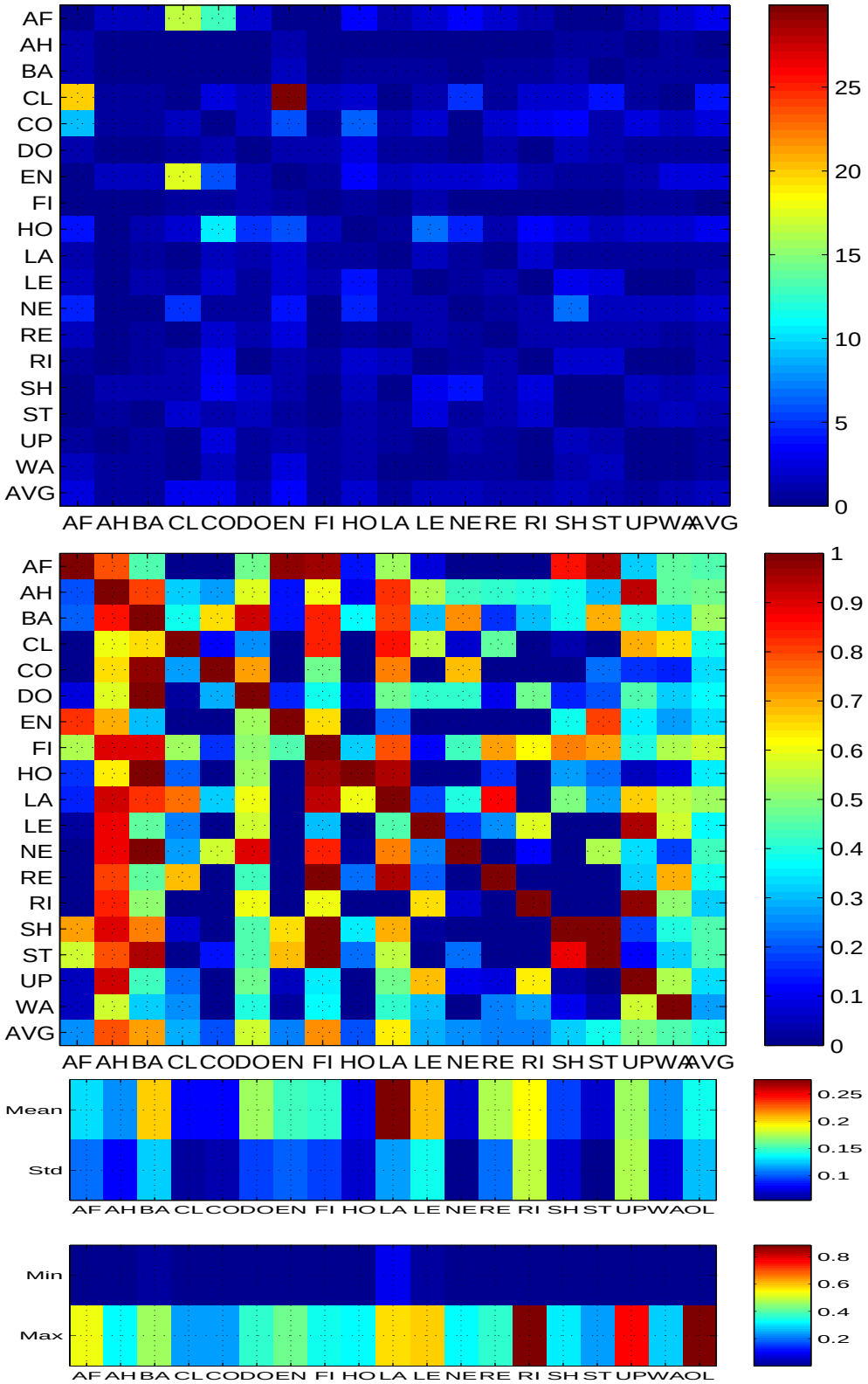


Figure 4.51: Fisher Linear Discriminant Analysis J(w,t) (top), Intersection Similarity (second from top) and Period Variance Percentage (PVP) (last bottom two) results for the FDO_CV dataset. The last column of PVP is the overall aggregation of each respective row.

4.7 Summary

Class model construction and intra/inter similarity between either class models or samples with class models have been investigated in this chapter. These two topics are illustrated using the FDO dataset and parametrised artificial dataset W_Test.

The traditional class model construction procedure is employed, which consists of data acquisition, data pre-processing, feature selection/extraction, and construction of class models. Class construction is an off-line procedure, in other words, class models are created in advance.

Two different schemes, tracker and computer vision, were implemented to collect the data of FDO gestures. The tracker-based scheme deploys the Polhemus FasTrak input device which acquires 3D Cartesian coordinates of each hand. An important note about this device is that, inspite of filter settings, it accommodates high noise. In the computer vision scheme, an average quality desktop webcam is used. The web cam obtains image sequence of gestures, from which, 2D Cartesian coordinates of the hands are extracted. According to the data collection scheme, FDO gestures are divided into two subgroups: FDO_PT and FDO_CV, point tracker and computer vision-based, respectively.

Apart from a couple of differences, these FDO_PT and FDO_CV datasets are similar to each other in their complexity. For example, as mentioned above, the data collection schemes. But the major difference is the number of users employed for data collection. For FDO_CV four users are deployed, whereas for FDO_PT only the author performed the gestures. Therefore, the FDO_CV dataset accompanies inter person variance, even though, during the data gathering process, by employing specific settings (grid-based), this inter personal variance is aimed to be minimized. Apart from that, the FDO_CV gestures aimed to simulate night/foggy scenarios in which the FDO uses light sticks to perform gestures.

For both FDO_PT and FDO_CV, specialized data collection settings are used in order to automate and make easier subsequent pre-processing, feature extraction/selection and class model construction tasks. For example, during FDO_PT gestures, users specify their physical properties (arm length, level of their shoulder) and FDO_CV, users are positioned in the middle of the image to cover the image when they stretch their arms horizontally and vertically. For FDO_PT, three raw Cartesian coordinate data from Polhemus FasTrak(x, y, z) for each hand are utilised, whereas for FDO_CV, x, y of the hands Cartesian coordinate data are used.

In the pre-processing stage, raw data is smoothed subsequently. But in the FDO_PT dataset, the integrity of the raw data is also verified, as the input device, Polhemus

FasTrak produces high noisy and out of boundary data, due to volume limitation of the device. The smoothing operation of data is based on getting the weighted average of its neighbours in a fixed window length (4). The smoothing operation is tuned to be sensitive to sub-events in data.

Pre-processed data is converted to spatial and temporal features. Spatial features deal with spatial characteristic classes, such as the whereabouts of classes in raw data space, whereas temporal properties of raw data, such as direction of gradient, velocity, angular velocity, acceleration, are encompassed with temporal features.

For the FDO_PT and FDO_CV datasets, various feature sets (raw, angular and grid) were considered. Since, grid-based feature set F_{Grid} achieved better performance over the segmented FDO_PT dataset with the proposed recognition algorithm, the grid-based feature set F_{Grid} is used main feature set for both the FDO_PT and FDO_CV dataset. The main idea behind of the grid-based spatial feature set is that, raw data is normalized according to the physical limitation of the users. For normalization, it is assumed that users are fitted into a fixed $15 \times 15 \times 15$ unit 3D (or 15×15 2D, in case of FDO_CV dataset), grid cube such that when the user stretches his arm horizontally or vertically upwards at the shoulder level, the user's hands touch the boundary of the grid cube on the top, left and right edge of the cube. Special settings of FDO_PT and FDO_CV data collection aim to incorporate grid-based feature computing.

For temporal features, the fuzzy gradient features (x') of a spatial feature (x) are used. The fuzzy gradient feature accommodates the direction of consecutive spatial grid features such as whether it is increasing (1), decreasing (-1) or between them [-1,1] in spatial grid feature space. Therefore, the feature vector for FDO_PT and FDO_CV are as follows:

$$\begin{aligned} F_{FDO\_PT} &= F_{Grid} = [R_x, R_y, R_z, L_x, L_y, L_z, R'_x, R'_y, R'_z, L'_x, L'_y, L'_z] \\ F_{FDO\_CV} &= F_{Grid} = [R_x, R_y, L_x, L_y, R'_x, R'_y, L'_x, L'_y] \end{aligned}$$

where $[R_x, R_y, R_z, L_x, L_y, L_z]$ corresponds to spatial grid features for the Right and Left hands and $[R'_x, R'_y, R'_z, L'_x, L'_y, L'_z]$ accommodates the fuzzy gradient temporal feature of the grid feature.

Class models are the summary representations of the training cycles. For construction of a class model, template-based modelling is preferred. Template-based modelling represents the trajectory of classes in forms of features with summary, compact statistical parameters. Channel construction is based on estimating the parameters which represent best the underlying statistical distribution of training data at each time point

of the channel. In this study it is assumed that features are independent of each other and training data at a time index in a channel obeys normal statistical distribution. Therefore, the channel construction procedure is based on estimating the parameters of the statistical mean μ and standard deviation σ to represent the training data at each time index in the channels. Class construction procedure includes: Feature Vector Analysis (F_{Grid}) period estimation (L_i), stretching/compression and sub-event alignment and the statistical estimation of model parameters.

For construction of a class, for instance, firstly, raw training cycles are transformed into the spatial grid feature ($rR_z \rightarrow R_z$). Since training cycles have different lengths, a common period (L_i) is estimated for the class. Later, training cycles are stretched, compressed and aligned to have the same length of L_i. Temporal fuzzy gradient cycles are estimated from the aligned spatial grid cycles. Finally, the statistical Gaussian mean and standard deviation of the aligned spatial and temporal cycles are calculated for each index.

Training cycles, due to temporal variances, accommodate high variance in their lengths and location of sub-events. Therefore, before statistical parameters are estimated, training cycles are organized, to maximize the common properties of training data at each time index. Therefore, stretching, compressing and sub-event alignment is applied over channels before statistical parameter estimation. These operations are vital for obtaining meaningful and robust statistical parameter estimation. Compression or stretching operations are channel-based. The stretching operation is applied by uniform linear interpolation. In other words, new interpolated data points are uniformly inserted into the channels. The value of new data points are the average value of the adjusting data points. Similarly, the compression operation is done uniformly. Instead of interpolating new data, two data points are merged into one with the average value of them being used. On the other hand, sub-events are organized in a supervised manner for each channel and class by employing linear interpolation or dropping some points by averaging consecutive two points after detecting sub-events with ad-hoc methods.

Having constructed the class models, in the second half of the chapter, a comprehensive complexity and similarity analyses of class models were presented. For this purpose, several existing and new advanced techniques, which are mostly proposed for statistical classes were adapted for temporal classes. These techniques focus on different aspects of class models such as channel complexities, verifying assumed statistical properties, inter class similarities, similarity between samples and their associated and other class models, and principal feature analysis.

For example, the entropy-based information complexity technique is used to analyse

the similarity between samples and class models. It also addresses the channel and feature complexity or variance which directly affects the proposed recognition algorithm. Chi-Square, skewness and kurtosis statistical analyses are applied to analyse the fitness and robustness of the parameters (μ and σ) of the assumed underlying statistical distribution (Gaussian distribution). Fisher linear discriminant analysis addresses the inter similarity between class samples. It utilises the *within* and *between* class ratio to establish a similarity measurement. The principal component analysis-based *EROS* also implements a similarity measurement technique, which utilises the extended version of a Frobenious norm (Euclidean distance) over eigenvectors and eigenvalues of covariance matrix of class samples by implementing a k-nearest neighbourhood recall/precision scheme. Apart from these, a novel approach, intersection volume between classes' models was also used as another disparity measurement. As part of temporal complexity analysis, variance in length of class samples and sub-events indices was also discussed.

These techniques were shown in action over the parametrised artificial W_Test dataset, before been applied comprehensively to the FDO_PT and FDO_CV datasets. These analyses over FDO_PT dataset reveal the following: Entropy analysis shows that the dataset contains high frame entropy (large bandwidth), low channel and class entropies for static gestures, low mutual information precision (MIP) and high noise signal ratio (NSR). While high frame entropy increases the intersection probability between class models, low class entropy, MIP and high NSR cause wrong index predictions in proposed recognition algorithm. The fitness analysis of statistical parameters concludes that while spatial channels of dynamic gestures behave as a normal distribution, temporal channels do not show any normal distribution because of the way the temporal channels are constructed. PCA-based *EROS* analysis, which is based on a covariance matrix of samples, obtains low performance due to high similarity in its covariance matrix of static gestures. On the other hand, the Fisher linear discriminant analysis-based algorithm which relies on the distance between and within class distances is successful only on static gestures. It fails on dynamic and hybrid gestures. FDO_PT consists of high temporal variance (PVP) in terms of length of samples, and intersection volume analysis reveals that dynamic gestures mostly encapsulate the rest of static and hybrid gestures.

The analysis of the FDO_CV dataset obtains similar results to the FDO_PT dataset. The one important difference between the FDO_PT and FDO_CV datasets is that FDO_CV accommodates more inter class and intra class similarity and variance due to multiple users (four users performs the gestures in FDO_CV) and the reduced data dimensionality (x, y).

Chapter 5

Gesture Recognition Algorithm

" ... we try to interpret the present, understand the past,
and perhaps predict the future, when very little is crystal clear. " [107]

In this chapter, a detailed analysis of the proposed recognition algorithm (Recognition Machine, *RM*) is presented. Remind that, the primary aim of this study is to develop an on-line recognition system to recognize FDO's gestures as part of a virtual training system. A more formal definition presented in the second chapter is as follows:

Given class models C comprising H number of channels and feature set F with period of L, and incremental test data in the input band B, develop a system or recognition machine (RM) to recognize the classes R_C to which each part of B belongs in an on-line manner.

$$R_C \quad = \quad RM(B|C,L,H,F)$$

In order to recall these notations, readers are advised to refer to the formal problem definition, notation and terms sections (2.2 and 2.3) in the second chapter.

The recognition machine is implemented according to the classical pattern recognition framework [87]. The recognition machine (RM) has nine interacting components each of which corresponds to a task in the framework such as data acquisition, data pre-processing, feature extraction, class modelling and temporal recognition algorithm.

The recognition machine (RM) conceptually is an on-line template matching technique. The main idea behind the recognition algorithm is to exploit the sequential consistency of the input frames according to class models by using a dynamic programming paradigm and the Markovian process. Sequential consistency or so-called *Score* (S) addresses the similarity between the incremental input data and the class models. *Scores* employ similarity factors (Θ) for each class with an on-line sequential decision process which involves some predictions. In other words, the degree of

similarity (awards) and dissimilarity (punishment) are used to score the gestures. In this sense, the technique involves reinforcement learning. The prediction process is a probabilistic estimation of the index of frames (N) in each class model (C) which are spatially closest to the input frame (X), given the most recently predicted frame index (V). Having estimated *Score* (S), some heuristics are employed to control the on-line recognition declaration in the final stages.

RM addresses the general gesture recognition issues such as real-time recognition of dynamic and static gestures, the start/end of gestures in continuous streams and inter-intra personal temporal and spatial variance. The first two issues and temporal variance are addressed using the algorithm. For overcoming the spatial invariance, class models are constructed using independent features. RM is intuitively able to detect the start/end of gestures (automatic segmentation) in continuous streams as part of the index prediction scheme.

In literature, several techniques have been described to address real-time, dynamic gesture recognition problems with various degrees of success. These attempts include some of the following major techniques: neural networks (NN), hidden Markov model (HMM), dynamic programming (dynamic time warping, (DTW)). Even though, all of these techniques approach the problem from different the perspective, at the core, similar to the Recognition Machine (RM), they aim to exploit the underlying temporal-spatial dynamo.

As elaborated upon in the literature review chapter, these techniques have their own disadvantages and advantages. RM aims to take into account these advantages and disadvantages. Especially, RM has similar properties to HMM. RM can be reduced to a special version of HMM. But, RM addresses several traditional issues of HMM such as evaluation, decoding, training, and topology for gesture recognition domain. Unlike HMM, employed heuristics in RM during and after the recognition phase provide a more controlled recognition declaration. Moreover, HMM tends to achieve more reliable recognition results with a smaller number of states, which is not useful for training feedback in case of wrong gesture performance. But RM employs ergodic topology (which is biased from left to right) in which the number of states are sufficiently large enough to provide meaningful feedback in case of wrong gesture performance. In addition, unlike HMM, RM uses heuristics to spot and reject undefined movements (transition data) from one gesture to another [152]. In HMM, these undefined movement are modelled from another HMM, which may not represent efficiently all the movements [153, 73, 148].

Remind that, in the conclusion of the literature review, it was noted that, hybrid approaches, which utilise the best aspects of existing techniques, are have been gaining

more attention in recent years. As mentioned above, RM is a modular, component-based system. Therefore, an efficient and specific technique can be implemented to carry out the task for each component. In order to show the modularity of RM and to utilise the advantages of various techniques, a multilayer perceptron network, which aims to approximate the prediction function, is alternatively introduced for the prediction process (component) in this chapter.

In the remainder of the chapter, the proposed algorithm will be elaborated in detail. Discussion starts with the intuition behind the algorithm. The intuition will be explained with a simple fictitious example. Having explained the main idea, a formal definition of proposed algorithm will follow. Detailed discussion of the phases of the algorithm occupy the next sections. Then, the time and space complexity of the algorithms are discussed. Prior to concluding the chapter, an analogy of the proposed algorithm with other popular algorithm is considered. The analogy focuses on the Hidden Markov Model and Dynamic Time Warping techniques.

5.1 Foundations of the Proposed Algorithm

Prior to elaborating the recognition algorithm in detail, let us have a look at the following example, which paves the way for the proposed algorithm.

Given a class template, C_i, and incremental test data D, the problem of interest is to estimate whether D belongs to class C_i. As figure 5.1a depicts, test data D, is akin to the class template, C_i, except that a small initial part of C_i is shifted. This can be interpreted to show that the test data does not start immediately from its starting point but a bit later on. Despite this difference, both gestures (signals) convey the same meaning or information. Bear in mind that the test data is incrementally available. In other words, at time t, only the data at $d(t)$ and previous data are only available.

Figure 5.1b-d demonstrate the main idea behind the proposed algorithm. Let us assume that at time $t = 1$, the first test frame $d1$ matches (spatially closest) the frame of C_i at the index n and k (Figure 5.1b). And if it is assumed that $d1$ is about beginning of the class C_i, then n can be taken as the first predicted frame to which $d1$ most closely matches both spatially and temporally. For the time being, simply, the matching operation will be interpreted as finding the spatially closest frames along the template C_t. Then, at time $t = 2$, the subsequent test frame $d2$, matches frame of C_i at the index $n + a$ and $k - z$, as the figure 5.1c shows. Since $d2$ is succeeding $d1$, at least one of the matched frames of C_i ($n + a$ or $k - z$) is intuitively expected to be succeeding at the latest predicted frame (n), if D belongs to class C_i. Therefore, $n + a$

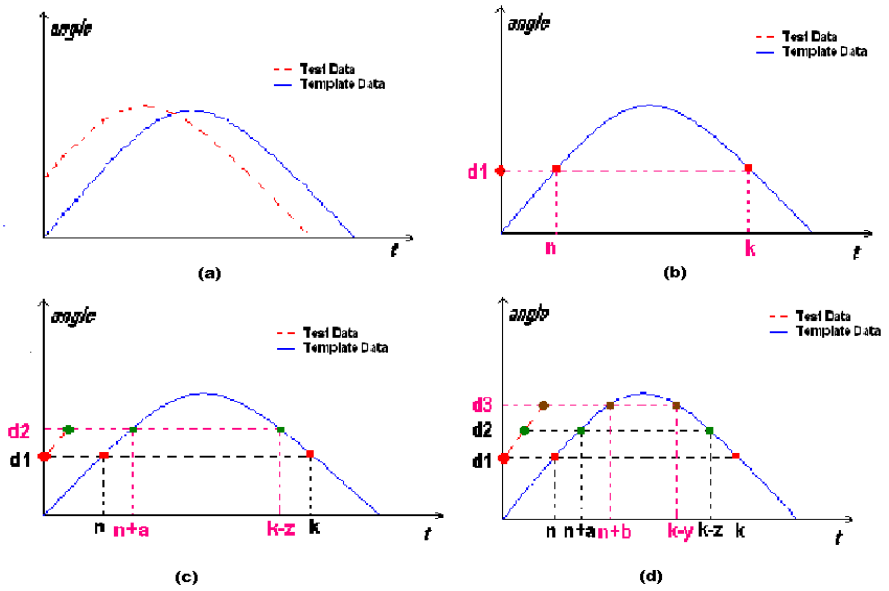


Figure 5.1: An example depicting the proposed technique

is predicted as the second predicted frame. Similarly, at time $t = 3$, among matched frames $(n + b, k - y)$ of subsequent test frame $d3$, $(n + b)$ is predicted as the result of the latest predicted frame $(n + a)$.

The main idea behind this example is that the distance between consecutive predicted frames indicates the similarities. In other words, the similarity of the two signals can be expressed in terms of a function of distances $(\Delta = a, b)$ of predicted frames $(n, n+a, n+b)$. The lesser the distance between predicted frames $(\Delta = a, b)$, the more similarity between test D and class C_i. Therefore, the distance function $\Psi(\Delta)$, is a major component of the similarity factor. Consequently, the cumulative product of all similarity factors accumulates the total similarity between D and C_i, which is called *Score* (S_i) in the remainder of the thesis. For the time being, if the distance function is assumed to be the reciprocal of square distance, *Score* (S) can be estimated as follows:

$$\begin{aligned} \Delta_{i,t} &= |N_t - V_t| = |V_{t+1} - V_t| \\ \Psi(\Delta) &= \frac{1}{\Delta^2} \\ \Theta &= \Psi(\Delta); \\ S &= \prod^{T} \Theta \end{aligned}$$

where Θ, V_t and N_t correspond to the similarity factor, current predicted index $(n+a)$

and next predicted index $(n+b)$, respectively. Note that the current index in the next time step $(t+1)$ actually corresponds to N_t, $(V_{t+1} = N_t)$.

Let us consider the example again. The number of the matched indices can be reduced by considering the gradient of d_t and C_i. For example, at time $t = 2$, since the gradient of test data $(d_2 - d_1)$ is increasing, the frame $k - z$ can be eliminated among the matched set$((n+a,\ k-z))$ because at $k - z$ frame, C_t tends to have a decreasing gradient. Approximately, half of the matched points can be reduced by using a gradient feature. Therefore, adding features (for example gradient) to the algorithm improves prediction accuracy by reducing the matched frame set.

Note that the a, b, y, z, predicted frame distances, account for temporal variance. One of the most important advantages of the algorithm is being able to overcome the temporal (intra and inter personal variance and noise) issues by the means of predicting the possible matched frames monotonically, incrementally and partially (time warping) rather than matching two signals frame by frame and in an off-line manner.

The approach also suggests an automatic way for resolving the issue of the *start/end* of a gesture. Monotonic, incremental and partial prediction enables us to dismiss this issue since the algorithm is not based on a classical matching method (such as in dynamic time warping, DTW), which requires start/end frames for matching.

The following oversimplified example explains the scheme more comprehensively. Given the example of (C, P, H, F, B):

$$
\begin{aligned}
C &= (C_1, C_2, C_3) \\
P &= (5, 9, 7) \\
C_1 &= H_{1,1} = \{-1, -1, 1, 1, 0\} \\
C_2 &= H_{2,1} = \{1, 1, -1, -1, 0, 0, 1, 1, 1\} \\
C_3 &= H_{3,1} = \{-1, 1, 1, 1, 0, 0, 0\} \\
f &\in (-1, 1, 0) \\
B &= \{-1, 1, 1, 0\}^+
\end{aligned}
$$

where the problem of interest is to estimate the class to which B belongs. $...^+$ operators guarantee that B contains at least one element. The example consists of three classes each of which has a different period. The feature vector f is one dimensional, discrete and takes one of the following values $(-1, 1, 0)$. Due to the characteristics of the feature alphabet, the templates are analogous with gradient templates (decreasing, constant and increasing).

In the example, B is shifted and is a repeated version of class C_1, therefore, the proposed scheme should result in favour of C_1. Table 5.1 shows the steps of manually

running the proposed scheme. In table 5.1, V_i is the predicted frame for the class C_i; $\Psi(\Delta)$ is the result of the distance function defined above and Δ_i is the distance between the current predicted frame ($V_{i,t}$) and the previous predicted frame ($-V_{i,t-1}$), ($\Delta_i = V_{i,t} - V_{i,t-1}$) for the class C_i. S_i corresponds to the score of the class C_i. The magnitude of S_i indicates the similarity degree between B and C_i at each time. From the table it is clear that in a couple of steps, C_2 is certainly not the recognized class. On the other hand, recognition of test data B, as class C_1 is a matter of time. If the algorithm is run long enough, class C_1 emerges as the recognized class against class C_3. Note that since class C_1 and class C_3 are highly similar to each other and their periods are small, an earlier recognition does not emerge.

t	B	V_1	V_2	V_3	Δ_1	Δ_2	Δ_3	$\Psi(\Delta_1)$	$\Psi(\Delta_2)$	$\Psi(\Delta_3)$	S_1	S_2	S_3
0	-	0	0	0	1	1	1	1.00	1.00	1.00	1.00	1.00	1.00
1	-1	1	3	1	1	3	1	1.00	0.11	1.00	1.0000	0.1111	1.0000
2	1	3	7	2	2	4	1	0.25	0.06	1.00	0.2500	0.0069	1.0000
3	1	4	8	3	1	1	1	1.00	1.00	1.00	0.2500	0.0069	1.0000
4	0	5	5	5	1	6	2	1.00	0.03	0.25	0.2500	0.0001	0.2500
5	-1	1	3	1	1	6	3	1.00	0.03	0.11	0.2500	0.0000	0.0278
6	1	3	7	2	2	4	1	0.25	0.06	1.00	0.0625	0.0000	0.0278
7	1	4	8	3	1	1	1	1.00	1.00	1.00	0.0625	0.0000	0.0278
8	0	5	5	5	1	6	2	1.00	0.03	0.25	0.0625	0.0000	0.0069
9	-1	1	3	1	1	6	3	1.00	0.03	0.11	0.0625	0.0000	0.0008
10	1	3	7	2	2	4	1	0.25	0.06	1.00	0.0156	0.0000	0.0008
11	1	4	8	3	1	1	1	1.00	1.00	1.00	0.0156	0.0000	0.0008
12	0	5	5	5	1	6	2	1.00	0.03	0.25	0.0156	0.0000	0.0002
13	-1	1	3	1	1	6	3	1.00	0.03	0.11	0.0156	0.0000	0.0000
14	1	3	7	2	2	4	1	0.25	0.06	1.00	0.0039	0.0000	0.0000
15	1	4	8	3	1	1	1	1.00	1.00	1.00	0.0039	0.0000	0.0000
16	0	5	5	5	1	6	2	1.00	0.03	0.25	0.0039	0.0000	0.0000
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.

Table 5.1: Steps of manual simulation of basic recognition algorithm over a simplified example. Band B consists of infinitely repeated instances of class C_1. In the first couple of steps, C_2 is eliminated among candidate classes as its score rapidly decreases. After $t = 8$, it is emerged that B belongs to class C_1 rather than C_3.

Although the example describes in some detail the gist of the proposed technique, it does not yet resolve some of the following issues: continuous data, multiple channels, and the conditions for announcing recognition.

So far, as was shown in the last example, templates and features are assumed discrete in which the matching operation is not greater than finding the frames in the template which is exactly equal to the test degree. In other words, the matching operation is binary discrete, namely, either equal (true) or not (false) ($\{0, 1\}$). But the discrete feature alphabet is not a realistic representation in most domains. Recall from the analysing and modelling chapter that statistical template-based representation is

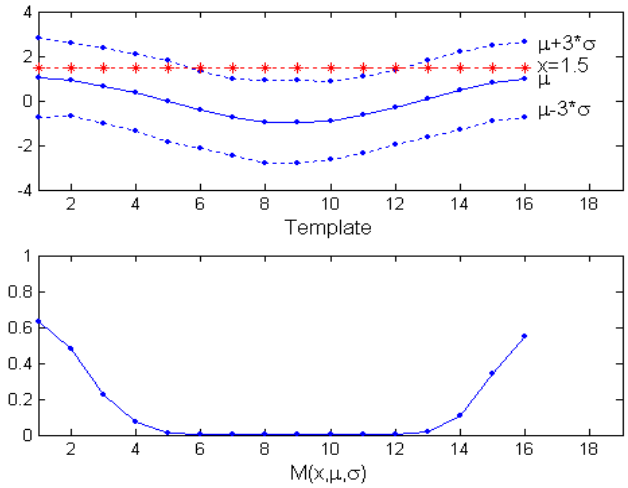


Figure 5.2: Degree of Membership Curve of a Channel

used in this study. In statistical template-based representation, each frame is represented with some distribution parameters. Therefore, the matching operation has to be modified to address the continuous data and statistical template presentation.

In the case of the continuous data and statistical template presentation, the modified matching operation will not be a binary discrete value anymore. It is a value representing degree of membership ($[0, 1]$) and corresponds to measurement (probability) of how much a test frame belongs to each frame in the template. Since statistical template presentation is based on some distribution parameters (such as μ and σ in Gaussian distribution), the estimation of degree of membership is trivial. Figure 5.2 illustrates the degree of membership curve of a channel, in which frames have properties of Gaussian distribution, given a test frame/point (x). Estimation of the'membership curve will be elaborated in detail in the following sections.

In a discrete case, the prediction operation does not extend beyond that from selecting the closest matched frames to the latest predicted frame. Note that in the first example, at time $t = 2$, $n+a$ is predicted among the matched set ($\{n+a, k-z\}$), since $n+a$ is the closest matched frame to the latest predicted frame n. But in a continuous case, the matched set is substituted for the degree of membership curve. Therefore, the prediction operation has to be modified. On careful analysis, we observed that the degree of membership curve around the possible frames is maximized. Therefore, the prediction operation, in the continuous case, is responsible for finding the local maximum frame which is closest to the latest predicted frame. Consequently, the degree of membership value of the predicted frame (M_N) can be used as a similarity

factor. Accordingly, the similarity factor can be re-defined as follows:

$$M_{N_i}(x, \mu_{i,N}, \sigma_{i,N}) = e^{-\frac{(x-\mu_{i,N})^2}{2\sigma_{i,N}^2}} \quad (5.1)$$

$$\Theta_i = M_{N_i}\Psi(\Delta_i) = \frac{M_{N_i}}{\Delta_i^2} \quad (5.2)$$

where x is the test data, N is the predicted index, $\mu_{i,N}$ and $\sigma_{i,N}$ are the mean and standard deviation of the template (class) C_i at the predicted index (N) and M_{N_i} is the degree of membership value of x to class model C_i at index N. Functional dependency of M_{N_i} on x, $\mu_{i,N}$ and $\sigma_{i,N}$ is suppressed from now on for brevity. Note that, $\mu_{i,N}$ and $\sigma_{i,N}$ normal distribution parameters of each index in channels (templates) are estimated out of training samples (x), as it is explained in Channel and Class Model Construction section 4.4. The idea behind the estimation of degree of membership values M_{N_i} for given x and class model parameters ($\mu_{i,N}$ and $\sigma_{i,N}$) will be elaborated in section 5.2.2.

So far, as can be seen in equation 5.1, the reciprocal of square distance (Δ) is assumed as being the distance function (Ψ), which is a naive approximation. The distance function (Ψ), under normal conditions, actually behaves as a normal, Gaussian membership functions ($e^{-\frac{(\Delta-\mu)^2}{2\sigma^2}}$), where the expected (mean, μ) transition distance between successive distance and standard deviation (σ) is 1. Therefore, the distance function can be rewritten as follows:

$$\Delta_{i,t} = |N_t - V_t| = |V_{t+1} - V_t|$$
$$\Psi(\Delta) = e^{-\frac{(\Delta-1)^2}{2}}$$

In previous examples, classes have consisted of only one channel, so without hesitation, the template term can be used interchangeably with channel. In the case of multiple channels, the proposed algorithm is applied over all channels to obtain their degree of membership curves for the given test frame. Then, these channel degree of membership curves are aggregated using the product aggregation operator to obtain a cumulative final degree of membership curve for the class.

Another important point is the condition and/or time when the algorithm can be sure whether a gesture is recognized or not. In the last example, it was said that the algorithm had to be run long enough to declare as a recognition. Here, obviously,

questions may arise. How long does the algorithm have to be run to get a result? What is the optimum time and/or conditions to obtain a reliable recognition? Should the algorithm give the immediate result as soon as a maximum score is emerged or wait until a score is dominant to others? And/or some other conditions such as temporal properties of classes should be considered when deciding whether a gesture is recognized or not. By temporal properties it is meant that, for example, if some parts (milestones) of classes are sequentially observed. Actually the algorithm deploys the latter approach, using temporal properties, with the assistance of *Scores* to declare a recognition.

The outlined scheme has a computational issue as the length of the test data T increases. Since the distance function (Ψ) is in the range of $[0, 1]$, repeated multiplication of Ψ, namely, score (S) approaches to zero with an exponential trend. This issue forces the precision boundaries of the computer. Hence, the proposed technique has to be modified to handle that issue. Simply, taking the *log* of score in the equation, 5.1, eliminates that issue, since the logarithm function transforms the multiplication operator ($\prod$) to summation ($\sum$).

Thus, the revised algorithmic steps are shown below:

$$\begin{aligned}
\Delta_{i,t} &= |N_t - V_t| = |V_{t+1} - V_t| \\
\Psi(\Delta_{i,t}) &= e^{-\frac{(\Delta_{i,t}-1)^2}{2}} \\
\Theta_{i,t} &= M_N \Psi(\Delta_{i,t}) = M_N e^{-\frac{(\Delta_{i,t}-1)^2}{2}} \\
log(S_i) &= \sum_{t=1}^{T} log(\Theta_{i,t})
\end{aligned}$$

Note that, in the remainder of the thesis, the summation index t is suppressed where there is no ambiguity.

5.2 The Proposed Algorithm In Detail

Having laid the fundamentals of the algorithm and surrounding issues, the proposed algorithm can be elaborated in detail now. Figure 5.3 and the pseudo source code in 5.4 illustrate the main components and their intra relationships. While figure 5.3 represents the general framework and the flow diagram of components, the pseudo source code in 5.4 denotes skeleton components which are covered in detail in later sections. Although, for example, the implementation of the language component in the flow diagram is quite simple, it is not covered in detail in this study. It is assumed

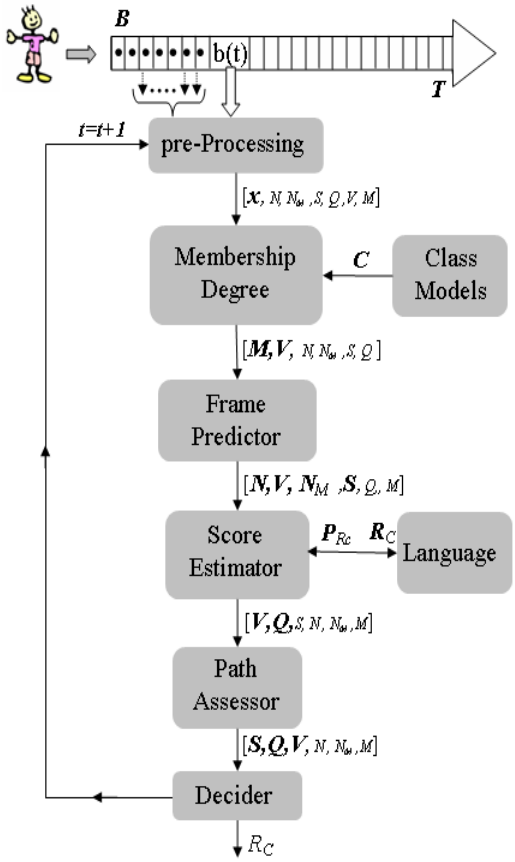


Figure 5.3: Components of a recognition machine (C=Classes; d=Current input point; M_d=Membership degrees; V=Current Matched Indices/Points; N=Next Indices ($N = V_{t+1}$); N_{M_d}=Membership Degree of Next Indices S=Scores; Q=Path Land marks ; R=Recognized Class). Bold, italic variables going into components are the main inputs of that component.

that in the case of continuous recognition, prior probabilities of all classes are the same, so can lead to a uniform distribution of classes over the band (B) by the language.

The recognition algorithm conceptually is an on-line template matching technique. The main idea behind the recognition algorithm is to exploit the sequential consistency of the input frames according to class models by using a dynamic programming paradigm and the Markovian process. Sequential consistency or so-called *Score* (S) addresses the similarity between the incremental input data and the class models. *Scores* employ similarity factors (Θ) for each class with an on-line sequential decision process which involves some predictions. The prediction process is a probabilistic estimation of the index of frames (N) in each class (C) which are spatially closest to the input frame (X), given the most recently predicted frame index (V).

Outline of the recognition machine is as follows: The recognition machine (RM)

```
RM(C, P, H, F, B)
   begin
     V=zeros(1,M); % Latest Predicted Frames
     N=zeros(1,M); % Current Predicted Frames
     S=zeros(1,M); % Scores
     Q=zeros(M,4); % Class Path
     RC=-1;         % Recogonized Class
     t=0;
     while (t<T)
        begin
          d=B(t);
          x=pre-Processing(d);    % Smoothing, tranformation, feature extraction
          M=MembershipDegree(x,C);         % Likelihood Function
          [N,M_n]=framePredictor(M,V);     % Transfer Function
          [S,V]=scoreEstimator(N,V,M_n,S);
          [V,S,Q]=PathAssessor(V,S,Q);
          [RC,V,S,Q]=Decider(V,S,Q);
          t=t+1;
        end
     end
```

Figure 5.4: Pseudo source code of the proposed algorithm/recognition machine covered in the study

has nine components each of which partially interacts with each other. RM is fed by an input band B, of which its properties are defined in the formal problem statement section in the second chapter. Subsequent to acquiring data from the band ($b(t)$) incrementally at each discrete time t, data is pre-processed. Pre-processing involves on-line smoothing, transformation and feature extraction processes. Pre-processed data (x), then, is matched with all the channels of classes to obtain channel degree of membership curves ($M_{i,j}$). In each class, channel degree of membership curves are aggregated to obtain a final degree of membership curve (M_i) which represents the membership degree of x to a class (C_i). Aggregation operation is a product operator which is based on the fact that channels are independent among themselves. Having estimated the membership degrees (M) in the *membership degree* component, given the latest predicted frame (V), the next predicted frame (N) is estimated in the following component by using some dynamic programming techniques. Then in the next component *(score estimator)*, scores (S) are estimated based on similarity factors which generally consist of a distance function (Ψ) and membership degree of the predicted next frame (M_N). The distance function is a type of radial basis function. In the final two components, conditions are checked to see whether or not a recognition has emerged. These conditions include the amplitude of scores S and observed paths or frames of classes.

5.2.1 Pre-Processing

The recognition machine is fed by a ϑ dimensional band (B) of which its properties are elaborated upon in the problem definition chapter. Typically, content of B is obtained from input devices. The pre-processing component carries out smoothing, transformation and feature extraction tasks in that order.

Data in the band is available incrementally, therefore an on-line smoothing technique is implemented. For that purpose, some historical frames in a fixed length window are utilised for robust smoothing. But note that these historical points are used only for the smoothing process. They are not used for feature extraction as the algorithm proceeds for an on-line recognition.

In the final phase of the pre-processing, the smoothed raw data is transferred into features. Please refer to chapter four, Analysis and Modelling, for a detailed discussion on pre-processing, feature selection and extraction.

5.2.2 Membership Degree

This component, as its name indicates, is responsible for estimating the likelihood, namely, the degree of membership curve of a given frame x for all the defined classes C.

$$M_i \quad = \quad P(x|C_i) = \chi_{j=1}^{\vartheta} M_{i,j} = \chi_{j=1}^{\vartheta} P(x_j|H_{i,j}) \tag{5.3}$$

where χ corresponds to a channel aggregation operator.

A membership degree component consists of two sub phases, intra and inter membership degree estimation. Intra membership degree estimation ($P(x_j|H_{i,j})$ is the likelihood of a particular dimension of input frame (x_j) being linked to the corresponding class channel ($H_{i,j}$). The second phase is the aggregation of the intra channel degree of membership curve to obtain the ultimate degree of membership curve which represents the class. Intra channel degree of memberships curves $M_{i,j}$ indicate how much the feature of the given frame (x_j) belongs to the particular channel $P(x_j|H_{i,j})$. Figure 5.2 illustrates the degree of membership curve of a frame and channel. Having estimated the intra channel membership degrees, the joint degree of membership curve (M_i) for the frame x and the class C_i is estimated by deploying an aggregation operator χ.

The way in which the channel is constructed or represented, is the essential factor when estimating the degree of membership curve. For example, the in case of spatial

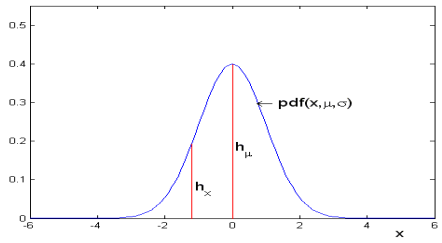


Figure 5.5: Estimation of Membership Degree under a normal distribution. It is also known as "Gaussian curve membership function" in literature. Membership degree is the ratio of pdf of x and μ frame ($M_{i,j} = \frac{pdf(x,\mu,\sigma)}{pdf(\mu,\mu,\sigma)} = \frac{h_x}{h_\mu}$) for the normal distribution $N(\mu, \sigma)$.

channels, it is assumed that the underlying distribution at each time index is a Gaussian distribution and channels are independent of each other, and the statistical mean and standard derivation parameters of the Gaussian distributions are used to estimate membership degrees. Therefore, in the case of Gaussian distribution, the intra membership degrees ($M_{i,j}$) of the given frame feature $x_{j,t}$ to index t in the channel $H_{i,j}$ with standard mean and deviation ($\mu_{i,j,t}, \sigma_{i,j,t}$) are calculated as follows:

$$\begin{aligned} M_{i,j} &= \frac{p(x_{j,t}, \mu_{i,j,t}, \sigma_{i,j,t})}{p(\mu_{i,j,t}, \mu_{i,j,t}, \sigma_{i,j,t})} \\ &= \frac{h_x}{h_{\mu_{i,j,t}}} \\ &= e^{-\frac{(x_{j,t}-\mu_{i,j,t})^2}{2\sigma^2_{i,j,t}}} \end{aligned}$$

In literature, this estimation is also known as "Gaussian curve membership function" and figures 5.2 and 5.5 show this.

Similar to intra channel membership degree estimation, the inter channel estimation (aggregation operators) is also domain-based. In fuzzy logic literature, various aggregation operators (product, geometric mean or min) are defined for a variety of problems [58]. In the current study, a product aggregation operator is implemented. Actually, the product operator is a direct consequence of the independent channel assumption in the domain of probability theory.

$$M_i = \sqrt[\vartheta]{\prod_{j=1}^{\vartheta} M_{i,j}} \quad where \quad 0 < i \leq \varpi \tag{5.4}$$

Note that the length of intra and inter channel degree of membership curves are periods of the classes (L_i)

5.2.2.1 Non-Alinement of Sub Events

Because of temporal variances, sub events can occur at various indices of templates. As explained in the template construction section in chapter four, sub events are aligned to occur at the same indices, in order to obtain meaningful templates (standard mean and deviation). Therefore, membership degrees of aligned indices (I_{se}) of sub events are redistributed over the other possible indices (I_{pse}) where the sub events could also occur. A redistribution operation is applied to indices ($I_{zpse_{i,j,k}}$) of which membership degree is zero and the next predicted index (N_i) is among the kth sub event indices of class C_i and channel $H_{i,j}$ ($I_{se_{i,j,k}}$).

$$I_{zpse_{i,j,k}} = I_{pse_{i,j,k}} \bigcap_t (M_{i,j,t} == 0) \cap N_i \in I_{se_{i,j,k}} \tag{5.5}$$

Actually, the aim of this operation is to fill the zero membership degrees around the observed kth sub event's indices ($I_{pse_{i,j,k}} \cup I_{se_{i,j,k}}$) in order to make more reliable frame prediction in the next component. The maximum membership degrees of kth sub event index's $M_{i,j,I_{se_{i,j,k}}}$ are distributed exponentially over $I_{zpse_{i,j,k}}$ indices. The exponential proportion is a function of the maximum membership degree of aligned indices $M_{i,j,maxI_{zpse_{i,j,k}}}$ and the distance which is between the index of maximum membership degree of aligned indices $maxI_{se_{i,j,k}}$ and tth index of $I_{zpse_{i,j,k,t}}$

$$\begin{aligned} \Delta_t &= |maxI_{se_{i,j,k}} - I_{zpse_{i,j,k,t}}| + 1 \\ M_{i,j,I_{zpse_{i,j,k,t}}} &= \frac{M_{i,j,maxI_{zpse_{i,j,k}}}}{e^{2*\Delta_t}} \end{aligned} \tag{5.6}$$

5.2.3 Frame Predictor

The *Frame Predictor (allocator)* component is a transition function in which the next index time (N) is estimated or, in other words allocated (predicted), given the degree of membership curves (M) and most recently predicted frame index (V).

$$N = Frame\_Predictor(M, V) \tag{5.7}$$

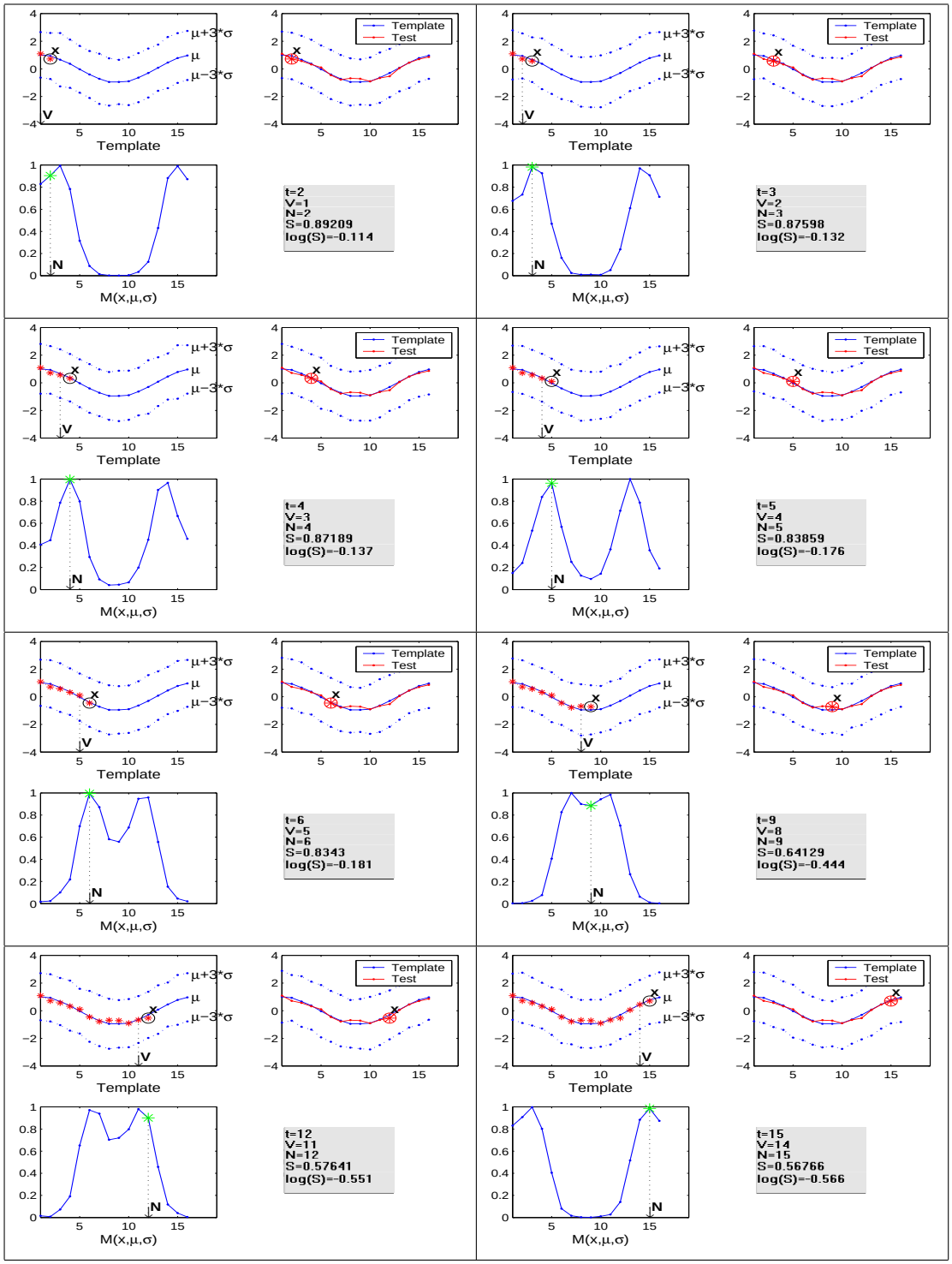


Table 5.2: The principle idea of the frame prediction process with artificial test data and class which has one channel (represented with mean and standard deviation μ and σ) over various time steps (t={ 2, 3, 4, 5, 6, 9, 12, 15}), when test data (X) is provided incrementally.

The *Frame predictor (allocator)* component predicts the possible position of the input frame in the degree of membership curves. In other words, prediction is an operation in which the whereabouts (index) of the input frame in the channels (template) is allocated. Before getting into a detailed discussion, the framed predictor is summarised as follows. The index of the local maxima (N_i) travels within the degree of membership curve from beginning to end with a monotonic and increasing order, if the input data belongs to the class of interest. This idea, the allocation of next index (N), is illustrated in the figures in table 5.2. The input frame creates local maxima (or global maximum) in the degree of membership curve wherever the frame is closer to the template frames. This characteristic of a degree of membership curve, namely the position of the local maxima, serves to predict the possible frame index. In the cases of multiple local maxima in the degree of membership curve, the nearest local maxima in the neighbourhood of the most recently predicted frame index is considered, in order to maximise the similarity function. For example, the figure (third row, first column) in table 5.2 at time $t = 6$, two local maxima have emerged (around index 6 and 12). But since the latest predicted index is 5 ($V_i = 5$), the nearest possible maxima index, namely 6 is chosen as the next predicted index ($N_i = 6$).

Figures in table 5.2 depict the prediction of an artificial one dimensional test input to a class which comprises one channel over various time steps (t={ 2, 3, 4, 5, 6, 9, 12, 15}), where the test data (X) is provided incrementally. Alongside other operations (degree of membership curve estimation, score calculation) the figure details the prediction process and its effect on the score for each time step. Each entry of the table accommodates three graphs and some relevant data. In each entry of the table, the top left graphs show the channel (μ, $\mu \mp 3 * \sigma$), historical and the latest incremental test data (X, blue circle). The top right graphs in the entries show the channel with the complete test signal (the latest test data (X) over complete test signal); The bottom left graphs show predicted (allocated) index (N, green star) over estimated degree of membership curve given X and the channel; and finally bottom right panels in the entries show estimated V, N, and score (S) for the time step t.

The frame allocation (prediction) operation depicted in table 5.2 is similar to the first synthetic example explained at the beginning of this chapter with figure 5.1. For the sake of clarity, a test signal is chosen similar to the template with some degree of noise, as shown in table 5.2 (top right). Therefore, it is expected that the prediction operation allocates (predicts) monotonic and ordered indices (N), given the test signal and the template. For better understanding of the prediction operation, let us have a closer look at the graphs for some time steps. Note that, the time step of each entry is shown in the bottom right panel.

Initially, at time t=2 (See entry at first column, first row of table 5.2), it is assumed that, V (latest predicted index) is 1 and the prediction operation allocates 2 as the next predicted index ($N = 2$). Note that, actually, at t=2, although the closest maxima point in the degree of membership curve (bottom left graph) is at index 3 (which, at the first sight, looks like the next predicted index), but index 2 is decided as the next predicted index (N=2). This is due to membership degree tolerance (ϵ) which will be discussed in detail in the next section. Briefly, the parameter ϵ is introduced to maximize the score (S) by minimizing the distance function (Δ).

At t=3 (See entry at first row, second column of table 5.2), the latest predicted index V is 2 and the next predicted index (N) is allocated 3, since the closest maxima point to V in the degree of membership curve is at index 3. Similar situations (as at t=3) arise at t=4, 5, 6 and 15. But at t=9 and 11, the situation is similar to the situation at t=2, and the prediction operation allocates indices (N) accordingly to maximize score(S) by optimizing distance function (Δ) by considering membership degree tolerance (ϵ). This is further elaborated at page 190.

5.2.3.1 Characteristic of Degree of Membership Curve (M)

Estimating or extracting the position of the *next index*, N, along a degree of membership curve (M) is non-trivial. This task is implemented as follows: In the neighbourhood of possible *next index* N on M, membership degrees should be non-zero values. Hence, firstly, clusters (sections) (κ), which consists of sequential non-zero membership degree values, are determined in M. For example, figure 5.6 illustrates some clusters along a degree of membership curve in which the latest predicted index (V) is at index of 30. A degree of membership curve (M) can accommodate none or more than one cluster. Therefore, in order to decide on the primary cluster (κ_p), which is the cluster accommodating the next index (N), the following criteria are considered:

- Length of primary cluster has to be greater than one
- Primary cluster should include the most recently predicted index V
- If V is not on border of any cluster, the next cluster subsequent to the latest predicted index V is chosen as the primary cluster.

All of these operations and criteria can be expressed in a formal way as in the following pseudo MATLAB style code for class C_i:

```
M_i=[M_i,M_i];  % Duplicate the Degree of Membership Curve
nonZero=(M_i>0);
```

```
dp=diff(nonZero);
dp=[nonZero(1),dp];
if (nonZero(end)==1)
dp=[dp,-1];
end
startIndex=find(dp==1);
endIndex=find(dp==-1);
% starting points of clusters
clusters.sp=startIndex;
% end points of clusters
clusters.len=endIndex-startIndex-1;
primaryCluster= clusters.len>1 and
                ((clusters.sp<=v & v<= clusters.sp+clusters.len)
                 or (min(abs(v-clusters.sp)))
```

Note that as depicted in the above pseudo code, a cluster structure consists of a starting point and its length. If the primary cluster is determined via a second criterion, the starting primary cluster has to be shifted to V, in order to guarantee a monotonic increase in the prediction of the index. Furthermore, the section of M between the start of the primary cluster and V ($M_{i,[sp,V-1]}$) can contain noisy and not monotonic increasing membership degree values, which can lead to a wrong next index prediction (N). For example, in figure 5.6, the vertical line around index of 30, indicates the starting point modification of the primary cluster.

Having found the primary cluster (κ_p), the next step is to find the next index N inside the primary cluster. The shape of the primary cluster (κ_p) and the maximum membership degree in the primary cluster provide us with some clues about the whereabouts of the next index N. A cluster can be in the form of the following four shapes in a degree of membership curve. Figure 5.6 illustrates these shapes.

- Steady Increase Cluster (SIC): The gradient along this cluster tends to be positive. SIC generally emerges at the end of M, because the cluster does not have a chance to decrease again as M terminates. Therefore, if the primary cluster is a SIC, N index can be predicted to be around the end of SIC with some tolerance.

- Steady Decrease Cluster (SDC): The gradient in this cluster is always negative. Unlike SIC, SDC is usually formed at the beginning of the degree of membership curve (M), since SDC has just started. Hence, if SDC is the primary cluster, the *next index* N, should be around the beginning of SDC with some tolerance.

- Single Bell Cluster (SBC): These clusters are in the shape of a bell and have

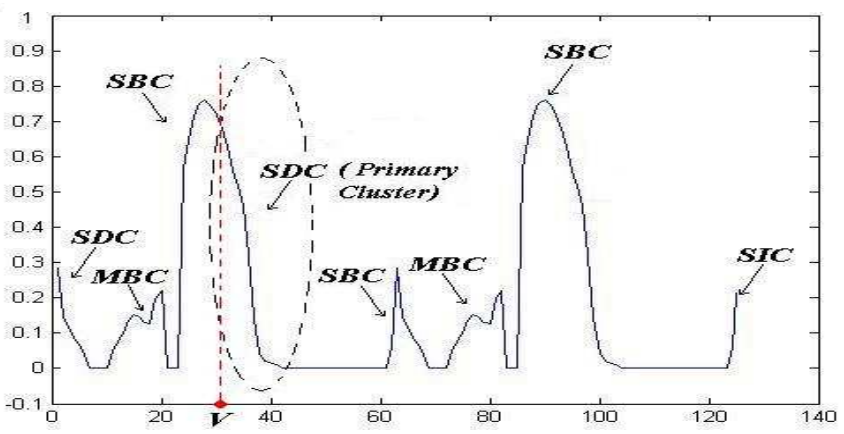


Figure 5.6: Cluster Distribution Shapes in a degree of membership curve, in which four different cluster shapes emerge: Steady Increase Cluster(SIC), Steady Decrease Cluster (SDC),Single Bell Cluster (SBC), Multiple Bell Cluster (MBC). While SIC and SDC clusters occur at the beginning and end of the curves respectively, the SBC and MBC emerge between the SDC and SIC. The primary cluster is chosen according to criteria elaborated in the previous page. The Frame Predictor (allocator) component utilises the shape of the primary clusters and the latest predicted index (V, index 30 in the figure) to allocate the next predicted index.

only one distinctive maximum point and emerge between SIC and SDC clusters. Extracting a *next point* N along a SBC cluster is more trivial compared to the SDC and SIC, as in SBC there is only one maximum point. The *next point* N is chosen around the maximum membership degree with some membership degree tolerance (ϵ).

- Multiple Bell Cluster (MBC): These clusters have more than one distinctive and dominant maxima. Similar to SBC, they can emerge between SIC and SDC clusters. A constant curve neighbourhood on a degree of membership curve creates MBC in the neighbourhood, as the given test point d can belong to several successive indices. Therefore, the *next index* is selected as the successor of the current index ($N = v + 1$), as we cannot determine the exact place of the *next index* with this information.

Although, SIC, SDC, SBC, MBC are not domain dependent, the temporal nature of the classes (dynamic or static) determines the shape of the degree of membership curve (Md_i). In the case of static classes, given a test frame (d), and since frames data and consequently degree of membership curves are constant (Eq: 2.1), the degree of membership curve (M) does not convey enough information to predict the next index N. Moreover, there is only one cluster and it is automatically taken to be the

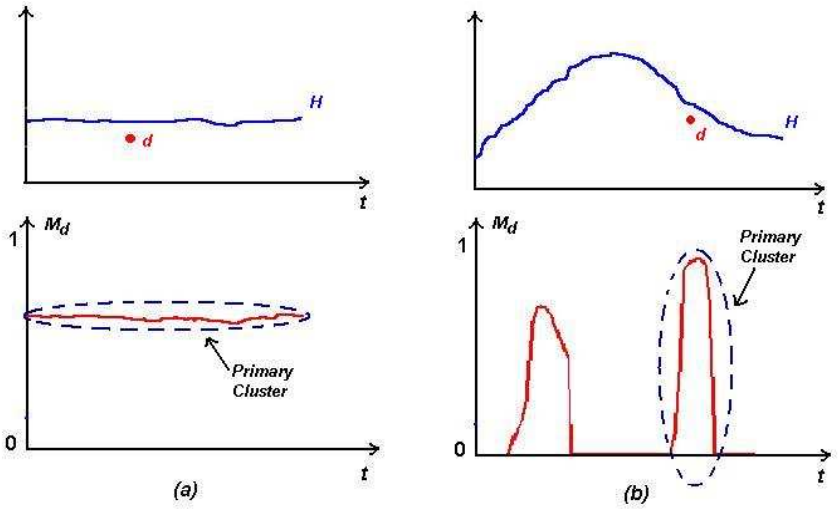


Figure 5.7: Shapes of Primary Clusters in the case of a static class (a) and a dynamic class (b)

MBC primary cluster (Figure 5.7-a). On the other hand, namely in dynamic classes, all shapes of the primary cluster can emerge in a degree of membership curve (Figure 5.7-b).

It was mentioned above, when determining the place of the *next index (N)* in the primary cluster, some membership degree tolerance (ϵ, manually set to 0.1 in the study) is taken into account in the cases of SIC, SDC and SBC. The concept of tolerance (ϵ) is deployed to maximize the similarity function (Θ) which is a function of the distance function (Ψ) and membership degree value of the predicted frame (M_N). Θ is maximised by optimizing Δ and M_N such as choosing the index N which minimizes the distance (Δ) in the neighbourhood of the maxima index with the membership degree tolerance interval. In other words, membership degree tolerance (ϵ) approximately allows to be considered the previous and successive indices of the maxima index in the primary cluster.

The equation 5.8 formulates the computation of *next index* along a primary cluster κ_p in a formal way. Bear in mind that $\kappa_{p_{mp}}$ is the maxima index in the primary cluster and ϖ is an auxiliary variable to determine the possible next index in the primary cluster via some membership degree tolerance (ϵ).

$$N_i = \left\{ \begin{array}{lll} V+1 & \text{if} & \kappa_p \; is \; MBC \\ \kappa_{p_{mp}} + \varpi & \text{if} & \kappa_p \; is \; SIC \; where \; \varpi \in \{0,-1\} \\ \kappa_{p_{mp}} + \varpi & \text{if} & \kappa_p \; is \; SDC \; where \; \varpi \in \{0,1\} \\ \kappa_{p_{mp}} + \varpi & \text{if} & \kappa_p \; is \; SBC \\ -1 & \text{if} & \kappa_p \; is \; empty \end{array} \right\} \tag{5.8}$$

In fact, from a different perspective, the frame prediction operations are based on maximising the similarity function (Θ), by utilising the distance function.

5.2.4 Score Estimator

The Score Estimator component is responsible for the estimation of *Scores* (S) for each class, given the incremental test frames. Scores play a major role in the recognition declaration in further components. *Score* (S_i) accommodates the similarity between input frame B and corresponding classes (C_i). Scores aggregate the cumulative product of similarity factors (Θ), which consist of the distance function (Ψ), and the membership degree of predicted indices (M_N).

The smaller the distance between the consecutive predicted next index (Δ) and the greater membership degree (M_N), the greater the similarity between input data and corresponding classes. [118].

$$\begin{aligned} log(S) &= \sum^{T} log(\Theta) \\ &= \sum^{T} log(M_N \Psi(\Delta)) = \sum^{T} log(M_N e^{-\frac{(\Delta-1)^2}{2}}) \end{aligned} \tag{5.9}$$

As expressed before, *log* is applied on Scores to prevent exponential decay of the cumulative product of similarity factors.

Unfortunately, score estimation in Eq:5.9 has some deficiencies. For example, in the frame predictor component, if the same index ($N = V$) is predicted continuously and consequently for a period of time in Eq:5.9, S may lead to a wrong recognition, because a consistent monotonic increase ($\Delta > 0$) of next index (N) is not obtained. If this case occurs several times consecutively, the accuracy of scores S will be suspect. Therefore, a variable, number of consecutive predicted same indices (η_i) is deployed to reset the score S_i if this condition is not met consecutively over a certain time period (η_{Max}). The value of η_{Max_i} for each class is approximately a quarter of the class's

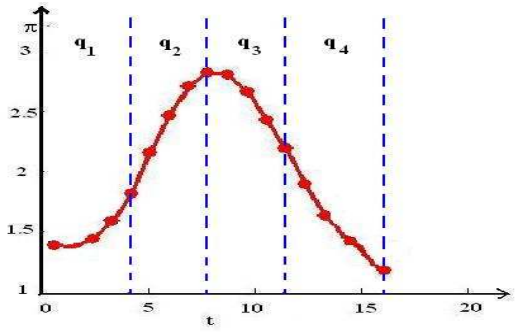


Figure 5.8: Class State (q_1, q_2, q_3, q_4) Boundaries

period ($\eta_{Max_i} = L_i/4$). In addition to resetting *Score*, η is also embedded into the score estimation as a Gaussian membership function ($e^{-\frac{(\eta-\mu)^2}{2\sigma^2}}$) where the expected mean (μ) and standard deviation (σ) for η is 0 and 1 respectively as follows:

$$log(S) = \sum^{T} log(M_N \Psi(\Delta_i) e^{-\frac{\eta_i^2}{2}}) = \sum^{T} log(M_N e^{-\frac{(\Delta_i - 1)^2 + \eta_i^2}{2}}) \tag{5.10}$$

As mentioned earlier, in an ideal case, a consistent monotonic increase among consecutive current indices ($\delta > 0$) is expected. If a sudden *long jump* occurs between two consecutive indices in a class (C_i), it can be interpreted that the given input point (d) does not belong to the class C_i. Therefore, the score of that class can be reset ($S_i = 0, V_i = 0$). If the jump is larger than a quarter of the class period, it is considered a large enough jump ($\delta < L_i/4$).

In the light of the η and *long jump* heuristics modifications, the final score estimation equation 5.9 is modified as follows:

$$\begin{aligned} \Delta_i &= |N_i - V_i| \\ log(S_i) &= \begin{cases} \sum^{T} log(M_N e^{-\frac{(\Delta_i - 1)^2 + \eta_i^2}{2}}) & \Delta_i \leq L_i/4 \text{ and } \eta_i \leq L_i/4 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{5.11}$$

5.2.5 Path Assessor

Even though, *score* (S) is one of the major measurements indicating similarities, it does not accommodate any information in itself, especially in the case of continuous recognition, what time or in which conditions it is appropriate to declare if a class recognition has emerged. The order of predicted indices (N) or the path of observed

indices can give a more accurate declaration. These heuristics are deployed in the Path Assessor component. It prevents premature or wrong recognition and provides auxiliary information to the *Decider* component, in order to evaluate all status and declare a recognition if one has emerged.

It is stated that in a consistent recognition, the predicted frame index N_i must be in an order, namely, follow a monotonic increasing path from beginning to end within the degree of membership curve (M_i). In this study, it is assumed that, M_i is consolidated by six consecutive parts or milestones, Q_i=$\{q_{i,0}, q_{i,1}, q_{i,2}, q_{i,3}, q_{i,4}q_{i,R}\}$ which are referred to as *path* in the rest of the thesis. Each part occupies a quarter of the class period ($0 < q_{i,1} < 0.25 * l_i < q_{i,2} < 0.5 * l_i < q_{i,3} < 0.75 * l_i < q_{i,4} \leq q_{i,4}$). q_0 and q_R are the starting and final milestones. When the score or current index ($S_i = 0, V_i = 0$) is reset, the milestone of the class is also reset($\chi_{*\rightarrow 0}$). On the other hand, while the states q_1 and q_4 stand for starting and end sections of the template respectively, q_2 and q_3 stand for the middle sections. This component ensures that all the parts are observed with a monotonically increasing order from $q_{i,1}$ to $q_{i,4}$ ($\chi_{0\rightarrow 1\rightarrow 2\rightarrow 3\rightarrow 4\rightarrow R}$) as illustrated in figure 5.8 on an imaginary template. Note that $q_{i,R}$ is automatically followed by q_0 for $R = 1, 2, 3, 4$ once recognition is obtained. If any jump occurs in the path, for example from $q_{i,1}$ to $q_{i,3}$ or $q_{i,4}$ rather than $q_{i,2}$, the score and path will be reset ($S_i = 0, V_i = 0, Q_i = q_{i,0}$). Actually sequential path observation is also guaranteed in the previous *Score Estimator* component via the *long jump* heuristic.

$$
\begin{aligned}
\chi_{0\rightarrow 1} &: V_i < \ 0.25L_i \\
\chi_{1\rightarrow 0} &: V_i == 0 \| V_i > \ 0.5L_i \| (qa_{i,1} \leq 0.1L_i \& V_i \geq 0.25L_i \& V_i < 0.5L_i) \\
\chi_{1\rightarrow 2} &: qa_i, 1 > \ 0.1L_i \& V_i \geq 0.25L_i \& V_i < 0.5L_i \\
\chi_{2\rightarrow 0} &: V_i < \ 0.25L_i \| V_i \geq 0.75L_i \| (qa_{i,2} \leq 0.1L_i \& V_i \geq 0.5L_i \& V_i < 0.75L_i) \\
\chi_{2\rightarrow 3} &: qa_{i,2} > \ 0.1L_i \& V_i >= 0.5L_i \& V_i < 0.75L_i \\
\chi_{3\rightarrow 0} &: V_i < \ 0.5L_i \| (qa_{i,3} <= 0.1L_i \& V_i >= 0.75L_i \& V_i < L_i) \\
\chi_{3\rightarrow 4} &: qa_{i,3} > \ 0.1L_i \& V_i > 0.75L_i \& V_i < L_i \\
\chi_{4\rightarrow 0} &: V_i < \ 0.75L_i \| (qa_{i,3} <= 0.1L_i \& V_i < 0.25L_i) \\
\chi_{4\rightarrow R} &: qa_{i,4} > \ 0.1L_i \\
\chi_{R\rightarrow 0} &: V_i < \ 0.25L_i
\end{aligned}
\tag{5.12}
$$

Until a class has not reached the end part of the template namely the milestone q_R a with monotonically increasing milestone order, a recognition is not announced. But,

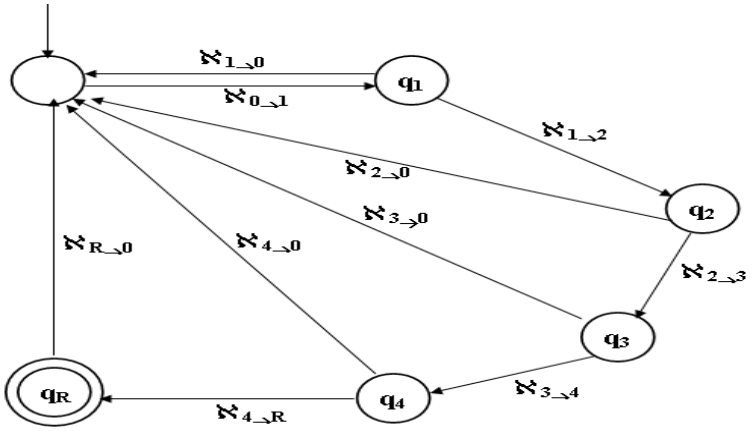


Figure 5.9: Ideal monotonic increasing path order (Milestone Transitions) q_0 and q_R are initial and final milestones. Transition conditions from milestone i to j ($\chi_{i\rightarrow j}$) are formulated in equation 5.13.

recall that one of the most challenging aspects of continuous on-line recognition is that the starting and end frames of the samples on band (B) are not known in advance. Yet, the current matched index (V) is used as the place indicator to decide the current milestone.

But unfortunately, just following the path in an incremental and monotonic manner is not enough. For example, a monotonic increasing milestone order can be obtained by observing just one index at each path section. Therefore, a sufficient set of indices V_i (a threshold, at least, 10 % of class period, $0.1L_i$) has to be observed in each path section to build a confidence for the observed path. So, this component also holds the number of $N_i \neq V_i$ observations (path age, QA) for each path part ($qa_{i,1}, qa_{i,2}, qa_{i,3}, qa_{i,4}$).

Figure 5.9 illustrates the path or milestone transition and equation 5.13 shows the transition conditions from milestone i to j ($\chi_{i\rightarrow j}$). q_0 and q_R is the initial and final milestones respectively. Recognition is announced at milestone q_R, from which the system milestone is transited to q_0 ($\chi_{R\rightarrow 0}$), when an index V is predicted in the first part of the path ($V_i < 0.25L_i$). Transition conditions are basically based on path age and path boundaries. If at any time point, the current index V_i is suddenly placed out of the current k path boundary or not enough path age is gathered in the current boundary ($qa_{i,k} \leq 0.1L_i$), the system milestone is reset ($\chi_{k\rightarrow 0}$). Otherwise, if enough index is observed ($qa_{i,k} > 0.1L_i$) and V is in the next path section, the system milestone is transited to the next one ($\chi_{i\rightarrow i+1}$).

For further process such as feedback, training or synthesis purposes, predicted indices, namely the sequence of N_i, is also stored.

5.2.6 Decider

The *Decider* component is responsible for deciding whether reliable intelligence is gathered for a recognition announcement. This component utilises Score (S) and Path heuristics (Q) to announce recognized class C_R as follows:

- S_R has to be maximum among all scores.
- Path milestone of the class (C_R) has to be Q_R as explained in the *Path Assessor.* component. In order to be at the final milestone, conditions and transitions shown in equation 5.13 and figure 5.9 have to be obtained.

$$C_R = \underset{i}{argmax}\ S_i \ \&\ Q_i == q_R \tag{5.13}$$

5.3 Algorithm Analysis

So far, the proposed algorithm, Recognition Machine (RM) has been described in detail including the intuition behind its development using a synthetic example. For future analysis and discussion, the component of RM is summarized in the following pseudo code. Note that pseudo code is only for the class C_i and inter membership degree estimation ($M_{i,j}$) is carried over all channels ($H_{i,j}$) of the class C_i. For the sake of clarity and space, the function of *Pre-Processing* on line 6 and *Path Assessor* on line 22 are omitted. Refer to the related sections (sections 5.2.1, 5.2.3 and 5.2.5, respectively) for further information about these sections.

Because of the vital role of the *Frame Predictor* in the ***RM***, this gist of the component is presented here again. The *Frame Predictor* on line 9 in the pseudo code is responsible for :

- N_i:Allocating the next index (N_i) of the test frame (x) on the class template (C_i)
- M_{N_i}:The degree of membership of the test data (x) to the class C_i,

given the degree of membership curve (M_i) (as computed in line 7 and 8) of test data and the latest allocated index (V_i). The *Frame Predictor* component utilises the local maxima points near to the latest predicted index (V_i). The input frame creates local maxima (or global maximum) in the degree of membership curve wherever the input frame is closer to the template frames. This characteristic of the degree of membership curve, namely, the position of the local maxima, serves to allocate possible frame index

(N_i). The principle idea behind the *Frame Predictor* process is further illustrated with an artificial example in the figures in table 5.2.

% Initialization

1 $V_i \leftarrow 0;\ S_i \leftarrow 0;\ Q_i \leftarrow q_0;\ QA_i \leftarrow 0;\ C_R \leftarrow -1;\ t \leftarrow 0;$

3 $while\ (t < T)$

4 $begin$

% Data Pre-Processing

5 $d = B(t);$

6 $x = pre - Processing(d);$

%Intra Channel Membership Degree Estimation

7 $M_{i,j} = \frac{p(x_{j,t}, H_{\mu_{i,j,t}}, H_{\sigma_{i,j,t}})}{p(H_{\mu_{i,j,t}}, H_{\mu_{i,j,t}}, H_{\sigma_{i,j,t}})} = e^{-\frac{(x_{j,t} - H_{\mu_{i,j,t}})^2}{2H_{\sigma^2_{i,j,t}}}}$

% Inter Channel Membership Degree Estimation

8 $M_i = \sqrt[\vartheta]{\prod_{j=1}^{\vartheta} M_{i,j}}$

% Predictting Next index N_i

9 $[N_i, M_{N_i}] = Frame\_Predictor(M_i, V_i)$

10 $\Delta_i = |N_i - V_i|$

% Check if same index predicted

11 $if\ (V_i \neq N_i)\ then$

12 $\eta_i \leftarrow 0$

13 $else$

14 $\eta_i \leftarrow \eta_i + 1;$

15 $endif$

% Score Estimation

16 $log(S_i) = \left\{ \begin{array}{ll} \sum^{T} log(M_N e^{-\frac{(\Delta_i - 1)^2 + \eta_i^2}{2}}) & \Delta_i \leq L_i/4\ and\ \eta_i \leq L_i/4 \\ 0 & \text{otherwise} \end{array} \right\}$

% Update Current Index (V_i), if Score is non-zero

17 $if\ (S_i \neq 0)\ then$

18 $V_i \leftarrow N_i$

19 $else$

20 $V_i \leftarrow 0;\ Q_i \leftarrow q_0;\ QA_i \leftarrow 0;$

21 $endif$

% Update Path Milestione as in figure 5.9 and Equation 5.13

22 $[S_i, V_i, Q_i] = PathAssessor(S_i, V_i, Q_i)$

% And finally judge if recognition is emerged

23 $C_R \leftarrow argmax_i\ S_i\ \&\ Q_i == q_R$

24 $t \leftarrow t + 1;$

25 end

The *Frame Prediction* component in the recognition machine addresses temporal variance and the *start/end* issues of temporal pattern recognition. On-line prediction and the piecewise matching operation pave the way for resolving the issues of temporal variance, and automatically identify the *start/end* of a gesture (automatic segmentation). For each input frame, corresponding frames in the class templates are predicted in the *Frame Prediction* component. Therefore, these operations enable us to detect the start and end frames of gestures and adapt them to temporal variances.

Furthermore, in the *PathAssessor* component, recognition milestone ($Q_R = q_4$) can be taken backwards to ($Q_R = q_3$) with some more heuristics to make earlier recognition, if a confidence is built on score S. Especially in isolated gesture recognition, it is observed in the experiments that it is possible to declare recognition during the milestone q_3.

Having discussed the components of RM, an analysis into the time complexity of RM will be appropriate. A reasonable amount of time has been spent on estimating intra channel degree of membership curves ($H_{i,j}$) for a given test frame (d) ($L_i\vartheta$). Furthermore, inter degree of membership curves and frame predictor components take time proportionally with periods ($2L$). And all of these operations are carried out for each class (ϖ) and test frame (T). Thus, the complexity of the proposed algorithm RM is:

$$O((L\vartheta + 2L)\varpi T) \tag{5.14}$$

where ϖ, ϑ and L correspond to the number of classes, number of channels and average periods of class.

As equation 5.14 illustrates, the time complexity of RM is mostly correlated with periods of classes. It is possible to reduce time complexity of the algorithm by estimating and evaluating only valid the section of templates. For example, the Score Estimator and Path Assessor components deploy *long jump* heuristics to secure a monotonic incremental frame index prediction. If any next frame index (N_i) is estimated at a quarter period length far away ($\Delta_i > L_i/4$) from the current index (V_i), scores are reset. In other words, therefore, it is not always necessary to estimate and evaluate all indices. During membership degree estimations and frame predictors, the indices, which are $L_i/4$ far away from the current index V_i can be omitted. Hence, the time complexity of RM can be reduced by half to ($O((\frac{L}{2}\vartheta + 2\frac{L}{2})\varpi T)$).

The recognition algorithm under discussion exploits temporal and spatial characteristics of gestures via dynamic programming and the Markovian process. The frame prediction operation in RM is based on the first order Markovian process in which,

the next index N prediction is based on only the current index (V) and degree of membership curve (M). RM, however, can readily be modified for the high order (4-5) Markovian process as follows: Instead of utilising only the latest $M_{i,t}$, if an aggregated version (aM_i) of the latest n degree of membership curves $M_{i,t-n...t}$ is used, n order Markovian process can be obtained. A geometric aggregation operator can be employed for the aggregation function. $n-order$ Markovian process in Frame Prediction will be more reliable, as it will reduce the noise (more smoother) in the degree of membership curve, which will lead to a better maxima analysis on the curve for the prediction of the next index N.

$$\mathtt{a}M_{i,t} \quad = \quad \chi_{j=1}^{n} M_{i,t-n+j} \tag{5.15}$$

where χ corresponds to an aggregation operator.

The recognition machine is a component-based modular system. RM is a combination of well-defined and connected components. For each task, the input and output of each component are well defined. Therefore, in terms of performance, accuracy and effectiveness, the components of the recognition algorithm can be modified or completely substituted for other conventional algorithms for different domains and temporal pattern recognition problems, if needed. For instance, a different smoothing algorithm can be used in pre-processing or in membership degree estimation; a different statistical distribution function can be used rather than the "Gaussian curve membership function" for a membership degree. Furthermore, for example, the function of the frame predictor component can be replaced by a function approximation algorithm such as MLPNN or RBN neural networks [8]. In the next subsection, such an implementation will be explained in detail.

5.3.1 Multilayer Perceptron Neural Network as the Frame Predictor

Since RM is a modular system, it is versatile enough to implement a hybrid implementation of itself. In this sense, the *Frame Predictor* is a very promising component because the task of the *Frame Predictor* is a kind of function approximation. As described earlier in the frame predictor section, the next index prediction (N) is a transition function in which (N) is estimated given the degree of membership curves (M) and the most recently predicted frame index (V).

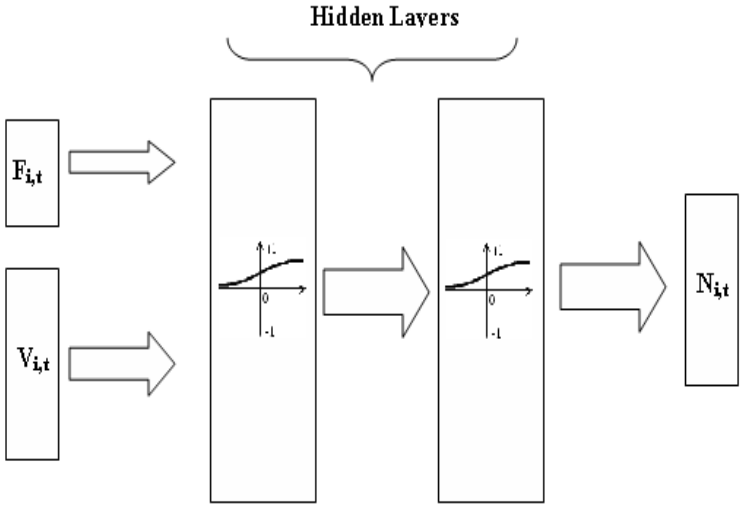


Figure 5.10: Multilayer Perceptron Neural Network for Frame Prediction. MLPNN predicts N_i directly from the preprocessed input frame (feature vector,$F_{i,t}$) and $V_{i,t}$ for the class C_i at time t.Current index input (V_i) and output (N_i) is represented as binary. Input, hidden and output layers have $(\vartheta + L)$, 90, 90, and L_i neuron respectively. As activation function *logsig*, *purelin* is used for two hidden and output layer respectively. Since the output layer is also coded, the node which has the maximum value in the output layer is considered as the predicted node, namely the next index (N).

$$N = Frame\_Predictor(M, V)$$

The frame predictor exploits the degree of membership curves to locate maxima points around the current index V. Therefore, it is possible to deploy a multilayer perceptron or radial basis neural network to approximate the function of the *Frame Predictor* component.

In this thesis, in order to validate the modularity of RM, MLPNN is employed for *Frame Predictor* approximation with some modification. Unlike the *Frame Predictor*, MLPNN predicts N_i directly from the pre-processed input frame (feature vector, $F_{i,t}$) and $V_{i,t}$ for the class C_i at time t. Therefore, by omitting degree of membership curve estimations ($M_{i,j}$ and M_i), the time and space complexity of RM is reduced.

$$N_{i,t} = MLPNN(F_{i,t}, V_{i,t})$$

Note that for each class, a MLPNN is employed. The current index input (V_i) and

output (N_i) are represented as binary vector. $(\vartheta + L) \times 90 \times 90 \times L_i$, two hidden layers feed and forward MLPNN architecture is deployed for the class C_i. More specifically, the input, hidden and output layers have $(\vartheta + L_i)$, 90, 90, and L_i neurons respectively. As the activation function *logsig*, *purelin* is used for the two hidden and output layers respectively. Since the number of neurons and training cycles are large, in order to keep training time and resource lower, the supervised adaptation method in MATLAB (*newff, adapt*) is preferred for training. In training, the adaptation number of epochs is set to 1000. Since output layer is also coded, the node which has a maximum value in the output layer is considered as the predicted node, namely, next index (N).

Apart from predicting N_i, the output layer of the proposed MLPNN hybrid system can be used as a membership degree (M_{N_i}) of the input test ($F_{i,t}$). This assumption can be justified if N_i and output values are mapped into the range of [0, 1]. Since during the training phase, target values are coded as either 0 or 1, the trained frame predictor MLPNN can be used trivially for M_{N_i} estimation with mentioned output mapping.

The FDO_PT dataset is used to validate the hybrid MLPNN/RM system for isolated gesture recognition. Detailed discussion of this experiment is presented in section 6.4 in the following experiment chapter. Briefly, the following observations are noted: Approximation power of the hybrid MLPNN/RM system is reasonable on dynamic gesture but on static gesture it is limited compared to RM, because the spatial and temporal feature vector in the case of static gesture does not discriminate enough between frames.

5.4 Analogy with other Algorithms

As illustrated in figure 5.11, in RM, a class C_i can be thought of as a chain of L_i states (s_j), each of which consists of ϑ channels. Approaching the template as a chain of states enable us to make the analogy between the proposed recognition algorithm and widely used algorithms such as the Hidden Markov Model (HMM) and Dynamic Time Warping (DTW). As discussed in the Literature Review chapter, HMM is a stochastic finite state automata, in which the emission of observations and transitions between states are expressed in a probabilistic manner [12, 97]. DTW is an off-line template matching algorithm, in which time dimension is warped monotonically and increasingly in a window bandwidth, in order to minimize the distance between input and reference template.

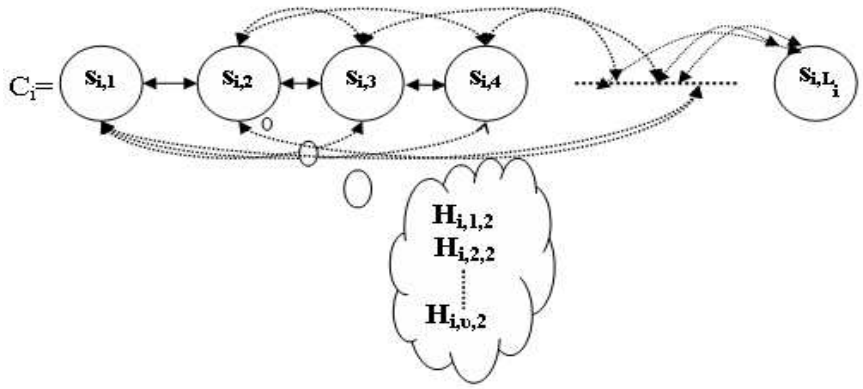


Figure 5.11: Representation of class C_i in RM as chain of L_i states s_i in order to make an analogy with HMM and DTW. Each state s_t accommodates ϑ channels ($H_{i,1...\vartheta,t}$). Even RM is partially fully connected, but for the sake of clarity some transitions are skipped. Transition are biased from left to right and transition probability from one state to its right neighbour is higher (bold solid) than others (dashed transitions). In RM, transitions are controlled by *Frame Predictor* and *Path Assessor* components.

5.4.1 Analogy with Hidden Markov Model (HMM)

The proposed algorithm can be reduced to a Hidden Markov Model as a special case. The distance function (Ψ) and the membership degrees values of the predicted index (M_i and M_N) correspond approximately to the transition and emission probabilities in HMM, respectively. Even though, RM addresses some common issues of HMM such as training, decoding, and evaluation [97].

Model designing, namely the optimal number of states and topology, is one of the main issues in HMM. Determining a model is based on trial and error. Note that HMM is mainly exploited and developed in speech recognition domains. But compared to speech, a gesture trajectory does not contain that much complexity and variety. In other words, unlike HMM in speech recognition, the modelling of gesture data does not require *hidden* states which aim to represent an unknown infrastructure. Gesture data or trajectories, roughly speaking, are easier to observe than speech. The trajectory of gesture in a 3D coordinate system is adequate to represent a state. In other words, each frame observed in 3D corresponds to a state in HMM. Undoubtedly, using all the data points in a trajectory will increase the number of states, which is somehow wrongly believed that it is not optimal in HMM. Contrary to that myth, the latest research concludes that increasing the number of states in HMM models leads to a remarkable recognition rate [102]. Therefore, from this aspect, namely by employing all frames in a trajectory, the optimal number of states in HMM is addressed.

Moreover, the algorithm does not consist of the main training issues of HMM such as transition and emission probabilities. In HMM, EM or Bauch-Welsher algorithms are employed to estimate optimal transition probabilities. EM or Bauch-Welsher utilise the transition and emission *expectations* to estimate the probabilities. But due to RM design, RM already accommodates the *expectation* in its *Frame Predictor* component. RM does not involve any training for transition probabilities. Since gestures are based on more observable states and each frame in a trajectory is represented by a state (s_i), it is expected that the transition between neighbourhood frames/states (s_i) is more frequent than the other remote frames. By employing all frames in a trajectory as a state, and using a large number of states this can ensure a small transition and emission probabilities variance between consecutive indices. This expectation phenomena is employed by the Gaussian curve membership function ($e^{-\frac{(\Delta-\mu)^2}{2\sigma^2}}$) where both expected (mean, μ) transition distance (Δ) and its standard deviation (σ) are 1. Note that RM is a partial connected graph which is biased from left to right transition. It is partial because, as was explained earlier, long jump or transitions are prevented in *Score* and *Path Assessor* components.

The *Frame Prediction* component implements a straight forward mechanism for decoding problems of HMM. Intuitively, the *Frame Prediction* utilises the degree of membership curves M_i (emission probabilities) to decode transition paths, unlike HMM in which emission and transition probabilities are not related at all. But, it is intuitive that transitions in the neighbourhood of maximum emission probabilities (maxima in degree of membership curves, M_i) are more probable. Therefore, emission probabilities can be directly utilised for decoding as has been deployed in RM unlike HMM. Briefly, in RM, emission probabilities (degree of membership curve, M_i) directly play an important role in deciding transitions.

Unlike HMM, by employing a large number of state and transparent decoding, RM provides valuable feedback for training purposes and synthesis. In HMM, a smaller number of hidden states is good for the recognition rate, although it is not representative enough for training purposes because a small number of states does not produce a meaningful path when state transition path are decoded.

As explained above in the recognition analysis section, and similar to HMM, proposed RM is a first order Morkovian process. But unlike HMM, RM can easily be extended to a high order Markovian process by utilising degree of membership curves (M), which enables us to obtain more reliable recognition.

In addition, in on-line recognition, the proposed algorithm provides more control

parameters (e.g. path assessors) to prevent premature or incorrect recognition, unlike HMM. Maximum likelihood criteria and some threshold mechanism are the only available methods in HMM. It is worth noting that controlled recognition is critical for training and feedback. For example, as we will see in the experimental chapter, in *Yang* and W_TTest2 experiments, it was observed that, while HMMs misrecognise some deformed and uncompleted gestures, the proposed algorithm rejects any recognition, which is vital for reliable training.

Another issue regarding HMM is the modelling of undefined connector movements (transition data) between defined gestures in case of continuous recognition. In literature, HMMs generally employ various techniques to model these undefined movements [153, 73, 148]. But unlike other temporal recognition domains such as speech and handwriting, undefined connectors or movements in gesture recognition domain are more complex. Therefore representing these movements with models does not provide meaningful representation. In RM a different approach, heuristic-based rejection, is employed [152]. Instead of modelling of every undefined movements, RM uses heuristics to decide if a movement is an undefined gesture. For RM, if test data is not classified as any defined gesture, it is assumed that the test data is a type of undefined gesture (C_{NoN}).

5.4.2 Analogy with Dynamic Time Warping (DTW)

The algorithm conceptually is a template matching technique in which time warping is employed in an on-line mode. In this sense, it is similar to dynamic time warping (DTW) apart from being in off-line mode. Recall that, DTWs make comparisons between a reference and input template. But in the proposed algorithm, only an input frame X is compared to reference templates C_i. Moreover, in the proposed algorithm, since the distance operations are carried out over the degree of membership curves (membership probabilities), the issue of common distance units in DTW is eliminated.

5.5 Summary

The recognition machine (RM) conceptually is an on-line template matching technique. The main idea behind the recognition algorithm is to exploit the sequential consistency of the input frames according to class models by using a dynamic programming paradigm and the Markovian process. Sequential consistency or so-called *Score* (S) addresses similarity between the incremental input data and the class models. *Scores* employ similarity factors (Θ) for each class with an on-line sequential decision process

which involves some predictions. The prediction process is a probabilistic estimation of the index of frames (N) in each class (C) which are spatially closest to the input frame (X), given the most recently predicted frame index (V).

The recognition machine is implemented according to the classical pattern recognition framework [87]. The recognition machine (RM) has nine interacting components. RM is fed by a sequence of input frames or input band B, of which its properties are defined in the problem statement. Subsequent to acquiring data incrementally from the band ($b(t)$) at each discrete time t, data is pre-processed. Then, the pre-processed data (x) is matched with all the channels of classes to obtain channel degree of membership curves. In each class, channel degree of membership curves are aggregated to obtain a final degree of membership curve (M), which represents the membership degree of x to the class. In the frame predictor component, given the most recently predicted frame (V) and M, the next frame (N) is predicted. Then, in the following component *(score estimator)*, scores (S) are estimated based on the cumulative product of similarity factors (Θ), which consists of distance function (Ψ), and the membership degree of the predicted frames (M_N). In the final two components, some auxiliary conditions are checked to see whether a recognition has emerged.

The following two metrics can be considered as similarity factors: A function of the distance ($\psi(.)$) between consecutive predicted frame indices (N), and a membership degree of input frame to the predicted frames (M_N). The distance function ($\psi(.)$) utilizes the consistency along the sequence of predicted input frames index (N). A monotonic, steady incremental behaviour in the sequence of the predicted frame indices points out consistency or similarity between the input frames and the class model of interest. The distance function is a Gaussian membership function ($e^{-\frac{(\Delta-\mu)^2}{2\sigma^2}}$) where the expected (mean, μ) transition distances between successive distance and standard deviation (σ) is 1.

Degree of membership curves (M_i) estimation involves a partial on-line template matching operation ($M_i = P(X|C_i)$). It estimates the probabilities (M_i) of the input frame (X) belong to the frames of each class model (C_i) in two stages. The first stage is a low level channel membership degree ($M_{i,j} = P(X_j|H_{i,j})$) estimation. The second phase is aggregation of channel membership degrees ($M_{i,1\cdots\eta}$) in order to obtain the ultimate class membership degree (M_i). Note that the parameter $M_{i,j}$ contains intra-membership degree redistribution. Intra redistribution regulates membership degrees among the indices which are aligned during training phases because of temporal variances of sub events.

The frame predictor component predicts possible the position of the input frame

in the class templates given the degree of membership curve (M_i) and most recently predicted frame index. The index of the local maxima (N_i) travels within the degree of membership curve from beginning to end with a monotonic and increasing order, if the input data belongs to the class. The input frame creates a local maxima in the degree of membership curves wherever the frame is closer to the template frames. This characteristic of the degree of membership curve, namely, the position of the local maxima, serves to predict the possible frame index. In the cases of multiple local maxima in the degree of membership curves, the nearest local maxima in the neighbourhood of the most recently predicted frame index is considered.

Score (S) is the primary criteria for classification. But, in order to prevent premature recognition and increase recognition reliability, some auxiliary heuristics *Path* and *Path Age* are proposed beside Score S. As a heuristics, the order of predicted indices or the path of observed indices can help determine a more accurate declaration. These operations are employed in the *Path Assessor* component. It prevents premature or wrong recognition and provides auxiliary information to the *decider* component, in order to evaluate all status and declare a recognition if one has emerged.

Path heuristics guarantee that the predicted frame index N_i is in an order, namely, follows a monotonic increasing path from beginning to end within the degree of membership curve (M_i). In this study, it is assumed that, M_i is consolidated by six consecutive parts or milestones, Q_i=$\{q_{i,0}, q_{i,1}, q_{i,2}, q_{i,3}, q_{i,4}q_{i,R}\}$. q_0 and q_R are starting and final milestones. Other parts occupy sequentially a quarter of the class period, which have to be observed with a monotonic increasing order from $q_{i,0}$ to $q_{i,R}$ $((\chi_{0\rightarrow1\rightarrow2\rightarrow3\rightarrow4\rightarrow R}))$.

Path Age heuristics assures that a sufficient N_i (a threshold, at least, 10 % of class period, $L_i/10$) is observed in each part to build a confidence for the observed path.

Having accumulated current status (path assessor, scores), finally, it can be decided whether or not a recognition has emerged. The following conditions have to be met for an on-line recognition (R_C): (1). S_i has to be maximum (2). The path (Q_i) has to be in a sequential order in terms of the predicted frame indices and N_i must be in the final milestone part ($Q_i = q_R$). (3). The duration in each part QA_i must be greater than a tenth of class period ($0.1L_i$).

RM can be readily modified for a high *n-order* Markovian process by utilising the last n degree of membership curves for each class.

On-line prediction and piecewise matching operations pave the way for resolving issues of temporal variance and identifying the *start/end* of a gesture. For each input frame, corresponding frames in the class templates are predicted. Therefore, these operations enable us to detect the start and end frames of gestures and adapt to temporal variances.

The complexity of the recognition machine RM is: $O((L\vartheta + 2L)\varpi T)$ where ϖ, ϑ and L correspond to the number of classes, number of channels and the average periods of class. The time complexity of RM is mostly depends on the periods of classes. It is possible to reduce the time complexity by half by estimating and evaluating only the valid sections of templates.

The recognition machine is a component-based modular system. Tasks, and the input and output of each component are well-defined. Therefore, in terms of performance, accuracy and effectiveness, the components of the recognition algorithm can be modified or completely substituted for other conventional algorithms for different domains and temporal pattern recognition problems, if needed. As an example, an MLP neural network was proposed to replace the function of the frame predictor component.

The recognition algorithm exploits temporal and spatial characteristics of gestures via dynamic programming and the Markovian process. It has similarities with the Hidden Markov Model (HMM) and Dynamic Time Warping (DTW). RM can be represented in the HMM framework as a special case for more observable (not hidden) temporal pattern recognition. The distance function (Ψ) and the degree of membership curves (M_N) in RM approximately correspond to the transition and the emission probabilities in HMM, respectively. But moreover, with auxiliary heuristics and assumptions, RM is designed to overcome some of the common issue associated with HMM, such as training, decoding and evaluation. In addition, in on-line recognition, RM provides more control parameters (e.g. path assessors) to prevent premature or incorrect recognition, unlike HMM. RM deploys some rejection heuristics to spot undefined movements, whereas HMM represents these undefined movement as other models. In respect of DTW, the frame prediction component implements an on-line frame-based DTW, in which the input frame is matched with all classes of templates.

Chapter 6

Experiments And Results

This chapter focuses on isolated and continuous gesture recognition experiments and discussions surrounding the proposed recognition algorithm (Recognition Machine, RM). It also covers the comparison of RM with other well-established recognition algorithms using various artificial and supplementary real world datasets in addition to FDO datasets. In addition, various data analysis techniques, which are elaborated in the Gesture Modelling and Analysis chapter, are deployed to reveal the complexity and inter similarity of datasets before experiments were conducted.

In this thesis, in addition to FDO datasets (FDO_PT and FDO_CV), one artificial (W_Test [57]) and three supplementary real world datasets (Gesture Panel [146], Yang [64, 63], Perrotta [89]) are investigated. The artificial dataset, W_Test, contains several control parameters which are utilised to analyse performance of RM and other recognition algorithms with different combinations. On the other hand, supplementary real world datasets are covered to analyse the performance of RM in various real world scenarios in terms of the characteristics of data, data acquisition techniques (for example computer vision, tracker-based) and the number of deployed users for data collection.

Two types of gesture recognition experiments are conducted in this study: isolated and continuous. In the isolated case, each sample is fed to the recognition algorithm separately; hence, the issue of start/end points is eliminated for recognition algorithms. Isolated experiments are conducted on artificial, supplementary real world and FDO datasets. In the continuous case, FDO_PT and FDO_CV gestures are considered separately. Isolated FDO_PT gestures are concatenated with various control parameters to construct continuous gesture sentences. In this case, the start and end points of each gesture is not known in advance by the recognition algorithm.

In this chapter we compare the performance of the proposed algorithm with Dynamic Time Warping (DTW), the Hidden Markov Model (HMM) and the Elman Recurrent Neural Network (ERNN) for isolated recognition over the above mentioned datasets. In case of continuous recognition, only HMM and RM are applied as these two algorithms perform better compared to others in an isolated recognition case. In addition, a hybrid version of RM is also introduced by substituting its frame prediction component with a multilayer perceptron neural network (MLPNN) for an isolated FDO_PT dataset in order to validate the modularity of RM.

For the remainder of this chapter, firstly, the methodology used for experiments is laid out. Then, the artificial and supplementary dataset with its description and inter/intra complexity analysis is covered in detail before recognition algorithms are applied for isolated recognition. Before continuous gesture recognition, a variant of RM with MLPNN is investigated in the case of isolated recognition gestures for the FDO_PT dataset. After this, some continuous gesture recognition experiments on FDO_PT and FDO_CV sentences with various lengths and control parameters are considered. The chapter concludes with the outcomes of the experiments and a summary.

6.1 Methodology

In order to validate the recognition machine (RM), several isolated and continuous recognition experiments were conducted over artificial and real world datasets with various recognition and data analysis algorithms. Before looking at the recognition experiments, datasets are first described with comprehensive data analysis. These data analysis techniques are elaborated in detail over the W_Test dataset, FDO_PT and FDO_PT datasets in Chapter four's, Gesture Analysis & Modelling. Therefore, for the W_Test and FDO dataset, data analysis is skipped, but the main outcomes of these analyses are presented for easy reference.

As a training and testing technique, the K-Fold cross validation technique is generally considered for datasets, when there is no study of this dataset in the literature. The K-Fold cross validation technique is a variant of the cross validation technique. It divides dataset into K fold and uses K-1 fold for training and the remaining fold for testing. The recognition algorithm is run K times in order to consider each fold for testing. Mean and standard deviation of the recognition results of K-folds are used as final performance merit for the recognition algorithm. In this thesis, 10-fold cross validation is considered. For example, in the case of the Gesture Panel, in order to

compare the results of RM with the existing study on this dataset [146], every dataset is used both for training and testing. [1]

In order to analyse dataset complexity, several data analysis techniques, which are discussed in the Gesture Modelling and Analysing chapter, are deployed. These techniques are entropy analysis; Chi-Square, skewness and kurtosis analysis; Fisher linear discriminant analysis; principal component-based *EROS* analysis; intersection volume analysis and temporal analysis for variance in sample length and position of sub-events.

The entropy-based information complexity technique is used to analyse the similarity between samples and class models. It also addresses channel and feature complexity or variance which directly effects the proposed recognition algorithm in terms of frame prediction and shared volume between classes. Chi-Square, skewness and kurtosis statistical analysis are applied to analyse the fitness and robustness of the parameters (μ and σ) of the assumed underlying statistical distribution (Gaussian distribution). The Fisher linear discriminant analysis addresses the inter similarity between class samples. It utilises the *within* and *between* class ratio to establish a similarity measurement. The principal component analysis-based *EROS* also implements a similarity measurement technique, which utilises the extended version of Frobenious norms (Euclidean distance) over eigenvectors and eigenvalues of a covariance matrix of classes' samples by implementing a k-nearest neighbourhood recall/precision scheme. Precision/Recall metrics in EROS accommodates a proportion of k to the volume (recall) which consists of k number of samples of class of interest. High values of precision (100 %) indicate higher disparity in the dataset. Also, the intersection volume between class models is considered as another disparity measurement. For temporal complexity analysis, the variances in length of class samples and sub-events indices are also discussed for datasets.

In this thesis, we compare the performance of the proposed recognition algorithm (RM) with Dynamic Time Warping (DTW), Elman Neural Networks (ERNN) and the Hidden Markov Model (HMM) in the case of isolated recognition. Since the recognition results of ERNN are the lowest over a real world dataset (Perrotta) and other accompanying disadvantages of ENN, (such as long training time, non-meaningful and human readable representation (black box) and output, and off-line training) ENN is not applied over other datasets. In this thesis, the main emphasis is given to HMM as an alternative recognition algorithm in experiments, due to its reported good performance in literature. In continuous gesture recognition, only HMM and RM are

[1] Although the paper [146] mentions that the leave-one-out validation technique is used, the result table indicates that all the samples in the dataset is used for both training and testing.

considered because these two recognition algorithms obtain better results in the case of isolated gesture recognition.

DTW is implemented with 0.2 Sakoe-Chiba band windowing [99]. Class models of the recognition machine (C) are used as the reference templates in DTW and input templates are stretched or compressed to have identical length with the reference templates.

The implementation of the Elman Neural Network in Matlab is used for the ENN experiments [89]. The transfer function of hidden and output layers are *tansig* and *purelin* and the network has 80 and nine neurons in the hidden and output layers respectively. The network is trained using a back propagation algorithm with an adaptive learning rate of 1.05, a momentum parameter of 0.05 and 1500 epochs. The input matrix consists of a sequence of feature column vectors, and target vectors are represented as (1,-1), 1 denoting class membership, and -1 non-membership.

The HMM algorithm is applied using the HTK toolkit [157] and the MATLAB version of the Georgia Tech Gesture Toolkit (GT^2K) [146], which is implemented by the author. Several configuration of states (20,10,5,3) and topologies such as left to right (lr), left to right one skip($lr1s$) and ergodic (er) are considered.

6.2 Datasets and their Analyses

6.2.1 Artificial Dataset - W_Test

W_Test is a parametric dataset proposed by [57]. This artificial dataset was introduced to accommodate the following criteria in order to analyse the performance of recognition algorithms:

- Multiple channels
- Temporal and Spatial Variance in the form of:
  - Periodic Variance: Variations in the total duration.
  - Horizontal Variance: Variations in the duration and timing of "sub-events" or components of the instances
  - Vertical Variance: Variance in the amplitude of the signal
  - Gaussian Noise
- Fake Signal: In addition to Gaussian noise, some irrelevant or insignificant information, which seems plausible at first sight, is added to the main signal.

The W_Test dataset has three classes (A, B, C) and each class has three channels (α, β, γ) with a 100 unit period. Figure 6.1 illustrates the prototypes of these three classes. Apart from two sub-events (frames at index of 49 and 51) of *beta* channel of class B, prototype class A and B are identical. The mathematical structures of these classes are as follows:

$$
\begin{aligned}
A_\alpha(t) &= \left\{ \begin{array}{lll} \frac{1}{35}t & \text{if} & t \leq 35 \\ 0 & \text{if} & 35 < t \leq 65 \\ \frac{1}{35}(100-t) & \text{if} & 65 < t \leq 100 \end{array} \right\} \\
A_\beta(t) &= \left\{ \begin{array}{lll} 1 & \text{if} & t \in \{25, 48, 75\} \\ -1 & \text{if} & t = 52 \\ 0 & \text{otherwise} & \end{array} \right\} \\
A_\gamma(t) &= 0
\end{aligned}
\tag{6.1}
$$

$$
\begin{aligned}
B_\alpha(t) &= \left\{ \begin{array}{lll} \frac{1}{35}t & \text{if} & t \leq 35 \\ 0 & \text{if} & 35 < t \leq 65 \\ \frac{1}{35}(100-t) & \text{if} & 65 < t \leq 100 \end{array} \right\} \\
B_\beta(t) &= \left\{ \begin{array}{lll} 1 & \text{if} & t \in \{25, 75\} \\ 0 & \text{otherwise} & \end{array} \right\} \\
B_\gamma(t) &= 0
\end{aligned}
\tag{6.2}
$$

$$
\begin{aligned}
C_\alpha(t) &= \left\{ \begin{array}{lll} \frac{1}{35}t & \text{if} & t \leq 35 \\ 0 & \text{if} & 35 < t \leq 65 \\ \frac{1}{35}(t-65) & \text{if} & 65 < t \leq 100 \end{array} \right\} \\
C_\beta(t) &= 0 \\
C_\gamma(t) &= \left\{ \begin{array}{lll} -1 & \text{if} & t = 50 \\ 0 & \text{otherwise} & \end{array} \right\}
\end{aligned}
\tag{6.3}
$$

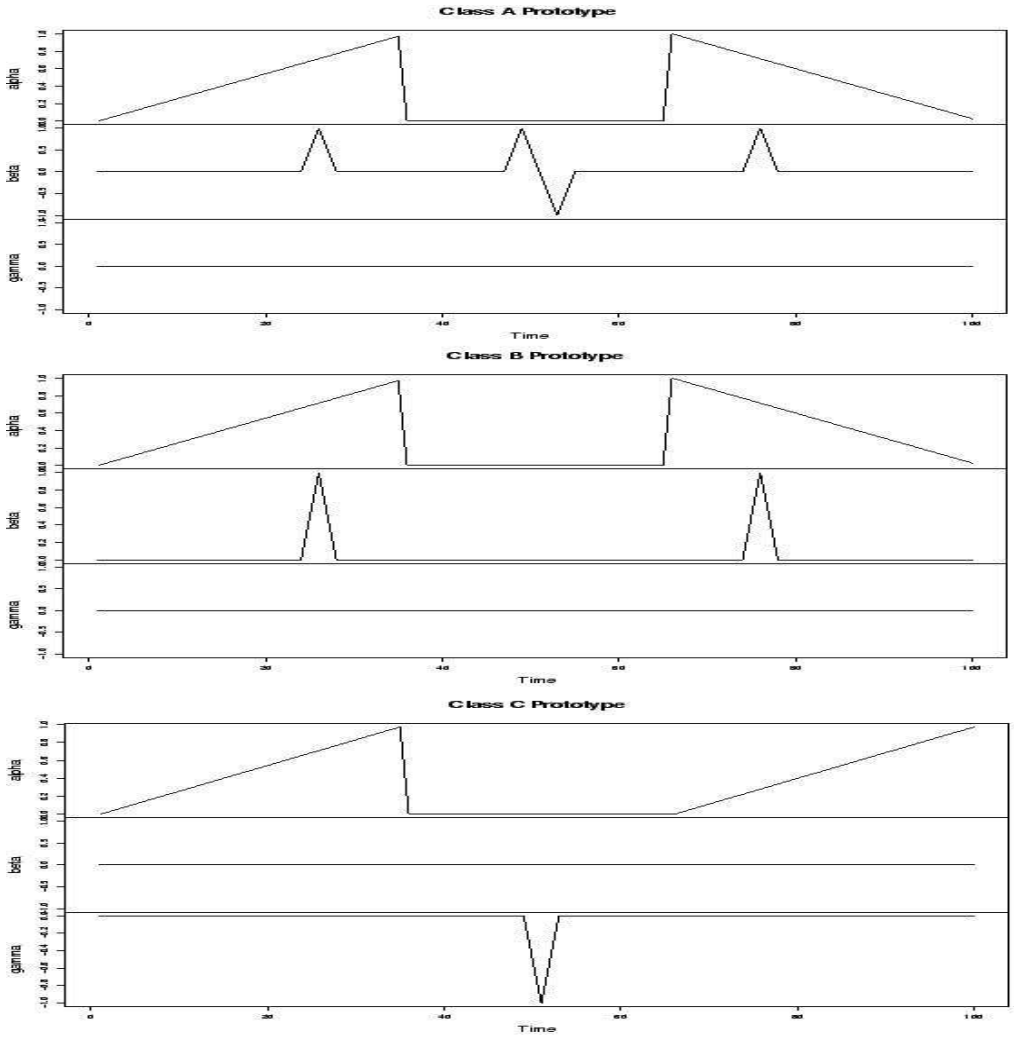


Figure 6.1: Protypes of A, B and C classes [57].

Recognition of these classes in this form is trivial as they have clear distinctive features which can be seen from figure 6.1. In order to make this dataset more challenging, the criteria expressed above are embedded into these equations. The quantity of embedded criteria is determined by control parameters with some degree of randomness. Randomness is obtained by two functions, the first of which is *unif()* and it returns a uniformly distributed real number in the interval of [-1, 1]. The second one is $\epsilon()$ which returns a real number from the unit normal distribution (N(0,1)). In the light of these criteria, the introduced control parameters with accompanying random functions are as follows:

- Temporal and Spatial Variance:
  - Periodic Variance (***d***): Prototypes of each classes' period is 100 units. The period of each sample is randomly changed by using the parameter ***d*** as follows:

$$dur \quad = \quad (1 + d * unif()) * 100 \tag{6.4}$$

This control parameter uniformly and linearly stretches or compresses the length of samples. Figure 6.2 depicts the effects of ***d*** on some samples of class A.

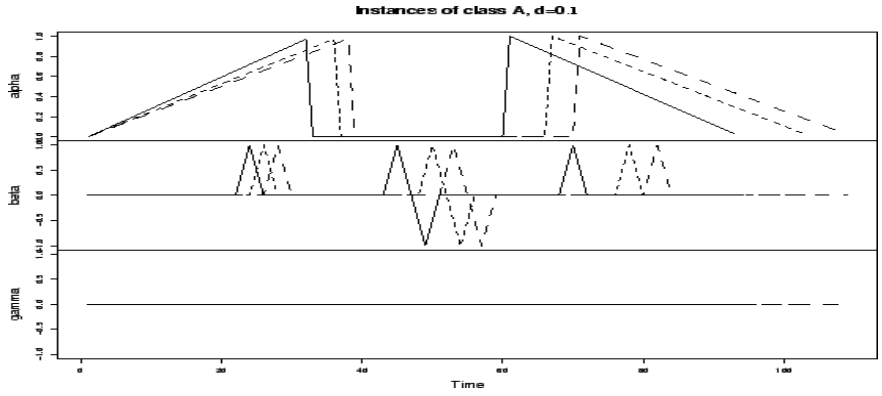


Figure 6.2: Effect of Periodic Variance Parameter (d) on prototype class A (d=0.1) [57].

  - Sub-Event Variance ***(c)***: Hence, periodic variance ***(d)*** only deploys a uniform linear temporal variance, a non-linear and non-uniform temporal variance can be obtained via changing the duration and timing of *sub-events*. The variation in the duration and index of these sub-events are controlled

by the parameter $\boldsymbol{c}$ $(c * unif())$ with a degree of randomness. Figure 6.3 illustrates the effects of $\boldsymbol{c}$ over some samples of class A.

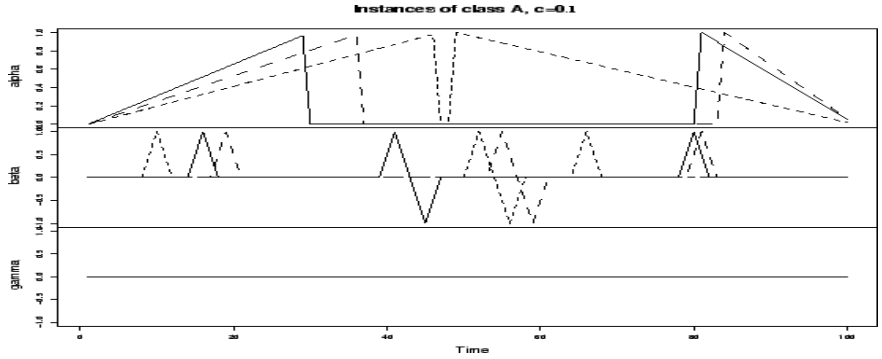


Figure 6.3: Effects of Sub-events Variance Parameter (c) on prototype class A (c=0.1) [57].

– Vertical Variance ***(h)***: In real life situations, in addition to sub-events variation, vertical variance, in other words, amplitude variance can also emerge on sub-events and on the other frames. That situation is integrated into the dataset with the parameter $\boldsymbol{h}$ with a randomness $(h * unif())$. The effect of vertical variance on some samples of class A is demonstrated in figure 6.4.

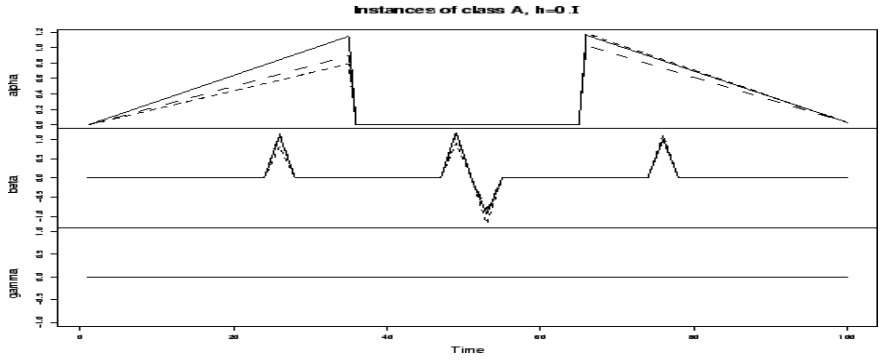


Figure 6.4: Effect of Vertical Variance Parameter (h) on prototype class A (h=0.1) [57].

– Gaussian Noise ***(g)***: In order to have a realistic and challenging dataset, Gaussian noise is added to all the channels. The amount of noise is controlled by parameter $\boldsymbol{g}$ as $(g * \epsilon())$. Figure 6.5 illustrates the effects of parameter $\boldsymbol{g}$ on some samples of A.

- Irrelevant Signal (*irrel*): Finally, a real world dataset that looks useful and plausible but in fact it is irrelevant. It does not convey any information. In order to

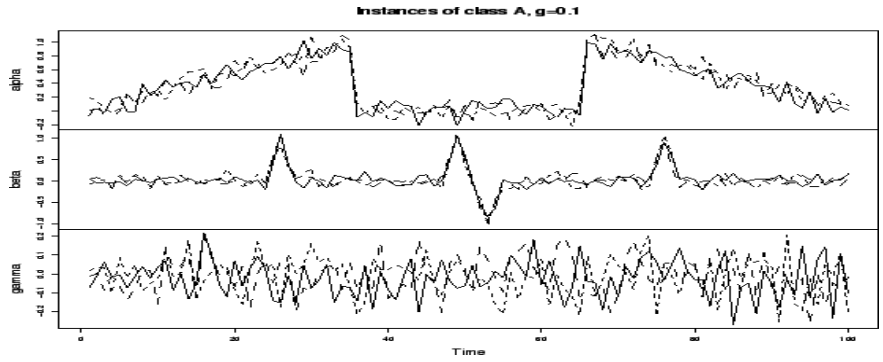


Figure 6.5: Effect of Gaussian Noise Parameter (g) on prototype class A (g=0.1) [57].

accommodate that phenomena, γ channel of classes A, B and β channel of class C are replaced with some random signals, which are sequential random line segments. The number of random line segments in each channel changes from two to nine. Note that the transition from one line segment to the next is smooth. In other words irrelevant signal does not accommodate jumps from a random line segment to another. Figure 6.6 illustrates some instances of class A with fake signals. Bear in mind, in all the W_Test related experiments in this study, the *irrel* parameter is always turned *on*.

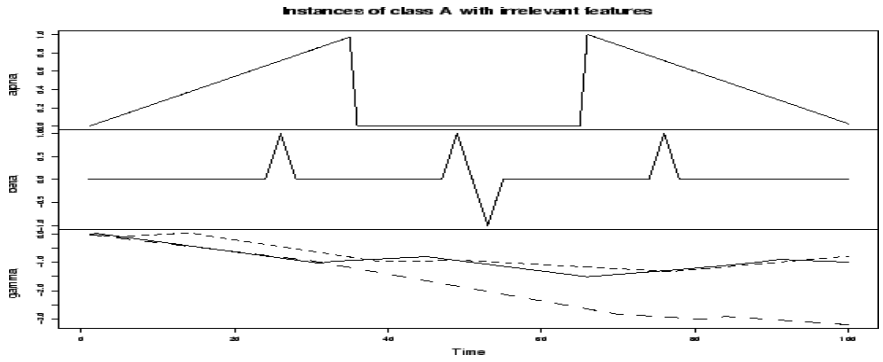


Figure 6.6: Effect of fake signals on the gamma channel of class A. [57].

Figure 6.7 illustrates some samples of class A, after being modified by these parameters (d=0.1, c=0.1, h=0.1 and g=0.1). Consequently, in view of these modifications, new mathematical definitions of the classes are as follows:

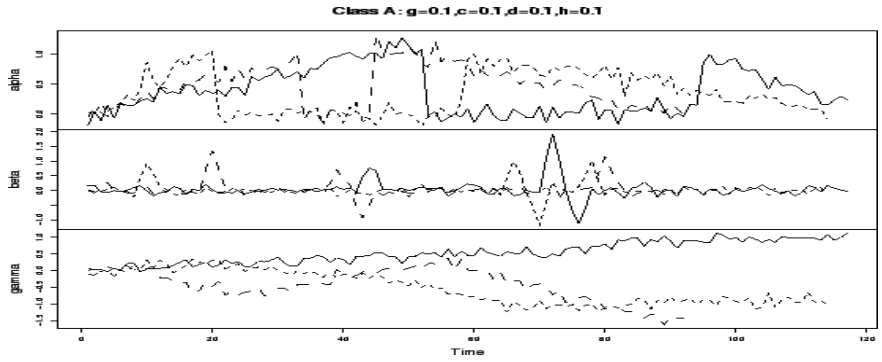


Figure 6.7: Some samples of class A after being modified by parameters d=0.1, c=0.1, h=0.1 and g=0.1 [57].

$$
\begin{aligned}
dur &= (1 + d * unif()) * 100 \\
t_{\alpha 1} &= (0.35 + c * unif()) * dur \\
t_{\alpha 2} &= (0.65 + c * unif()) * dur \\
h_{\alpha 1} &= (1 + h * unif()) \\
h_{\alpha 2} &= (1 + h * unif()) \\
A_\alpha(t) &= \left\{ \begin{array}{lll} \frac{h_{\alpha 1}}{t_{\alpha 1}} * t + g * \epsilon(t) & \text{if} & t \leq 35 \\ g * \epsilon(t) & \text{if} & t_{\alpha 1} < t \leq t_{\alpha 2} \\ \frac{dur - t}{dur - t_{\alpha 2}} * t + g * \epsilon(t) & \text{if} & t_{\alpha 2} < t \leq dur \end{array} \right\} \\
\\
t_{\beta 1} &= (0.25 + c * unif()) * dur \\
t_{\beta 23} &= (0.5 + c * unif()) * dur \\
t_{\beta 4} &= (0.75 + c * unif()) * dur \\
h_{\beta 1} &= (1 + h * unif()) \\
h_{\beta 2} &= (1 + h * unif()) \\
h_{\beta 3} &= (-1 + h * unif()) \\
h_{\beta 4} &= (1 + h * unif()) \\
A_\beta(t) &= \left\{ \begin{array}{lll} h_{\beta 1} + g * \epsilon(t) & \text{if} & t = t_{\beta 1} \\ h_{\beta 2} + g * \epsilon(t) & \text{if} & t = t_{\beta 23} - 1 \\ h_{\beta 3} + g * \epsilon(t) & \text{if} & t = t_{\beta 23} + 1 \\ h_{\beta 4} + g * \epsilon(t) & \text{if} & t = t_{\beta 4} \\ g * \epsilon(t) & \text{otherwise} & \end{array} \right\} \\
\\
A_\gamma(t) &= \left\{ \begin{array}{lll} g * \epsilon(t) & \text{if} & irrel\ is\ off \\ irrel(t) + g * \epsilon(t) & \text{if} & irrel\ is\ on \end{array} \right\}
\end{aligned}
\tag{6.5}
$$

$$
\begin{aligned}
dur &= (1 + d * unif()) * 100 \\
t_{\alpha 1} &= (0.35 + c * unif()) * dur \\
t_{\alpha 2} &= (0.65 + c * unif()) * dur \\
h_{\alpha 1} &= (1 + h * unif()) \\
h_{\alpha 2} &= (1 + h * unif()) \\
B\alpha(t) &= \begin{cases} \frac{h_{\alpha 1}}{t_{\alpha 1}} * t + g * \epsilon(t) & \text{if } t < t_{\alpha 1} \\ g * \epsilon(t) & \text{if } t_{\alpha 1} \leq t < t_{\alpha 2} \\ \frac{dur - t}{dur - t_{\alpha 2}} * t + g * \epsilon(t) & \text{if } t_{\alpha 2} \leq t \leq dur \end{cases} \\
t_{\beta 1} &= (0.25 + c * unif()) * dur \\
t_{\beta 23} &= (0.5 + c * unif()) * dur \\
t_{\beta 4} &= (0.75 + c * unif()) * dur \\
h_{\beta 1} &= (1 + h * unif()) \\
h_{\beta 2} &= (1 + h * unif()) \\
B\beta(t) &= \begin{cases} h_{\beta 1} + g * \epsilon(t) & \text{if } t = t_{\beta 1} \\ h_{\beta 2} + g * \epsilon(t) & \text{if } t = t_{\beta 2} \\ g * \epsilon(t) & \text{otherwise} \end{cases} \\
B\gamma(t) &= \begin{cases} g * \epsilon(t) & \text{if } \textit{irrel} \text{ is off} \\ irrel(t) + g * \epsilon(t) & \text{if } \textit{irrel} \text{ is on} \end{cases}
\end{aligned}
$$

$$
\begin{aligned}
dur &= (1 + d * unif()) * 100 \\
t_{\alpha 1} &= (0.35 + c * unif()) * dur \\
t_{\alpha 2} &= (0.65 + c * unif()) * dur \\
h_{\alpha 1} &= (1 + h * unif()) \\
h_{\alpha 2} &= (1 + h * unif()) \\
C\alpha(t) &= \begin{cases} \frac{h_{\alpha 1}}{t_{\alpha 1}} * t + g * \epsilon(t) & \text{if } t < t_{\alpha 1} \\ g * \epsilon(t) & \text{if } t_{\alpha 1} \leq t < t_{\alpha 2} \\ \frac{t - t_{\alpha 2}}{dur - t_{\alpha 2}} * t + g * \epsilon(t) & \text{if } t_{\alpha 2} \leq t \leq dur \end{cases} \\
C\beta(t) &= \begin{cases} g * \epsilon(t) & \text{if } \textit{irrel} \text{ is off} \\ irrel(t) + g * \epsilon(t) & \text{if } \textit{irrel} \text{ is on} \end{cases} \\
t_{\gamma} &= (0.25 + c * unif()) * dur \\
h_{\gamma} &= (-1 + h * unif()) \\
C\gamma(t) &= \begin{cases} h_{\gamma} + g * \epsilon(t) & \text{if } t = t_{\gamma} \\ g * \epsilon(t) & \text{otherwise} \end{cases}
\end{aligned}
$$

In this study various combinations of control parameters between 0 and 0.2 have been tested apart from the sub-event variance parameter (c). The sub-events parameter c is limited between 0 and 1, because in the case of a higher value of c, the time order of sub-events changes in the channels. Therefore, completely different signal behaviours are produced according to the prototype definitions. In this study, for the data complexity analysis part and the comparison with other recognition algorithms and datasets, two values of noise level g=0.1 and g=0.2 are relevant, while other parameters are especially set to (d=h=0.2, c=0.1 and *irrel=on*). These two cases are referred to as W_Test1 and W_Test2 in the rest of the thesis. For each combination of parameters in the experiments (W_Test1 and W_Test2), 1000 samples are created for each class.

Raw channel data with its fuzzy gradient feature is used as the feature set without any smoothing operation. Thus, the feature vector of the W_Test is as follow:

$$F_{W\_Test} = [\alpha,\ \beta,\ \gamma, \alpha',\ \beta',\ \gamma']$$

Since the W_Test is used as the main example during the elaboration of the dataset complexity analysis in the chapter Modelling and Analysis, and in order to avoid any repetition, a detailed complexity analysis of this dataset is skipped. But, in brief, the main outcome of this analysis is pointed out here: Due to definitions of Class A and B, these two classes have high inter similarity. The only differences between class A and B are the two sub-events in the β channel at frames 49 and 51. These distinctive two sub-events also appear in the β channel of class B or disappear in the β channel of class B, when the control parameters are kept high. For example, when the parameter is set (d=h=g=0.2 and c=0.1), it is observed that about 7-8% of class A and B samples are inseparable. Figure 6.8 illustrates the β channel of some of these inseparable samples from class A and B, when the parameter is set (d=h=g=0.2, c=0.1 and *irrel=on*).

The entropy analysis in chapter four on the W_Test reveals that higher values of control parameters increase all entropies (channel, frame and class). Consequently, it leads to a higher noise signal ratio (NSR) and lower mutual information precision (MIP) for higher values of control parameters, which makes the task challenging for recognition algorithms. Since Gaussian noise is used during the construction of samples it is skipped to the fitness analysis of statistical distribution with the chi-squared test, skewness and kurtosis. The Fisher linear discriminant analysis points out the frame-based (sub-events) disparity between classes. For example, in the case of g=0.2 and c=d=h=0 and with the *irrel on*, discriminant sub-events at the indices of 49 and 51 in the *beta* channel of class A and B are revealed (Figure, 4.27-top left). But, in the case

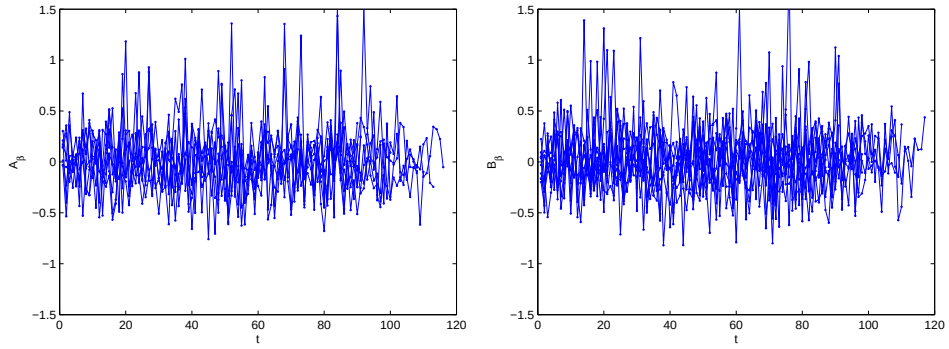


Figure 6.8: Some similar samples of the β channel of class A (left) and B (right) in dataset when parameters are d=0.2, c=0.1, h=0.2 and g=0.2. The beta channel is the only distinctive component between class A and B. In the case of high noise and amplitude, this distinctive channel in some cases disappears. Therefore, classification of these two classes is non-trivial.

of higher values of control parameters, g=d=h=0.2, c=0.1 and with *irrel on*, the Fisher linear discriminant analysis fails to reveal these sub-events (Figure, 4.27-button left). The PCA-based EROS and intersection based similarity techniques point out a high similarity between class A and B in the case of both low and high values of control parameters (Figure 4.30, tables 4.12 and 4.13). But on the other hand, class C is more distinctive than other classes in the case of lower and higher control parameter values. Since the parameter d and c correspond to the period variance percentage (PVP) and sub-event variance percentage (SEVP) respectively, the periodical and index variance analysis of the W_Test has been omitted.

6.2.2 Gestures for Interaction in VE - Yang Gestures [64, 63]

Yang dataset is an isolated real world dataset. It is a part of full body gesture dataset comprising over 40 body motions for a virtual environment application [64, 63]. The gesture set consists of eight hand gestures. Figure 6.9 illustrates these gestures. A tracker-based input device is used for data collection. Each gesture is represented by three coordinates (x, y, z) at a given time. The dataset accommodates approximately 100 samples for each gesture. Gestures are modelled using the following features: smoothed 3D Cartesian coordinate positions, their gradients and angular velocity. Therefore, the feature set is as follows:

$$F_{Yang} = [x,\ y,\ z, x',\ y',\ z', a]$$

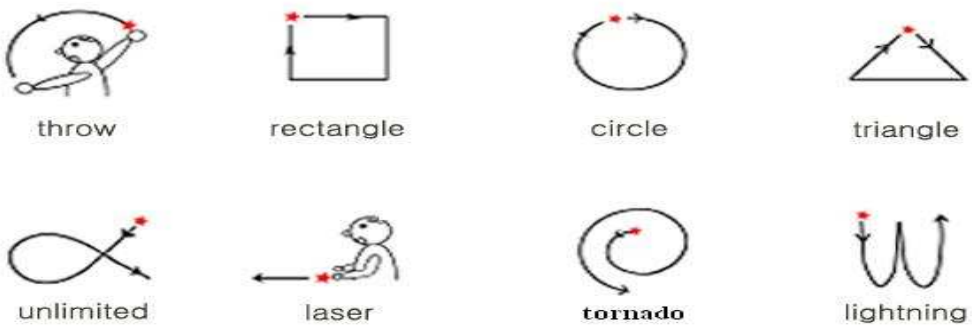


Figure 6.9: Yang Gestures.

where a is the angular velocity of the coordinate positions and calculated as follows:

$$\begin{aligned} X'(t) &= X(t) - X(t-1); \\ X'(t+1) &= X(t+1) - X(t); \\ norm(X') &= norm(X'_x, X'_y, X'_z) \\ &= \sqrt{X'^2_x + X'^2_y + X'^2_z} \\ a &= acos(\frac{X(t) * X(t+1)^T}{norm(X(t)) * norm(X(t+1))}) \end{aligned} \tag{6.6}$$

where the norm function corresponds to the length of vector X which consists the 3D Cartesian coordinates (x,y,z).

The angular velocity channel is taken as the base for sub-events alignment because it contains better sub-events for all gestures apart from *Laser*, which does not contains any sub-events due to its horizontal trajectory. Other channels are aligned according to the sub-events of the angular velocity channel, unlike, the W_Test dataset. The positions of sub-events are not independent of the channels.

The quality of the dataset is very poor. For example, it is observed that, while the definition of *Laser* gestures indicates a straight line, some samples of the *Laser* class have very curvy (similar to a step function) signals. In addition, samples have high noise at the initial and final stages.

Due to the geometric shape of *circle*, *rectangle* and *triangle* gestures in figure 6.9, at first sight, intuitively, it can be guessed that there will be a remarkable resemblance between these gestures. In addition, it is expected that due to some reason, the *Laser* gesture is more distinctive than other gestures. In fact, the complexity and similarity analysis in the following paragraphs on the dataset supports this initial observation.

Figure 6.10 illustrates the channel, frame, class and cross mutual entropy for the

	$\bar{hC}$	$h\bar{H}X$	$h\bar{V}X$	$I(hHX, hC)$	$\bar{MIP}$	$\bar{NSR}$
Mean	0.52	0.55	0.58	0.25	0.14	0.55
Std	0.24	0.19	0.12	0.02	0.20	0.06
Min	0.00	0.15	0.28	0.00	0.00	0.44
Max	0.95	0.39	0.77	0.49	0.62	0.62

Table 6.1: Entropy Characteristics of the dataset Yang

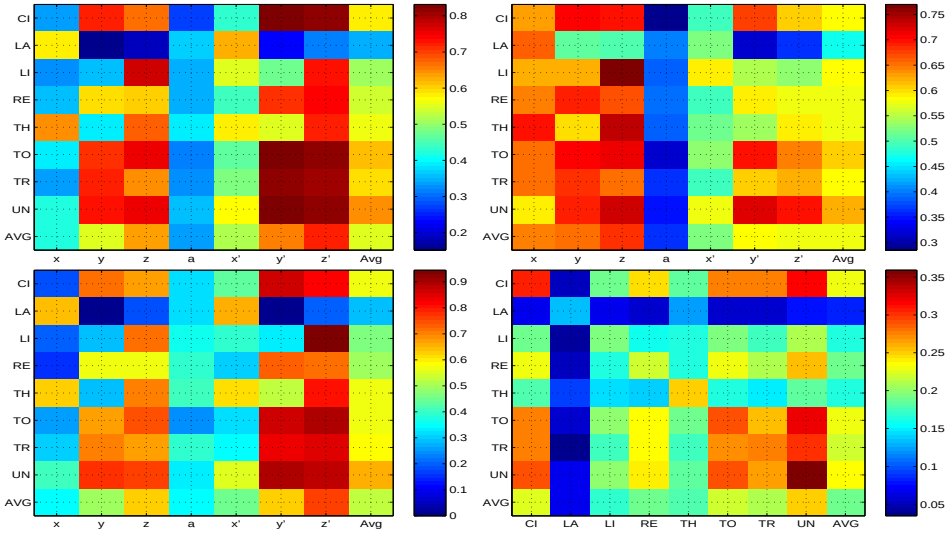


Figure 6.10: Entropy Analysis of Yang Gestures: Channel (top-left), Frame (top-right), Class (bottom-Left) and Cross Mutual Entropy (bottom-right)

Yang dataset. Class, channel and frame entropy indicate low entropy for the channel a angular velocity. In other words, the angular velocity channel has narrow variance at the index point. On the other hand, other channels have higher entropies, which is an indicator of high similarity and complexity in the dataset. Cross mutual entropy analysis supports the initial guess mentioned above. *Laser* gesture is more distinctive. In fact, because of the noise in this gesture's samples, the gesture has also more disparity between the class's models and its samples.

Table 6.1 summarizes the entropy analysis for the Yang dataset with additional entropy measurements. The dataset has a high noise signal ratio (NSR) and low mutual information between the class and its samples. $(I(hH\bar{X}, hC))$ and low mutual information precision MPI, which is a summary indicator of disparity between the class's models and their samples.

Figure 6.11 shows the results of the parameter fitting analysis using skewness, kurtosis and the Chi-Squared Test. Note that, the skewness and kurtosis of a Gaussian distribution are 0 and 3 respectively and the Chi-Squared Test figures show fitness of

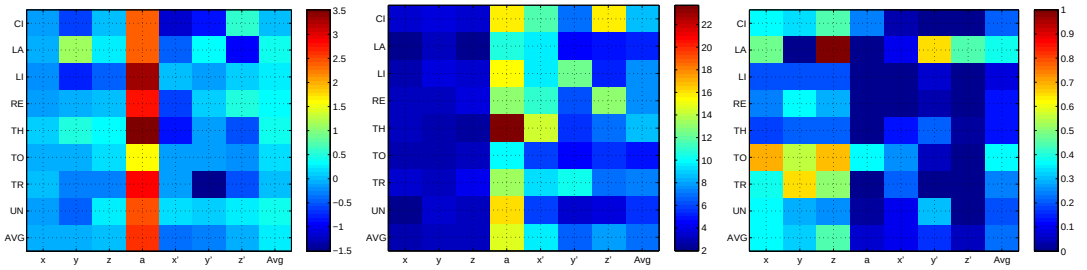


Figure 6.11: Statistical distribution parameter fitting analysis with Skewness (left) and Kurtosis (middle) and Chi-Squared Test (right) for the Yang dataset. For the Yang dataset it is proved that H_0 or in other words, the underlying statistical distribution is Gaussian, is correct. Most spatial channel and temporal channels (x', y', z' and a) do not fully support this assumption.

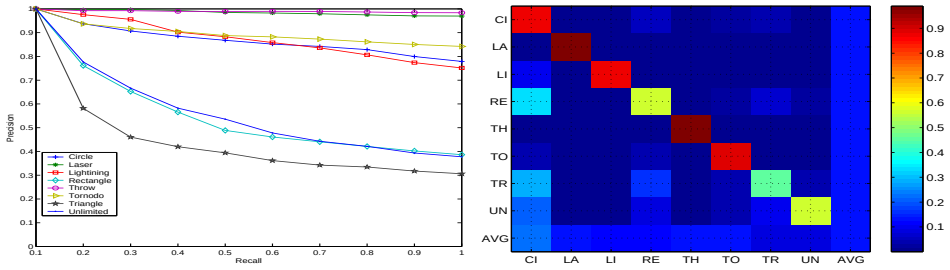


Figure 6.12: Inter (left) and average cross (right) recall/precision of Yang datasets using EROS which implements k-Nearest Neighbourhood algorithm (kNN, $k = \{1, 2, 3 \cdots 10\}$, recall $(0.1, 0.2, 0.3 \cdots 1)$) over the samples of the classes which are transformed into the PCA-based matrices. Average cross similarity figure (right) shows cross precision/recall rate among samples of row classes to column classes.

channel parameters using the ratio of frames numbers, which supports this assumption, to total number of frames (in other words to the period of class). For a detailed discussion refer to chapter four, Modelling and Analysis. For the Yang dataset, during construction of the class models, it is assumed that the underlying statistical distribution at each index of channels is Gaussian. Skewness, kurtosis and the Chi-Squared Test proved this assumption to be true for the spatial channel (x, y, and z). The temporal fuzzy gradient channels (x', y', z') and the angular velocity channel(a) do not support this assumption. This can be attributed to channel construction where, gradient values are out of a bandwidth that are truncated to 1 and -1.

Figure 6.12 illustrates the similarity among Yang dataset using PCA-based EROS similarity measurement in terms of precision (p) for recall (r) values. EROS algorithm implements k-Nearest Neighbourhood algorithm. Precision rate simply indicates how dense the samples of a class of interest are clustered in a neighbourhood of k ($k = 1, 2, 3, \cdots 10$) in feature space. This also represents the number of total samples (k) in

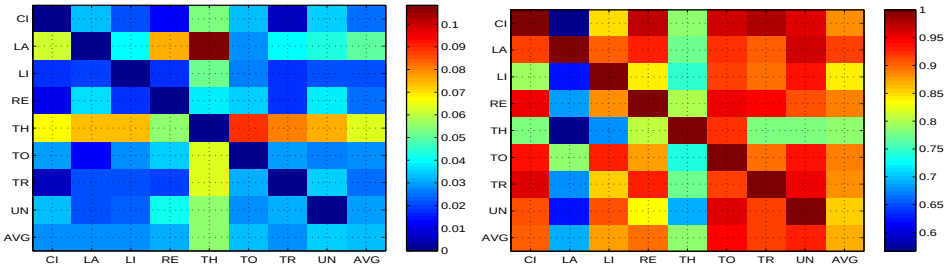


Figure 6.13: Fisher linear discriminant (left) and intersection similarity (right) analysis for the Yang dataset. These analyses obtain results in agreement with previous EROS and Entropy analyses.

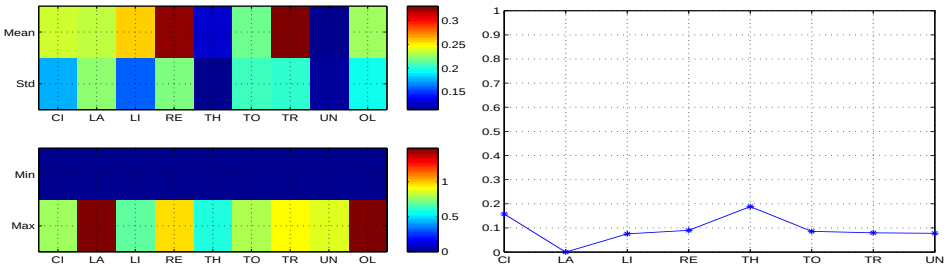


Figure 6.14: Periodical and sub-event variance for the Yang dataset.

the neighbourhood. In EROS, k/10 is referred as recall value (r= $r = 0.1, 0.2, 0.3 \ldots 1$). For example, in a $k = 7$ neighbourhood ($r = 0.7$), if the number of samples of a class of interest is 3, then the precision rate will be $p = 3/7$. A higher value of precision (p) indicates a higher disparity (dense clustering) for the recall value and class of interest. In other words, any two gestures that are similar will have lower values of precision. If recall value is increased, in other words, the volume of the neighbourhood, the precision value will decrease, as the increased neighbourhood would largely consist of samples from other classes in the feature space. For example, the potential similarity between *rectangle* and *triangle* becomes obvious if r is increased (greater than 0.2). But on the other hand, disparity of some gestures such as *throw*, *laser* are preserved for all r, which once again shows the distinguishing characteristic of these gestures among all other gestures. The figure also illustrates the shared features of *rectangle*, *circle* and *triangle*.

The Fisher linear discriminant analysis and intersection analysis obtain similar results regarding these gestures. Results of these analyses are illustrated in figure 6.13. Note that in an intersection similarity graph, while the last column indicates the average subset power of row class (C_i), the last row indicates the average encapsulation power of column class (C_j). For example, the intersection analysis points out that

Throw gestures are performed in a different subset of feature space. But, on the other hand, unlike the *Throw* gesture, while *Laser* has a greater subset degree, its encapsulation power is low. In other words, the volume space, where the *Laser* gesture is performed, is commonly shared by other gestures. Note that this situation is in agreement with the cross mutual entropy analysis illustrated in figure 6.10, which shows high scattering between the *Laser* class model and its samples. Yet, the EROS analysis indicates a contrary situation showing that the *Laser* gesture is one of the most distinctive gestures in the set. This situation can be attributed to the shape of the *Laser* (straight line) gesture.

The outcome of the periodical variance and sub-event variance analysis are depicted in figure the 6.14. *Triangle* and *Rectangle* have the highest 0.3 periodic variance percentage (PVP), which makes the classification task harder between these two gestures. A sub-event analysis is carried out on the angular velocity channel (a), which contains most meaningful sub-events. SEVP is not applied over *Laser* gesture because it does not contain any sub-events. The *Throw* and *Cycle* gestures contain highest SEVP.

The results of the dataset complexity and similarity analysis for the Yang dataset can be summarised as follows: While the *Laser* and *Throw* gestures are distinctive, the Rectangle and Triangle gestures have high similarity. Gaussian, as the underlying statistical distribution, is suitable for the gestures for most of the channels.

6.2.3 Gestures for Interaction in VE - Perrotta [89]

Perrotta collected a hand gesture dataset consisting of nine hand gestures similar to Yang [89]. Figure 6.15 illustrates the datasets. The difference between the Yang dataset and the Perrotta dataset is that for the data collection of the Perrotta dataset, three different (2 male, 1 female) users are deployed for gesture performing.

For collection of data, similar to FDO_PT, tracker-based Polhemus FasTrak is used. But in this case, unlike the FDO_PT dataset, only one sensor is used to acquire the 3D Cartesian coordinate and orientation angles. Reported limitations regarding Polhemus FasTrak are not reported because, unlike the FDO_PT dataset, Perrotta gestures are performed in a smaller volume. The pre-processing task involves off-line smoothing and normalization. The samples are normalized in the range of [-1,1] in an off-line mode, in respect of the global maximum and minimum point of the cycles. In addition, in order to reduce spatial variance, trajectories of gestures are shifted in a way that the starting point of each cycle is 0.

The dataset consists of approximately 28 samples of each class from three people. The first, second and third (female) users collected 14, 7 and 7 gestures respectively.

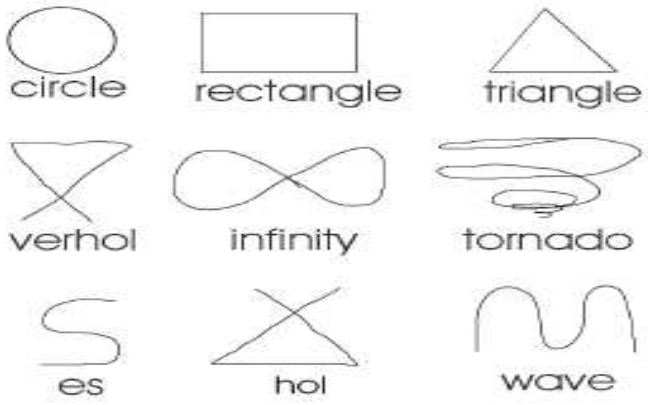


Figure 6.15: Perrotta Gestures

While 16 of those samples are used for training, the rest are used for testing.

Perrotta conducted experiments with three different feature sets (raw, shifted and extended) and achieved different success rates. In the shifted feature set, the starting point of each raw gesture is shifted to zero point, in order to reduce spatial variances. In this study, the most successful feature set (extended) by ENN (75% recognition rate) is considered. The extended feature set consists of 8 features: 3D coordinates (x, y, z), corresponding to orientation angles (yaw, pitch, roll) and curvature (k) and torsion (π). Similar to the shifted feature set, the start point of the gestures are shifted to zero in the extended feature case too. Features, curvature (k) and torsion (π) in the extended feature set are estimated as follows [72]:

$$k = \frac{|x' \times x''|}{|x'|^3} \tag{6.7}$$

where x' and x" correspond to the first and second derivative of curve $x(t)$ respectively and torsion (π):

$$\pi = \frac{|x' \times x'' \times x'''|}{|x' \times x''|^2} \tag{6.8}$$

where $|x' \times x'' \times x'''|$ indicates the triple scalar product and is calculated as $x' \times (x'' \times x''')$ [72].

During the class model construction phase, it is assumed that the channels do not have any sub-events. This assumption is carried out in order to use identical data

which are used by the ENN recognition algorithm in the study of Perrotta [89].

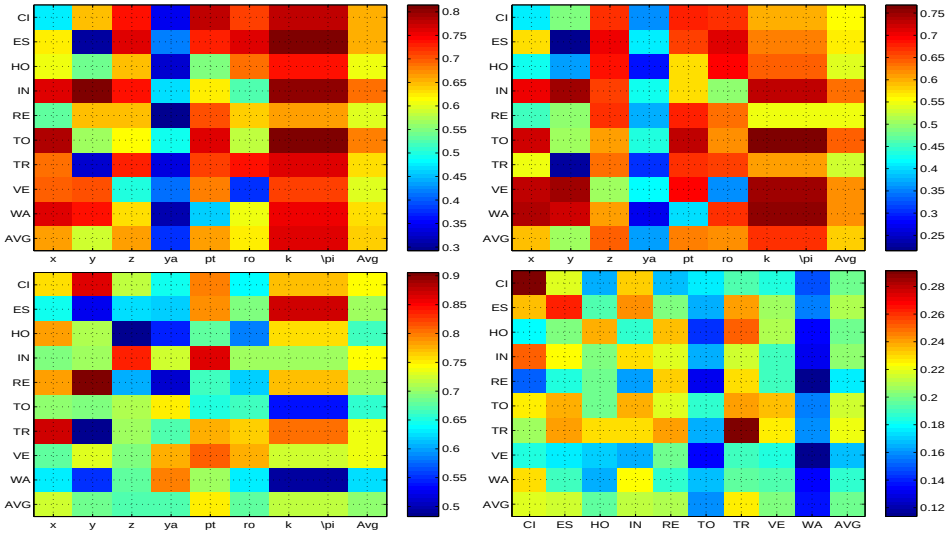


Figure 6.16: Entropy Analysis of Perrotta Gestures: Channel (top-left), Frame (top-right), Class (bottom-Left) and Cross Mutual Entropy (bottom-right)

In terms of the complexity and similarity analysis, the Perrotta dataset has similar characteristics to the Yang dataset. But, since the Perrotta dataset was constructed using three users, it has more spatial and temporal variance, which is observable in every complexity and similarity analysis.

Figure 6.16 and table 6.2 summarise the entropy analysis on the Perrotta dataset. Variance caused by multiple users is observed as a higher class, channel and frame entropy. Channels, z, k and *pi* have, in particular, high class, channel and frame entropy. Cross mutual information between class models and their samples is very low compared to the Yang dataset. *Circle* gestures accompany the highest mutual information among its samples in the dataset, even though it is very low (0.28). The *Wave* and *Tornodo* gestures have the lowest mutual information among its samples. In addition, the *Rectangle* and *Triangle* gesture models encapsulate high mutual information with other gestures. Table 6.2 also indicates a higher noise signal ratio (NSR) and mutual information precision (MIP) in the dataset.

	$\bar{hC}$	$\bar{hHX}$	$\bar{hVX}$	$I(hHX, hC)$	$\bar{MIP}$	$\bar{NSR}$
Mean	0.70	0.64	0.58	0.23	0.18	0.64
Std	0.09	0.15	0.15	0.04	0.10	0.08
Min	0.48	0.29	0.22	0.03	0.01	0.54
Max	0.91	0.81	0.77	0.46	0.32	0.79

Table 6.2: Summary Entropy of the Perrotta dataset

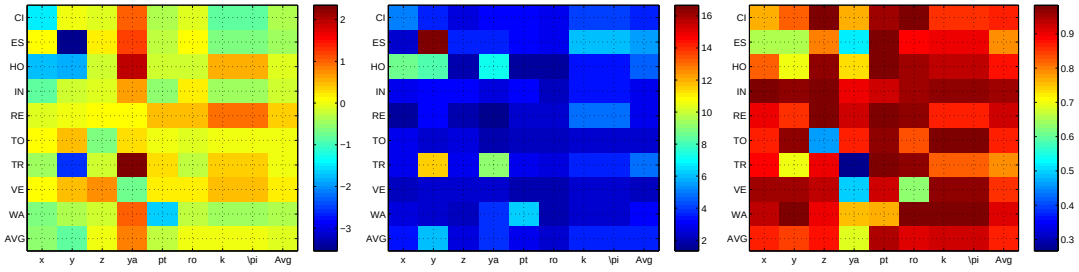


Figure 6.17: Statistical distribution parameter fitting analysis with Skewness (left) and Kurtosis (middle) and Chi-Squared Test (right) for the Perrotta dataset.

Figure 6.17 illustrates the channel-based statistical parameter fitting analysis of Perrotta gestures by skewness, kurtosis and chi-squared test techniques. These sets support the assumption for most of the channels (apart from the yaw channel) and gestures, that the assumed underlying statistical distribution is Gaussian. But, it has been observed that in the case of the spatial channel of *Es* and *Hol* gestures, the underlying distribution is not Guassian.

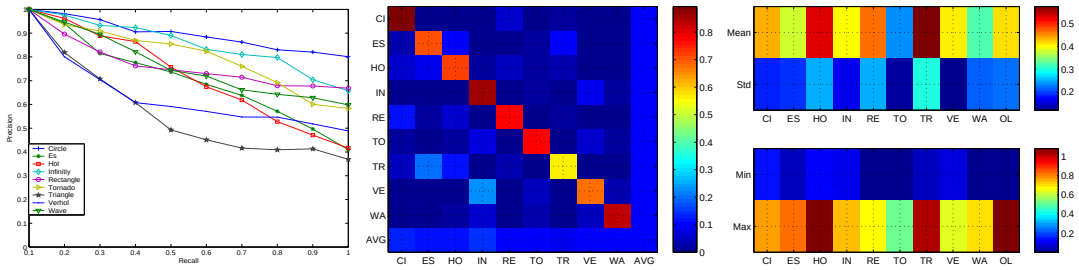


Figure 6.18: Inter (left) and average cross (middle) recall/precision of the Perrotta dataset using EROS. The right figure shows the periodic variance percentage (PVP).

The results of the EROS and periodic variance (PVP) analysis for the Perrotta dataset are depicted in figure 6.18. Both analyses reveal that the Perrotta dataset has the higher periodic variance and inter class similarity compared to the Yang dataset. The PVP analysis, in particular, points out the remarkably high temporal variance (average $\tilde{0}.4$) in the dataset. The *Triangle*, *Hol* and *Rectangle* gestures have the highest PVP in the dataset. The *Circle* gesture has the highest disparity in the EROS analysis.

Figure 6.19 illustrates the Fisher linear discriminant and intersection similarity analysis of the Perrotta dataset. The Fisher linear discriminant analysis shows that the *Circle* gesture is the most distinctive gesture and that the *Infinite* and *Tornado* gestures have the lowest distinction. In addition, the distinction degree between the *Triangle*

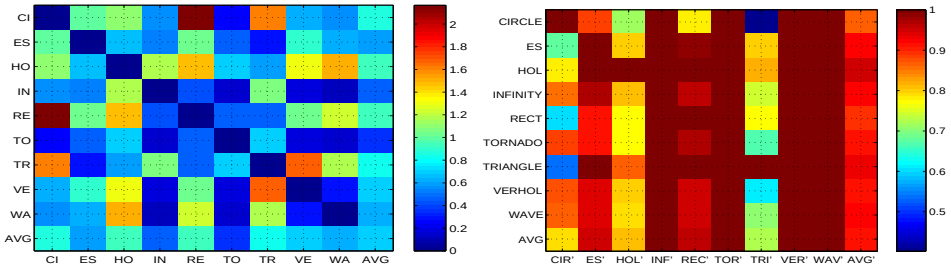


Figure 6.19: Inter class similarity of the Perrotta dataset using the Fisher Linear Discriminant (right) and Intersection Similarity (left) schemes.

and *Rectangle* gestures are remarkable low. The intersection similarity analysis also reveals that the `Circle` gesture has the lowest encapsulate and subset power. In other words, it is very densely clustered around a point in the feature space. Therefore, it is quite distinctive from the others. On the other hand, the *Triangle* and *Hol* gestures, have a low encapsulation rate but a high subset rate. They do not have much disparity. In the cases of *Wave*, *Verhol*, *Infinity* and *Tornoto*, because of both the high encapsulation and subset rates, disparity is lowest. These classes overlap largely in the feature space. Please note that these results are in agreement with the entropy (especially frame entropy), Fisher linear discriminant and EROS analyses, in which, while the *Circle* has the highest disparity, the *Triangle*, *Hol*, and *Es* gestures have the lowest disparity.

As a conclusion of the complexity and similarity analysis of the Perrotta dataset, the following can be listed: The dataset consists of high variance. The *Circle* gesture is the most distinctive in the dataset due to features (k and π) introduced to reduce the possible intra/inter similarity between the *Rectangle*, *Triangle* and *Circle* gestures, as was reported above in the Yang gesture. But these features fail to distinguish between the *Rectangle* and *Triangle* gestures in most analyses. The *Triangle*, *Hol*, and *Es* have lower disparity power. Finally, the dataset has remarkable high temporal variance (periodic variance).

6.2.4 Gesture Panel [146]

The Gesture Panel (GestPan) dataset is provided in the tutorial section of the Georgia Tech Gesture Toolkit (GT^2K) distribution [146]. (GT^2K) utilises HTK (Hidden Markov Toolkit for Speech Recognition [157]) binaries for gesture recognition applications. GestPan consist of eight gestures in the context of controlling the radio in a car by hand gestures. These gestures are used to minimize the distraction when controlling

the radio in a car when driving. Figure 6.20 illustrates the setting of the car cabin. Due to light conditions, instead of using a normal camera, Gesture Panel deploys a mono white/black camera which emits light from a 8×9 the grid of infra-red light (IR LEDs). Gestures are performed between the camera and grid panel. In order to minimize the distraction, gestures are kept simple such as a hand sweeping in one of eight directions: *Down*, *DownLeft*, *DownRight*, *Left*, *Right*, *Up*, *UpLeft*, *UpRight*.

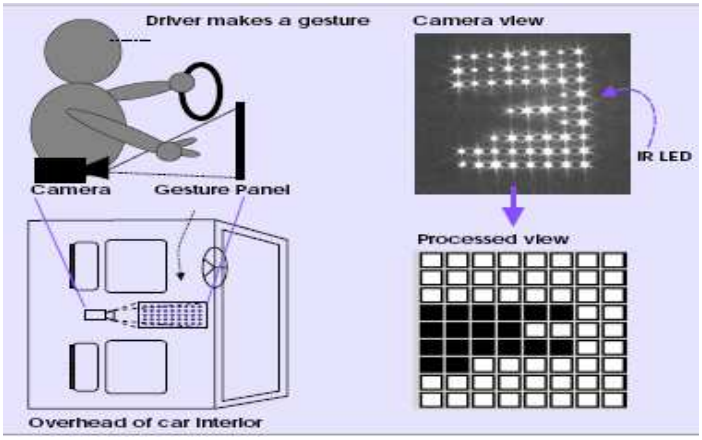


Figure 6.20: Gesture Panel in a vehicle. Interior design of car and data acquisition setting (right). Camera acquires hand configuration as gesture binary image (left) [146]. Eight simple gestures, each of which sweep in one of the eight directions, are defined.

Gestures are represented as binary images where 1 shows occlusion. In the study [146], binary images are represented in the form of a vector rather than a matrix. In order to be compatible with the original study, the same feature set is used in this study. Therefore, the feature vector for the GestPan dataset contains 72 binary digits for a time point. This characteristic of the GestPan dataset, namely, binary discreet data, is the primary reason for considering this dataset for validation of the proposed recognition algorithm. Since features correspond to pixels (cells) in images, in a way, dataset complexity and similarity analyses can be considered as cell or pixel-based. Therefore, for example, in the entropy analysis, the complexity and similarity of features addresses the complexity and similarity of the cells. Note that since the feature set, image itself, is not an optimal orthogonal set, the data complexity and similarity analysis can contain errors. Furthermore, it can be meaningless according to some analysis.

Figure 6.21 illustrates the entropy results of the GestPan. Class, channel and frame entropy analyses show low entropy for the all channels. The *DownRight* (DR) gestures, in particular, has the lowest class, channel and frame entropy. In addition, the cross

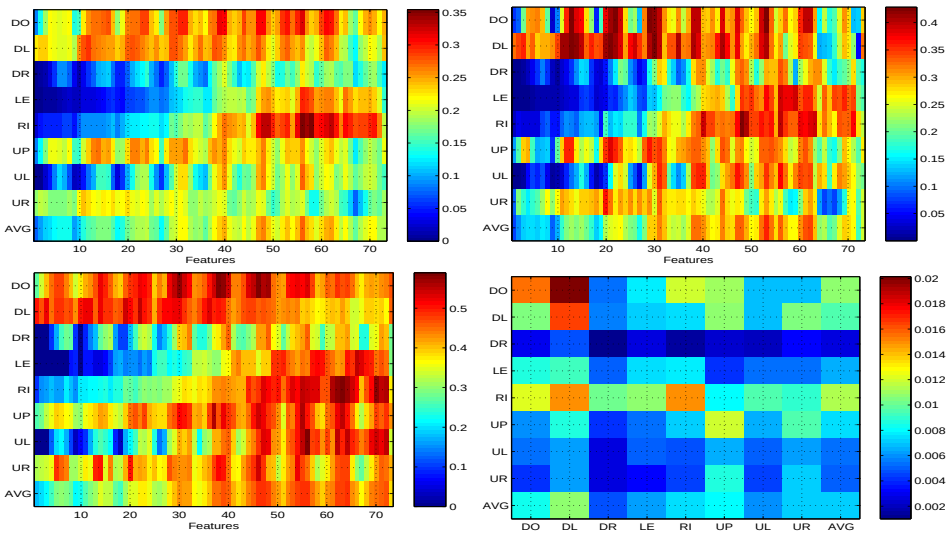


Figure 6.21: Entropy Analysis of GestPan Gestures: Channel (top-left), Frame (top-right), Class (bottom-Left) and Cross Mutual Entropy (bottom-right)

mutual entropy analysis points out low mutual information between class models and samples. Since class models of *DownRight* gesture have the lowest entropy, consequently they also have less mutual information with their and other classes, samples in the dataset. Table 6.3 summarises these observations: The dataset has a very high noise signal ratio (NSR), low mutual information and cross mutual information precision.

	$\widetilde{hC}$	$\widetilde{hHX}$	$\widetilde{hVX}$	$I(hHX, hC)$	$\widetilde{MIP}$	$\widetilde{NSR}$
Mean	0.38	0.19	0.24	0.01	0.14	0.95
Std	0.10	0.06	0.10	0.00	0.18	0.02
Min	0.00	0.00	0.00	0	0.00	0.92
Max	0.59	0.35	0.43	0.08	0.46	0.99

Table 6.3: Summary Entropy Table for the Gesture Panel dataset. The dataset has high NSR (Noise Signal Ratoi) and low MIP (Mutual Information Precision).

The statistical parameter fitting of the Gesture Panel dataset by Skewness, Kurtosis and Chi-Squared Test is shown in figure 6.22. Note that since some gestures are performed only on a small part of the grid (images), other cells on the remainder part of the grid do not hold the assumption that each cell or feature's underlying distribution is Gaussian. For example, as pointed out above in the channel and class entropy analysis, the *Left*, *DownLeft* and *Right* gestures have low complexity, in other words less usage of cells, at the beginning the features (cells). Therefore, in these cells, the underlying statistical distribution is not Gaussian, which is demonstrated by the parameter fitting analysis in figure 6.22. In fact, this situation is correct for most of

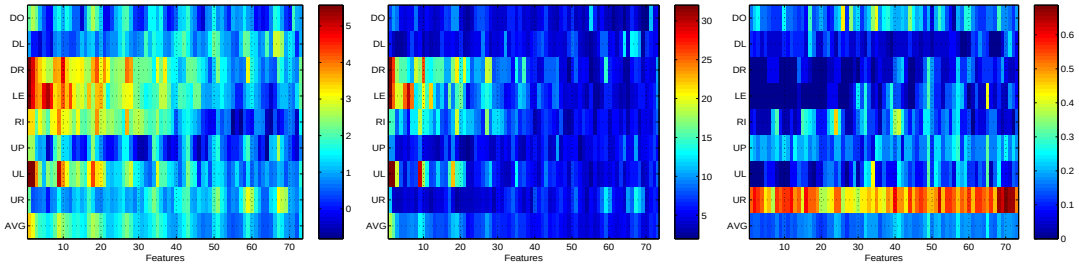


Figure 6.22: Statistical distribution parameter fitting analysis with Skewness (left) and Kurtosis (right) and Chi-Squared Test (right) for the GestPan dataset. Since cells (pixels) of binary images are used as features, the dataset is not Gaussian, in most cases.

the gestures apart from the *UpRight* gesture. This dataset generally does not behave in the Gaussian distribution way.

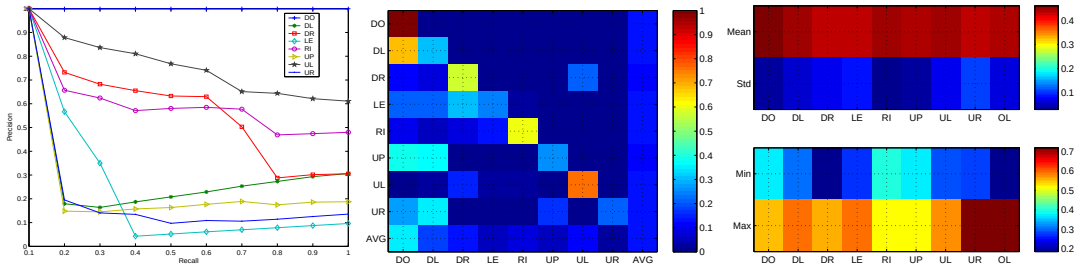


Figure 6.23: Inter (left) and average cross (middle) recall/precision of the Gesture Panel dataset using the EROS analysis. The Gesture Panel dataset does not have an orthogonal feature set. Consequently, EROS does not provide the triangle inequality for this dataset. Therefore, the EROS analysis on the Gesture Panel dataset does not obtain meaningful results and so its results are ignored. The right figure shows the periodic variance percentage (PVP) for the Gesture Panel dataset.

Figure 6.23-left and middle illustrate the EROS results. The EROS analysis reveals that there is remarkable confusion among the gestures. For example, as the EROS cross class similarity graph (6.18- middle) shows the *DownLeft* gesture is confused by the *Down* gesture. Another interesting point is that the *Up* and *UpRight* gestures are completely mixed with either *Down* or *DownLeft* gestures. On the other hand, the *Down* gesture is not confused by any other gesture. All of these abnormal observations are an indicator of the EROS algorithm's failure on the non-orthogonal feature set. For more detailed information please refer to the article [155]. The EROS analysis does not obtain meaningful results for the Gesture Panel dataset, so its results can be ignored. Figure 6.18-right is the result of the periodic variance percentage (PVP). The dataset accompanies an average 40% periodic variance among the gestures.

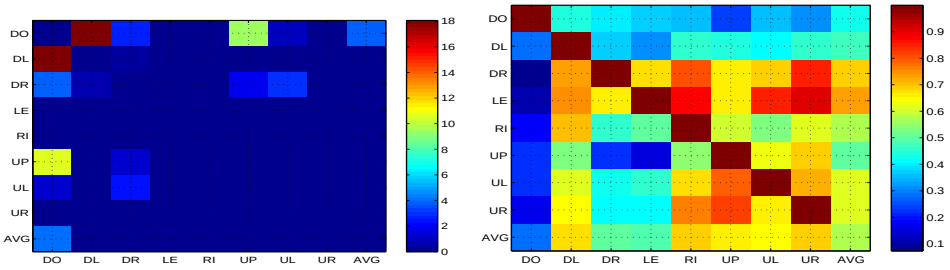


Figure 6.24: Fisher linear discriminant (left) and intersection similarity (right) analysis for the Gesture Panel dataset. Actually, the FLDA ratio in the figure is in order of $18x10^8$, and some are zero. Due to the non-orthogonal feature of the Gesture Panel, FLDA does not obtain reliable results. Therefore, its results are skipped. The intersection similarity analysis reveals a high sharing area between the *Up*, *UpRight*, *UpLeft* and *Right* gestures.

Figure 6.24 illustrates the Fisher linear discriminant and intersection similarity results. Actually, the FLDA ratio in the figure is in the order of 18×10^8, and some are zero. Both analyses indicate a high disparity of *Down* gesture. Also, the *DownLeft* and *DownRight* gestures have a higher disparity compared to other gestures. But when the analyses are looked into greater detail, the same issues faced in the EROS analysis due to the non-orthogonal feature set, once again, cause unreliable results. Therefore, the results of FLDA for the Gesture Panel dataset are ignored. The intersection analysis shows a high similarity between the *Right*, *Up*, *UpLeft* and *UpRight* gestures.

6.2.5 Flight Deck Officer

Even though, FDO gestures have been comprehensively elaborated upon and discussed in the previous chapters, a brief summary is presented here. FDO gestures are collected in two different ways, the first of which is tracker-based (FDO_PT) and the second one is computer vision-based (FDO_CV). Both datasets consist of 18 gestures of over all 94 FDO and Pilot gestures (Please refer to the related appendix for a full list of the FDO gestures). These gestures accommodate all the challenges which one would come cross during FDO's gesture recognition. The datasets consist of four static gestures (Affirmative, Clean, Hold On, Negative), six dynamic gestures (Ahead, Back, Wave Off, Down ...) and eight hybrid gestures (Left, Right, Fire ...).

6.2.6 Flight Deck Officer - Tracker-Based (FDO_PT)

FDO_PT is collected via a tracker-based device (Polhemus FasTrak) of which two sensors acquire the position of hands in a three dimensional coordinate system (x, y, z).

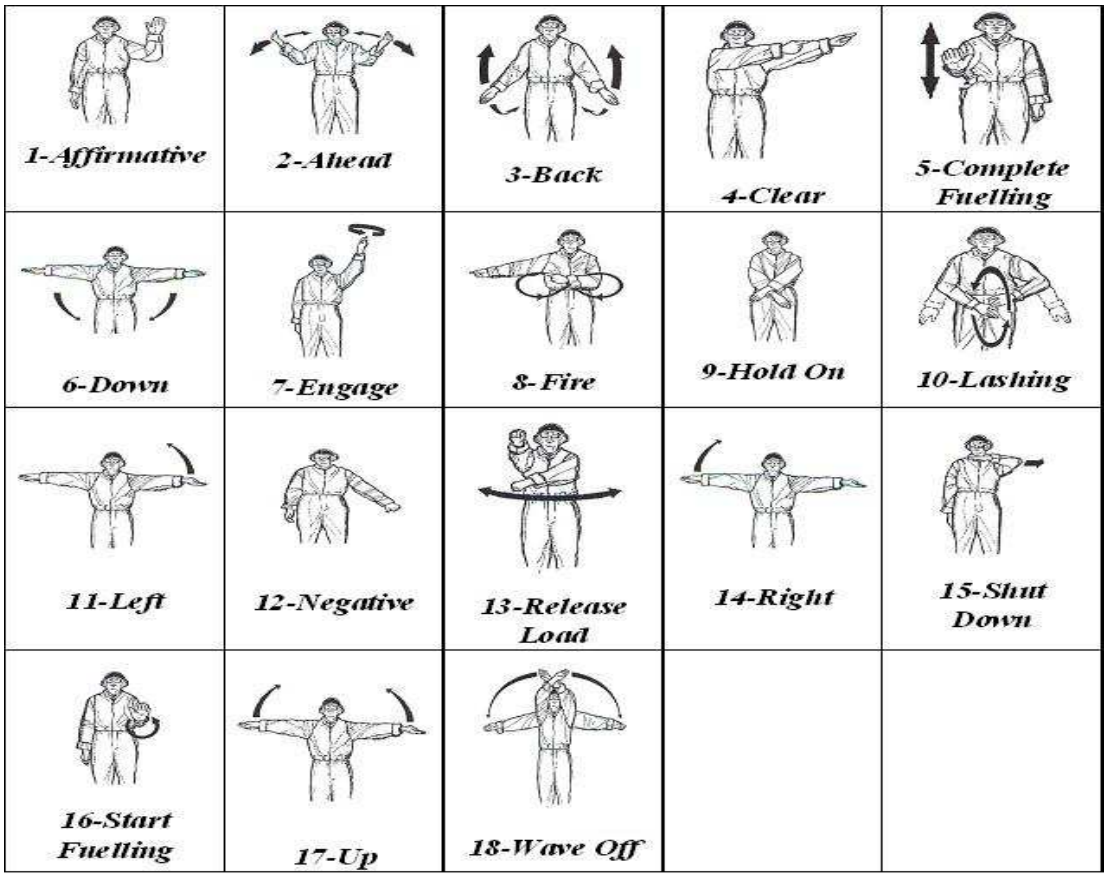


Figure 6.25: 18 FDO gestures. The subset FDO gestures of interest consists of four static (*Affirmative*, *Clear*, *Hold on* and *Negative*), six dynamic (*Ahead*, *Back*, *Down*,*Lashing*, *Up* and *Wave off*) and eight hybrid (while one hand static, the other hand is dynamic, *Complete Fueling*, *Engage*, *Fire*, *Left*, *Release Load*, *Right*, *Shut Down*, *Start Fueling*). Please refer to the related appendix for a full list of the FDO gestures.

FDO_PT consists of about 150 samples for each class and these samples are collected from only a single person in various sessions. Each raw gesture is represented by a stream of six coordinate data (x, y, z) for both hands. But these raw data are transformed to grid-based spatial features. Grid-based spatial features are raw data that are normalized according to the physical limitation of users. For normalization, it is assumed that the user is fitted into a fixed $15 \times 15 \times 15$ unit 3D grid cube such that, when the user stretches his arm horizontally or vertically upwards at the shoulder level, the user's hands touch the boundary of the grid cube on the top, left and right edge of the cube. And the feature vector of FDO_PT is :

$$F_{FDO\_PT} = F_{Grid} = [R_x, R_y, R_z, L_x, L_y, L_z, R'_x, R'_y, R'_z, L'_x, L'_y, L'_z]$$

where $[R_x, R_y, R_z, L_x, L_y, L_z]$ corresponds to the spatial grid features for the Right and Left hand and $[R'_x, R'_y, R'_z, L'_x, L'_y, L'_z]$ has the fuzzy gradient temporal feature of the grid feature.

A detailed dataset complexity and similarity analysis of FDO_PT revealed the following in previous chapter: The dataset consists of high temporal variance (PVP) and dynamic gestures that mostly encapsulates the rest of the static and hybrid gestures. EROS obtains low performance, due to a high similarity in the covariance matrix of static gestures. On the other hand, the Fisher linear discriminant analysis based algorithm, which relies on the distance between and within class distances, achieves a better disparity degree on static gestures than on the dynamic and hybrid gestures. While the spatial channel of dynamic gestures behaves as normal statistical distribution, the temporal channels do not show complete normal statistical distribution because of the way the temporal channels are constructed. And the entropy analysis shows that the dataset has high frame entropy (large bandwidth), as was expected, low channel and class entropy for static gestures, low mutual information precision (MIP) and high noise signal ratio (NSR). High frame entropy increases intersection similarity, low class entropy, MIP, and a high NSR causes wrong index predictions in the proposed recognition algorithm.

6.2.7 Flight Deck Officer - Vision-Based (FDO_CV)

The FDO_CV dataset is similar to the FDO_PT dataset apart from three points: the data collection part, the size of the dataset, and the number of users performing the gestures. Computer vision-based FDO gestures are collected via an average quality desktop web cam. Collected videos are pre-processed in order to extract the position of hands (x, y). Three different users perform the gestures. The dataset consists of over 70 samples of each gesture. Similar to FDO_CV, each raw gesture is represented by a stream of four coordinate data (x, y) for each hand. The smoothed coordinate data x, y of both hands and their gradients are used as feature vector.

$$F_{FDO\_CV} = F_{Grid} = [R_x, R_y, L_x, L_y, R'_x, R'_y, L'_x, L'_y]$$

where $[R_x, R_y, L_x, L_y,]$ corresponds to the spatial grid features for the Right and Left hands and $[R'_x, R'_y, L'_x, L'_y]$ are the fuzzy gradient temporal feature of the grid features.

The dataset analysis of the FDO_CV dataset obtains similar results to the FDO_PT dataset. The important difference between the FDO_PT and FDO_CV datasets is that the FDO_CV has more inter and intra class similarity and variance due to multiple deployed users and reduced data dimensionality (x, y).

6.2.8 Summary of Datasets

Having described and analysed each dataset individually, in this section, a comparative summary of these dataset is laid out.

Table 6.4 summarises the entropy analysis of the datasets in the form of mean∓standard deviation($\mu \mp \sigma$). The table indicates that Gesture Panel accommodates highest noise signal ratio (NSR) and lowest mutual information precision (MIP). This is mostly due to the structure of the feature set chosen. Another important observation is that the datasets deployed more than one user, accommodate a higher entropy. For example, the FDO_CV and Perrotta datasets have higher NSR and lower MIP compared to the FDO_PT and the Yang datasets, which deployed only one user for data collection.

Table 6.5 summarises the results of other complexity and similarity analyses for all the datasets. For some dataset, the SEVP (Sub-Event Variance Percentage), EROS, FLDA and parameter fitting analysis are omitted due to the either dataset being artificial, or the way these datasets are modelled which can not prevent the analyses being carried out. For example, for the Gesture Panel dataset, the SEVP, EROS and FLDA analyses are ignored. The table shows that the artificial datasets (W_Test1 and W_Test2) have more complexity compared to real world datasets. Similarly, the entropy analysis in multiuser datasets (Perrotta, FDO_CV) has more complexity than in single user datasets. the FDO_PT dataset has the lowest complexity and similarity among all dataset, because one single user is deployed for data collection and a more optimal feature set is chosen.

	$h\bar{C}$	$h\bar{H}X$	$h\bar{V}X$	$I(h\bar{H}X, h\bar{C})$	$\bar{MIP}$	$\bar{NSR}$
W_Test1	0.25∓0.31	0.78∓0.16	0.52∓0.18	0.12∓0.01	0.34∓0.25	0.84∓0.01
W_Test2	0.24∓0.24	0.70∓0.07	0.49∓0.25	0.10∓0.00	0.34∓0.36	0.86∓0.00
Yang	0.52 ∓ 0.24	0.55 ∓0.19	0.58 ∓0.12	0.25 ∓ 0.02	0.14 ∓0.20	0.55 ∓0.06
Perrotta	0.70 ∓0.09	0.64 ∓0.15	0.58 ∓0.15	0.23 ∓ 0.04	0.18 ∓ 0.10	0.64 ∓0.08
GestPan	0.38 ∓ 0.10	0.19 ∓0.06	0.24 ∓0.10	0.01 ∓0.00	0.14 ∓0.18	0.95 ∓0.02
FDO_PT	0.31 ∓ 0.20	0.33 ∓ 0.19	0.36 ∓0.17	0.19 ∓0.09	0.42∓ 0.36	0.55 ∓ 0.24
FDO_CV	0.35 ∓ 0.23	0.41 ∓ 0.18	0.47 ∓ 0.09	0.15 ∓0.07	0.39 ∓ 0.34	0.70 ∓0.19

Table 6.4: Summary of Entropy Analysis for all the datasets ($h\bar{C}$:Normalized Class Entropy;$h\bar{H}X$:Normalized Channel Entropy; $h\bar{V}X$:Normalized Frame Entropy; $I(h\bar{H}X, h\bar{C})$):Cross Mutual Information; $\bar{MIP}$:Mutual Information Precision; $\bar{NSR}$:Noise Signal Ratio. These results are the average of each dataset.)

	PVP	SEVP	Skewness	Kurtosis	Chi^2	EROS	FLDA	Intersection
W_Test1	0.2	0.1	-	-	-	78.70∓17.60	0.00∓0.00	0.91∓0.07
W_Test2	0.2	0.1	-	-	-	77.45∓18.17	0.00∓0.00	0.98∓0.02
Yang	0.23∓0.19	0.1∓0.06	0.26∓1.10	6.89∓4.67	0.22∓0.23	77.32∓21.32	0.03∓0.02	0.87∓0.11
Perrotta	0.42∓0.23	-	-0.10∓0.88	3.56∓2.43	0.86∓0.14	75.38∓17.42	0.70∓0.53	0.88∓0.15
GestPan	0.44∓0.07	-	1.42∓1.11	7.08∓4.44	0.17∓0.16	-	-	0.58∓0.26
FDO_PT	0.16∓0.17	-	-0.00∓0.45	6.20∓9.29	0.48∓0.37	78.98∓24.61	2.89∓3.29	0.17∓0.29
FDO_CV	0.14∓0.12	-	-0.09∓0.50	4.98∓3.70	0.50∓0.36	71.75∓28.65	1.76∓3.17	0.40∓0.34

Table 6.5: Summary Of Dataset Complexity and Similarity Analysis for all the datasets.(PVP:Period Variance Percentage;SEVP:Sub-event Variance Percentage; Skewness and Kurtosis: Measures of departure from normal distribution;Chi^2:Chi Squared Test; EROS:PCA Based Extended Frobenious Analysis;FLDA:Fisher Linear Discriminant Analysis;Intersection: Shared Hyper Volume Analysis. These results are the average of each dataset.)

Figure 6.26 illustrates the average EROS results of all datasets apart from Gesture Panel. The figure shows the FDO_PT dataset has the highest precision/recall rate.

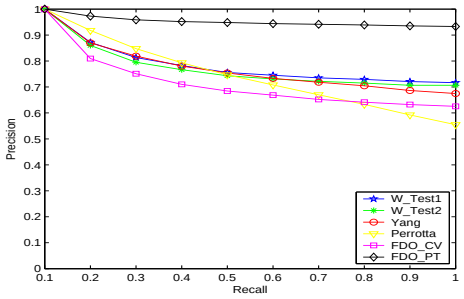


Figure 6.26: Average Recall/Precision of all datasets (apart from Gesture Panel) using EROS ($k = 1, 2, 3, \cdots 10$)).

6.3 Isolated Recognition Results & Discussion

This section discusses the isolated recognition results of the datasets analysed above. In isolated gesture recognition, the start/end points of gestures are known in advance. At the end of each gesture, the recognition algorithms are reset to expect a new gesture. In order to conduct our experiments on these datasets, as explained in the methodology section, Dynamic Time Warping (DTW), Hidden Markov Model (HMM), Elman Recurrent Neural Network (ERNN) and Recognition Machine (RM) are used as recognition algorithms. For accuracy of the recognition algorithm, as it was discussed in the Problem Definition chapter, the word recognition rate is used. The word recognition rate is actually the ratio of the correctly classified gestures to total test gestures in dataset. While for the W_Test1, W_Test2, Yang, FDO_PT and FDO_CV dataset, the 10-fold cross validation scheme is used for training and testing, for the Perrotta dataset, the basic cross validation scheme is used. The Gesture Panel dataset uses all the samples in the dataset for training and testing. Not that these validation schemes for the Perrotta and Gesture Panel datasets are deployed in order to compare our results with the existing studies on these datasets in literature.

Table 6.6 compares the recognition error (%) of the proposed algorithm (RM) with other recognition algorithms. The best results of HMM experiments from table 6.9 are shown in table 6.6 for HMM experiments. The proposed recognition algorithm (RM) achieves remarkable results compared to DTW and obtains a slightly lower (or better in some datasets) performance compared to HMM. In the case of the W_Test2 dataset, it is observed that the performance of RM is degraded with a high noise ($g = 0.2$), although HMM retains a lower recognition error. This phenomena is due to the strong

	RM (%)	HMM (%)	DTW (%)
W_TTest1	0.93∓0.43	0∓0	4.73∓4.05
W_TTest2	7.83∓2.32	2.9∓2.47	14.33∓7.46
Yang	0.86∓1.00	0∓0	27.08∓27.07
Perrotta	17.24	23.5	18.35
GestPan	2.4	0.8	21.28
FDO_CV	1.06∓0.72	0.3∓1.8	5.63∓14.71
FDO_PT	0.09∓0.14	0∓0	0.03∓0.01

Table 6.6: Recognition Error Results in Percentage (%) for on-line RM and HMM and off-line DTW ($\mu \mp \sigma$ in percentage (%) format is used for cross validations where applicable). For HMM, the best results of topologies shown in table 6.9 are selected. Although, HMM outperforms RM sligtly in some datset, detailed analysis shows that HMM makes an overestimation in the case of even unreliable and missing data, whereas RM rejects any recognition and declares a NoN_{Ges} recognition.

assumption of HMM which, actually, in most cases, leads to the misrecognition of unreliable data.

The important outcome of this study is that HMMs make strong assumptions during the recognition decision. HMMs declare recognitions even in cases where recognitions are impossible or unreliable due to high noise and missing data (table 6.6). These situations are observed in most of the datasets.

For example, in the W_Test2 dataset, in the case of high noise (g=0.2), the distinctive sub events at β channel of class A and B class are deformed and diminished as was explained in the W_Test dataset description (Figure 6.8). These situations occur approximately 6-8% of the W_Test2 dataset, in which it is impossible to segregate classes A and B. While HMM overestimates on these case, RM rejects these recognitions.

Similarly, in the Yang dataset, because of high noise and missing data, some gestures barely can be classified by even a human. Even though, as the result tables indicate HMMs (0∓0%, for the Yang dataset) make overestimations and assign them to a class without considering the quality of the signal.

This over estimation of HMM in the case of the Gesture Panel dataset is more noticeable. For example, in Gesture Panel dataset, especially two samples of the *UpRight* gesture have uncompleted, missing information. In these two samples, the only available data are the two occlusion pixels on the first image. The other pixels in the first and other images do not contain any information. They are completely blank. But HMM fails to detect this uncompleted, missing data, as the confusion table 6.8 of the Gesture Panel dataset using HMM shows.

In these circumstances, unlike HMMs, RM rejects declaring a defined class recognition and declares a non defined class recognition ($R_Ges = NoN_{Ges}$) instead. For example, in the case of the Gesture Panel dataset, table 6.7 illustrates that two samples of the *UpRight* gesture are recognized as NoN_{Ges} This advantage of RM is achieved by some heuristics along with maximum likelihood criteria employed in the *path assessor* components.

	Down	Down Left	Down Right	Left	Right	Up	Up Left	Up Right	NoN_{Ges}	Total Correct	Total Wrong
Down	***24***	1	0	0	0	0	0	0	0	24	1
Down Left	0	***33***	0	0	0	0	0	0	0	33	0
Down Right	0	0	***37***	0	1	0	0	0	0	37	1
Left	0	0	0	***36***	0	0	0	0	0	36	0
Right	0	0	0	0	***32***	0	0	0	0	32	0
Up	0	0	0	0	0	***25***	0	0	0	25	0
Up Left	0	0	0	0	0	0	***34***	0	0	34	0
Up Right	0	0	0	0	0	1	0	***24***	***2***	24	4

Table 6.7: Confusion matrix for Gesture Panel dataset using RM, which unlike HMM (table 6.8), is able to detect unreliable and missing data (For example, two samples of the *Up Right* gesture).

	Down	Down Left	Down Right	Left	Right	Up	Up Left	Up Right	Total Correct	Total Wrong
Down	***25***	1	0	0	0	0	0	0	25	0
Down Left	0	***33***	0	0	0	0	0	0	33	0
Down Right	0	0	***37***	0	0	0	0	0	38	0
Left	0	0	0	***36***	0	0	0	0	36	0
Right	0	0	1	0	***32***	0	0	0	31	1
Up	0	0	0	0	0	***25***	0	0	25	0
Up Left	0	0	0	0	0	0	***34***	0	34	0
Up Right	0	0	1	0	0	1	0	***24***	24	1

Table 6.8: Confusion matrix for Gesture Panel dataset using HMM reported in [146]. HMM makes huge assumptions during recognition. For example, in two cases, HMM misrecognizes gestures which have limited, premature information, while RM rejects these recognitions.

Table 6.9 shows HMM recognition error (%) for all the datasets. HMM experiments indicate that they perform better when the state number is smaller (3,5, 10) and topology is left to right. But, note that, these results are in contrast to the study [102], which reports that increasing the number of states in a HMM leads to a better recognition rate. In addition, it is also observed that, left to right topology is

more appropriate for gesture recognition tasks. Similarly, although RM is an ergodic topology, its frame prediction component is biased into making predictions from left to right direction. Long jumps or transitions from one frame (index/state) to others are controlled by heuristics. Even though, HMM obtains results that are comparable with RM, during the decoding of state sequences in HMM, a small number of states do not provide meaningful feedback, which is critical for the training purposes. Therefore, in the proposed recognition machine (RM), the number of states for modelling the gesture is kept to the close period of the gestures.

	W_TTest1 (%)	W_TTest2 (%)	Yang (%)	Perrotta (%)	FDO_CV (%)	FDO_PT (%)
3lr	0.6∓2.2	4.3∓3.7	0∓0	46.09∓29.73	1.2∓4.4	0.1∓0.3
5lr	0.1∓0.5	3∓2.6	0∓0	35.05∓25.20	0.4∓2.0	0.1∓0.3
10lr	0∓0	8.3∓7.6	0∓0	35.72∓30.07	0.4∓2.1	0∓0
20lr	1.1∓3.6	14.1∓12.7	0∓0	38.34∓19.94	0.4∓2.1	0.2∓0.4
3lrs1	1.4∓1.8	4.4∓3.7	0∓0	29.78∓28.34	0.3∓1.8	0∓0
5lrs1	0.1∓0.5	2.9∓2.4	0.3∓0.6	31.23∓34.23	1.2∓4.8	0.1∓0.4
10lrs1	2.3∓5.4	18.9∓25.2	0.3∓0.6	23.50∓18.37	0.8∓3.8	0.2∓0.5
20lrs1	6.2∓11.9	19.8∓23.2	0.2∓0.5	35.23∓18.23	0.4∓2.4	0.2∓0.4
3er	18.2∓20.7	23∓23.9	1.7∓1.4	46.38∓20.93	0.9∓4.1	0∓0
5er	16.83∓25.7	24.8∓27	1.3∓1.6	46.31∓25.30	1.6∓6.1	0∓0
10er	17.1∓27	20.7∓19	1.0∓1.6	29.86∓27.35	1.9∓2.7	0.1∓0.3
20er	22.2∓38.5	16.3∓2.6	1.6∓1.7	43.34∓23.23	2.5∓4.2	0.2∓0.5

Table 6.9: Recognition results in the format of $\mu \mp \sigma$ percentage (%) for W_TTest1 ($g = 0.1$) W_TTest2 ($g = 0.2$), Yang, Perrotta and FDO_PT, FDO_CV datasets with different states (3,5,10, 20) and topologies, left to right (lr), left to right 1 skip (lrs1) and ergodic (er) in the HMM experiments. 10-K fold cross validation scheme is applied.

The best results from the W_Test1 and W_Test2 obtained using HMM, are 0∓0% and 2.9∓2.47%, respectively, without considering the quality of the signal. The recognition machine obtains 0.93∓0.43% and 7.83∓2.32% recognition rates for W_Test1 and W_Test2. Besides W_Test1 and W_Test2, several other experiments are conducted on the W_Test using various combinations of noise parameters $0.025 \leq g \leq 0.2$, while other parameters are kept constant (d=h=0.2, c=0.1 and *irrel on*). The results of these experiments with RM are shown in figure 6.27. In this experiment it is shown that RM obtains a consistent recognition rate below some noise level ($g \leq 0.125$). In literature, Kodous implemented an off-line decision tree based recognition algorithm to recognize W_Test1 and W_Test2 artificial classes [57]. In the case of raw data with its derivative, the author obtained 1.0∓0.3% recognition error on W_Test1 using a

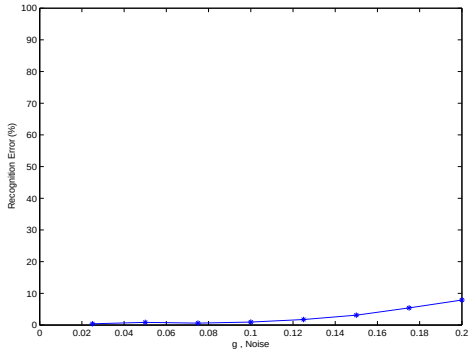


Figure 6.27: Average Recognition error for dataset W_Test with various noise levels (g) when c=0.1, h=d=0.2 and *irrel on*.

decision tree. The author reports an 9.9∓1.1% error rate as the best performance for W_Test2 using HMM. In addition, the author reports 0.9∓0.3% results in case of noise free setting (g=0) with an off-line decision tree recognition algorithm.

RM achieves an 0.86∓1.00% error rate on the Yang dataset, whereas HMM with left to right topologies and a small dataset obtains full recognition. As explained in the Yang dataset, some samples of Laser gestures are not representative. While, HMM fails to address this missing data, RM detects it. On the other hand, the author of the Yang dataset reports about 2.1% recognition errors [64] with an off-line PCA (principal component analysis) based recognition algorithm.

Although the complexity and similarity analysis on the Perrotta dataset does not point to a very high complexity among other dataset, it has been observed that the Perrotta dataset obtains a higher recognition error using all the deployed recognition algorithms. The interesting point here is that the DTW algorithm performs around 5% better than HMM. Low performance is attributed to a very large periodical variance (0.42∓0.23%) in the dataset.

The author of the Perrotta dataset implements an Elman Neural Network as the recognition algorithm [89] and achieves about 25.0% recognition errors. For a couple of reasons, ENN is not applied to the other datasets. First, ENN does not produce a meaningful and trivial outcome. The interpretation of ENN output even itself involves another sub-recognition task. In addition, the training time of ENN is very long. For example, it takes more than 36 hours to train ENN for the Perrotta dataset. Even if, all of these issues are resolved and resources are provided, ENN still fails to obtain remarkable results, compared to RM, HMM and even DTW.

Experiments on the Gesture Panel show the power of RM over HMM and DTW with image-based gestures. Note that due to non-orthogonal features, the complexity

and similarity analyses on the Gesture Panels fail to obtain meaningful results. Moreover the features do not show the assumed Gaussian distribution. Even though, RM obtains reliable results (2.4%). The original author of Gesture Panel dataset reports 0.8 recognition error by left to right 8 state HMM [146]. But unlike HMM, RM rejects unreliable and missing gesture recognition, which is a positive point for training purposes. Tables 6.7 and 6.8 illustrate the confusion matrix of the Gesture Panel dataset using RM and HMM, respectively.

RM achieves very accurate results on the FDO_PT dataset (0.09∓0.14%). Misrecognition on the FDO_PT dataset using RM generally occurs on the *Wave-Off* recognition due to the limitation of the input device (Polhemus FastTrak) as has been discussed in the data collection of the FDO_PT dataset section in the Gesture Analysis and Modelling chapter. Since the space where the *Wave-Off* is performed can exceed the Polhemus FastTrak range, the *Wave off* samples consist of a high noise.

The complexity and similarity analysis on the FDO_CV dataset points out more complexity compared to the FDO_PT dataset, due to multiple users being deployed during data collection. These complexities pave the way for a lesser performance on FDO_CV compared to the FDO_PT dataset. RM achieves 1.06∓0.72% recognition error on FDO_CV dataset, whereas HMM obtains 0.3∓1.8%. As illustrated in figures 4.9, 4.10 and 4.10 the trajectory of *Down*, *Engage*, *Left*, *Right*, *Shut Down*, *Up* and *Wave off* FDO_CV gestures contain some abnormal samples. While most of these gestures are detected by RM and rejected, HMM fails to detect and reject most of these gestures. A user-based recognition error is demonstrated in table 6.10. Recognition errors mostly occur among second and third users' samples, which can be seen in the trajectories of FDO_CV as mentioned above.

	User1	User2	User3	User4
Error (%)	0	0.56	0.49	0.06

Table 6.10: Recognition error for each user in the FDO_CV dataset.

In summary, we can make the followings observations regarding isolate recognition:

- RM achieves comparable results with HMM, even though, RM is an on-line algorithm and uses limited historical data.

- Although, HMM outperforms RM on some dataset, HMM lacks controlled recognition. HMM overestimates on unreliable and missing data, whereas RM rejects premature, missing and unreliable data recognitions.

- Unlike RM, HMMs with a small number of states perform better, although they produce less meaningful feedback for training in case of error. RM utilises all frames (states) in the trajectory for class modelling. Therefore, RM accumulates better feedback information.

- Left to Right HMM topologies obtain better results compared to ergodic topologies. Similarly, even tough RM has an ergodic structure, its frame predictor component is biased into making the frame prediction operation from left to right.

- Elman Neural Network has the following drawbacks: It does not produce meaningful outcomes, its performance is low compared to other considered recognition algorithms and it takes a long time to train the network.

- In addition to being off-line, DTW has the lowest performance amongst the other recognition algorithms.

- For continuous gesture recognition, only RM and HMM can be considered as recognition algorithms, due to their achievement in isolated gesture recognition.

6.4 Hybrid MLPNN/RM for Isolated Gesture Recognition

When RM was elaborated in the previous Recognition Algorithm chapter, it was pointed out that RM is a component-based algorithm. In section 5.3.1, a multilayer perceptron neural network (MLPNN) was proposed to substitute the *Frame Predictor* in order to validate the modularity of RM. Note that the *Frame Predictor* is a kind of function approximation which predicts the next index (N) given the degree of membership curves (M) and the most recently predicted frame index (V) ($N = Frame\_Predictor(M, V)$). But unlike the *Frame Predictor*, the introduced MLPNN approximates N_i directly from the pre-processed input frame (feature vector,$F_{i,t}$) and $V_{i,t}$ for the class C_i at time t.

For isolated gesture recognition the FDO_PT dataset is used to validate the MLPNN component as the frame prediction component. For each class, a MLPNN is employed. The current index input (V_i) and output (N_i) are represented as binary. $(\vartheta + L) \times 90 \times 90 \times L_i$, two hidden layer feed forward MLPNN architecture is deployed for the class C_i. Readers are referred to *Multi Layer Perceptron Neural Network as*

Frame Predictor section 5.3.1 in the previous chapter for a detailed discussion about the proposed MLPNN.

The experimental framework applied to the FDO_PT using RM is also deployed for isolated RM/MLPNN experiments. The 10 K-fold cross validation technique is used for training and testing. RM/MLPNN achieves $6.31 \mp 1.31\%$ recognition error rate. It has been observed that RM/MLPNN misclassifies the *Affirmative* and *Engage* gesture, because of the high inter temporal and spatial similarity between these two gestures. In addition, it fails to recognize some samples of *Clear* and *Complete Fuelling* gestures. It has also been observed that the approximation power of RM/MLP for the next index (N_i) in case of static gestures is limited compared to dynamic gestures, because the spatial and temporal feature vector in case of static gesture does not contain sufficent discriminancy between frames.

6.5 Continuous FDO Experiments

In addition to isolated gesture recognition, continuous gesture recognition experiments have been conducted on the FDO_PT and FDO_CV datasets. In the continuous case, a sample (sentence) contains sequential gesture and the start/end of gestures in sentences are not known in advance. Furthermore, sentences include transition data from one gesture to another. These situations, namely, unknown start/end points, transition data and multiple gestures in sentences, pave the way to more challenging recognition tasks.

Continuous gesture recognition experiments are conducted on FDO_PT and FDO_CV sentences with various lengths such as 5, 10 and 20. In the following experiments, the data for continuous gestures are synthesized. The synthesized data is referred to as a sentence. Each sentence is constructed out of isolated samples from the datasets. In a sentence, from one gesture to next one is implemented by a transition function if two consecutive gestures are different. The transition function is a linear function which produces data from the end of one gesture to the starting point of the next gesture by interpolating using a fixed sample rate for the dataset. The fixed sample rate is approximately the same for the dataset and it is the average of the first absolute gradient between consecutive points in the case of all dynamic gestures in the dataset. In addition, in order to make the transitions more realistic, transition function applies normal noise to transition data too. In the case of the FDO_CV dataset, since multiple users are employed during dataset collection, transition from one gesture to another one is only allowed among the same user's samples. In other words, a sentence does

not contain gestures from two different users in it in the case of the FDO_CV dataset.

The construction of sentences in this way paves the ways for having control parameters (similar to W_Test dataset) in sentences in order to make more detailed and controlled analysis for the behaviour of recognition algorithms. For example, this parametric scheme enables us to analyse all possible transitions from one gesture to another in the dataset. Construction of this type of real continuous dataset, which accommodates all reasonable transitions with some degree of variance, is extremely costly. In addition, a periodic control parameter (d) is employed to analyse temporal behaviour in the case of continuous recognition.

In isolated gesture recognition experiments, it is concluded that HMM and RM are the two recognition algorithm which obtain the most successful results. Therefore, in continuous recognition experiments, only these two algorithms are considered as the recognition algorithm. As HMM topology, 3 state, left to right 1 skip (3lrs1) HMM is employed, as table 6.9 illustrates. This topology obtains the best isolated recognition rates for the FDO_PT and FDO_CV dataset.

For continuous recognition, RM deploys the class models obtained from isolated gesture recognition without any modification. But on the other hand, HMMs are trained using two different approaches. First, similar to RM, class models obtained from isolated recognition are used without any modification with an additional class, NoN_{Ges}. Class model NoN_{Ges} represents all transitions data and is constructed out of all transitions data from sentences [148, 73]. In the second approach, some folds of sentences (10-fold cross validation) are used for training. In other words, HMMs are trained using sentences in which the start/end of gestures are known using HMM in order to provide robust segmentation. In addition, for these two approaches, HMMs are trained with and without grammar.

In order to assess the recognition results, two merits are considered: Sentence and Word (Gesture) rate. In order to be classified as correct sentence recognition, all gestures in the sentence have to be recognized in the same order. This means, during the recognition of a sentence, substitution, insertion and deletion errors should not occur [146]:

- S= Substitution error when the system incorrectly classifies a gesture.
- D=Deletion errors arise when the system fails to recognize a gesture
- I=Insertion error occurs when the system 'hallucinates' a gesture.

The second merit, word (gesture) recognition rate represents the ratio of correctly recognized gestures in sentences. For example, if a gesture is misclassified in a sentence,

	5		10		20	
	Sentence (%)	Word (%)	Sentence (%)	Word (%)	Sentence (%)	Word (%)
HMM_{FDO_PT}	40.23	93.99	19.0	94.71	10.94	91.74
RM_{FDO_PT}	89.60	98.73	74.00	98.31	63.87	98.69
HMM_{FDO_CV}	38.00	89.75	5.32	90.05	0	84.98
RM_{FDO_CV}	51.46	91.66	16.00	91.08	6.66	91.12

Table 6.11: Continuous recognition sentence and gesture recognition results (%) of the FDO_PT and FDO_CV datasets using HMM and RM over various sentence lengths (5,10,20).

while this sentence is automatically classified as wrong, the recognition of other gestures in the sentence are not affected. Therefore, word recognition tends to be greater than the sentence rate.

Several continuous recognition experiments are conducted with a range of sentence lengths (5, 10 and 20) on the FDO_PT and FDO_CV. Each experiment consists only of the same length of sentences. The number of sentences in the experiments is 250, 250 and 150 for 5, 10, 20 length-sentence in FDO_PT, respectively. In case of FDO_CV, 100 sentences are deployed for each length (experiments).

Table 6.11 illustrates the experiments results (sentence and word recognition rate) using HMM and RM. The table shows RM outperforms HMM in terms of sentence recognition. RM, in particular, obtains remarkable results on FDO_PT. Reduced feature set (x and y) and multiple users deployed for the data collection in the FDO_CV dataset lead to a relatively lower continuous recognition performance rate using RM and HMM.

Table 6.11 points out an interesting result regarding word and sentence recognition rates. It is observed that the sentence recognition rate decreases as the sentence length increases. Whereas, word (gesture recognition) rate is approximately constant, especially in the case of RM, for all sentence lengths. This may be interpreted as follows: the increased error rate in a sentence as the sentence length increases, is not related to the length of the sentence, but to the wrong recognition of some certain gestures. Transition confusion analysis reveals this phenomenon more clearly. Figure 6.28 shows the confusion transition matrix (transition from raw gesture to column gesture) for length 3, 5, 10 and 20 for FDO_PT sentences. Note that in the case of length 3 confusion matrix, all transitional combinations in the FDO_PT dataset are tested (total sentence number is $18^3 = 5832$). In this experiment, 94.00 and 98.70% sentence and word recognition rates are obtained, respectively.

As these confusion matrices on FDO_PT sentences show, some gestures, especially *Wave off* and *Up*, cause most of the misrecognition in sentence. This can be attributed

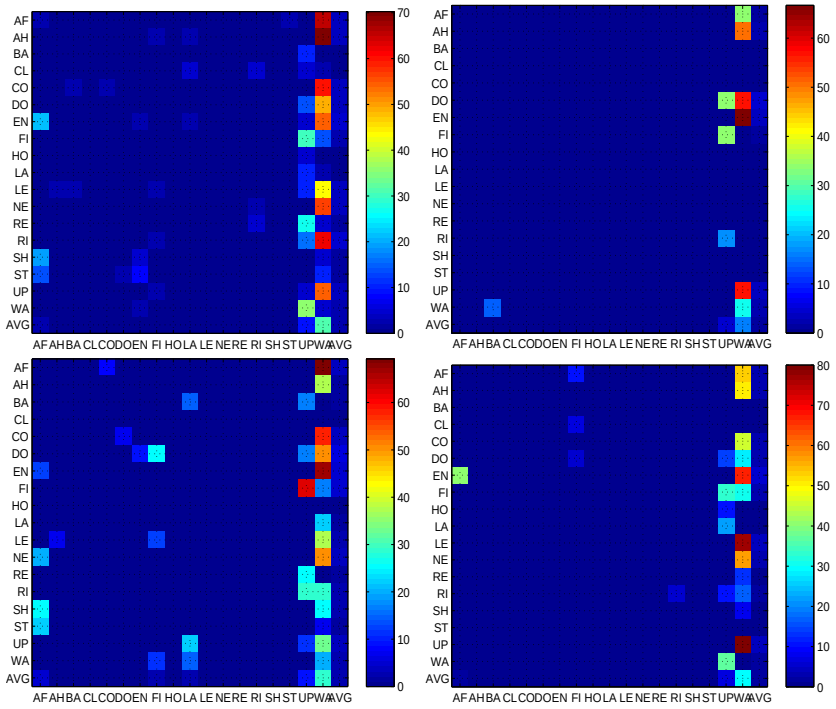


Figure 6.28: Transition confusion matrix rate (%) in case of RM for sentence lengths 3, 5 10 and 20 (from top left to right bottom) for FDO_PT sentences respectively. For length 3, all combinations of transitions are considered ($18^3 = 5832$ sentences). Insertion error occurs when the *Wave off* and *Up* gestures are performed. Transition data from a gesture to the *Wave off* gesture results in an insertion error of *Up* gesture in many cases. Transition data between *Affirmative* and *Engage* gestures also causes misclassification.

to insertion error due to the spatial similarity between the *Wave off* and *Up* gestures. The insertion error occurs (*Up* gesture emerges) when the *Wave off* gesture is performed immediately after a gesture. There is a remarkable spatial similarity between *Wave off* and *Up*. The *Up* gesture is a kind of half period shifted of the *Wave off* gesture. In other words, the starting point of the *Wave off* gesture is at halfway of the *Up* gesture. In the case of performing the *Wave off* gesture from a previous gesture, in most cases, the transition data consists of the first half of the *Up* gestures. And, if immediately *Wave off* gesture is performed, by the half way periods of the *Wave off* gesture, an *Up* gesture emerges (insertion error). Therefore, there should be explicit marking (such as a short pause) at the beginning of the *Wave off* or *Up* gesture to avoid this insertion error. Misclassification which occurs during the *Wave off* gesture is also related to volume limitation of the input device (Polhemus FasTrak). Note that,in reality, *Wave off* and *Up* gestures are, fortunately, not performed consequently because this sequence is not meaningful. In addition, the *Affirmative* and *Engage* gestures are

mixed because the spatial feature of these two gestures are similar and transition data from a gesture to *Affirmative* degrades the only difference (temporal) between these two gestures.

RM and HMM achieve lower sentence rates on FDO_CV sentences. Reduced feature dimensionality (only x and y) does not have distinctive properties of some gestures such as *Back*, *Engage*, *Lashing* which have z depth information. Especially during the transition one gesture to another, substitution error occurs. In figure 6.29, the transition confusion matrix of these gestures in varying length sentences (3, 5, 10 and 20) for RM is illustrated. In addition, as the complexity and similarity analysis reveals the FDO_CV dataset contains a high temporal and spatial variance due to multiple users deployed for data collection.

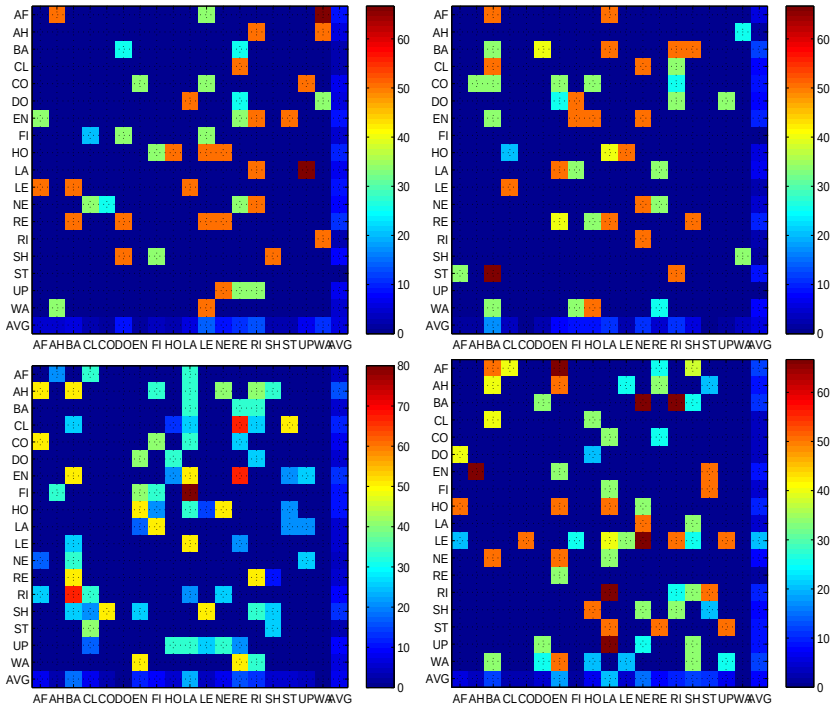


Figure 6.29: Transition confusion matrix using RM for sentence length 3, 5 10 and 20 (from top left to right buttom) for FDO_CV sentence length using RM respectively.

Although HMM obtains good results on isolated gesture recognition, it does not perform as well on continuous recognition as RM. HMM retains the approximate word recognition rate among the different sentence lengths (5,10,20).

Further analysis reveals the reason behind HMM's poor performance in the continuous case. As explained earlier, two different approaches are deployed for HMM's training (class models from isolated recognition with an additional NoN_{Ges} class and sentences themselves). Table 6.11 illustrates the results of the first approach without

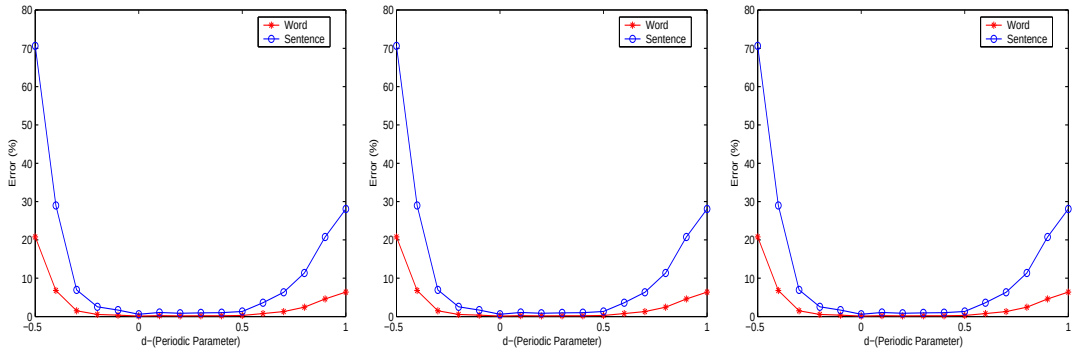


Figure 6.30: Sentence and word recognition error with various sentence length 3, 5, 10 (left, middle, right) in case of various d (periodic) control parameters ($-0.5 \leq d \leq 1$) for continuous gesture recognition

grammar. HMM obtains slightly better results with grammar (for example, 59.2% recognition rate when sentence length is 5), but since grammar is not yet embedded into RM, HMMs are trained without grammar, in order to be fair. But even when HMMs are trained with grammar using both approaches, HMM does not outperform RM. In these cases, it is observed that HMM fails to recognize static gestures and especially non-defined, transition data (NoN_{Ges}). This indicates that HMM fails to represent a wide range of transition data NoN_{Ges}.

Note that in other domains of HMM, for example, in speech recognition, the connector word (transition data from one gesture to another, e.g. *silence* in speech recognition) does not vary as much as in the gesture recognition domain [148, 73]. Therefore, HMM's success in the continuous recognition of other domains is not observed for the continuous recognition in the gesture recognition domain. Several studies have been conducted to model non-gesture, transition data [105, 153]. But, since modelling of undefined gestures contain large sets of movements, these attempts fail to deliver a reliable undefined gesture spotting. Whereas RM avoids model undefined gestures but instead implements on-line rejection heuristics. Therefore, RM is able to recognize undefined gestures or transitions NoN_{Ges} by employing embedded rejection heuristics.

In order to examine the temporal properties in case of continuous gesture recognition, a periodic control parameter (d) is proposed. Similar to the W_Test, the periodic control parameter (d) serves to change the length of sentences (stretch/compress). For example, when d=-0.2 and 0.2, all gestures in the sentences are compressed or stretched 20% of their length, respectively. The stretching and compression operations are implemented using uniform linear interpolation or averaging two consecutive points (compression), as explained in the channel construction section in chapter four.

Figure 6.30 illustrates the results of sentence and word recognition rates when $-0.5 \leq d \leq 1$ for FDO_PT sentences. The figure shows that RM obtains reliable results when d is in the range of -0.2 and 0.5. Note that in order to build confidence on the observed path, the *Path Assessor* components of RM deploys the *path age* (QA) heuristic which is required to observe 10% of period at each milestone of the path. Therefore, when $d < -0.2$, the RM cannot build enough confidence on the observed path and thus rejects recognition.

In summary, continuous experiments reveal the following observations:

- RM outperforms HMM on FDO_PT and FDO_CV sentences.
- Several experiments have been conducted with different sentence lengths [3, 5, 10 and 20]. In these experiments, while word recognition rate is approximately constant, sentence rate is degraded as the sentence length increases.
- Transition analysis shows that some gestures such as *Wave Off*, *Up*, *Affirmative* and *Engage* cause most of the misclassification, due to their spatial similarity and limitations of input devices.
- HMM fails to recognize transition data and static gestures. Even with grammar, HMM does not obtain reliable results as much as RM.
- Reduced feature set (x and y) and multiple users deployed for data collection cause the relatively lesser continuous recognition performance for FDO_CV sentences using RM and HMM.
- RM retains its sentence and word recognition rate in case of periodic temporal variance.

6.6 Summary

This chapter describes the datasets in detail and their complexity analysis, isolated and continuous gesture recognition experiments.

In order to validate the proposed recognition algorithm RM, one artificial parametric dataset (W_Test [57]) along with other three supplementary real world datasets (Gesture Panel [146], Yang [64, 63], Perrotta [89],) are investigated in addition to the FDO datasets, (FDO_PT and FDO_CV).

The artificial dataset W_Test contains several control parameters which are utilised to analyse performance of RM and other recognition algorithms with various combinations. On the other hand, the supplementary real world datasets are investigated, in

order to validate and analyse the performance of the proposed algorithm (RM) under various real world scenarios (data acquisition technique, single/multiple user).

This study carries out isolated and continuous gesture recognition on real world and artificial datasets. Continuous gesture recognition experiments are focused on the FDO_PT and FDO_CV dataset whereas isolated gesture recognition experiments are applied to all the datasets.

Performance of the proposed recognition algorithm (RM) is compared with Dynamic Time Warping (DTW), Elman Neural Networks (ENN) and the Hidden Markov Model (HMM) for isolated recognition experiments. Since the recognition results of ENN are low over a real world dataset and also taking into account its other disadvantages, such as long training time, non meaningful, human readable representation (black box) and output; off-line training, ENN is not applied to the other datasets. Several configuration of states (20,10,5,3) and topologies such as left to right (*lr*), left to right one skip(*lr1s*) and ergodic (*er*) are considered for HMM recognition. For continuous recognition experiments only HMM and RM are considered due to their better performance in the case of isolated gesture recognition.

Before proceeding to isolated and continuous gesture recognition, detailed complexity and similarity dataset analyses are performed upon each dataset. Several data analyses techniques are considered: Chi-Square, skewness and kurtosis analysis; Fisher linear discriminant analysis; principal component-based *EROS* analysis; intersection volume analysis and temporal analysis for variance in samples length and position of sub-events (SEVP).

For recognition algorithms, training and testing schemes are chosen to be in agreement with those studies reported in literature, if available on the dataset of interest. Generic and 10 K-Fold cross validation technique is considered for most of the datasets in the case of isolated recognition.

Complexity and similarity analyses reveal that the Gesture Panel dataset consists of the highest noise signal ratio (NSR) and the lowest mutual information precision (MIP). This is mostly due to chosen the nature (non-orthogonal) of the feature set for the Gesture Panel. Artificial datasets (W_Test1 and W_Test2) consist of more complexity than real world dataset. It is observed that, multiple user deployed datasets contain higher entropy. For example, the FDO_CV and Perrotta dataset contain higher NSR and lesser MIP, compared to the FDO_PT and Yang datasets, which deploys only one user. The FDO_PT dataset has the lowest complexity and similarity amongst all the datasets. The EROS analysis reveals that the FDO_PT dataset has the highest precision/recall rate.

Isolated gesture recognition experiments are carried out upon all the datasets. Since

each test data contains only one gesture, in the case of isolated gesture recognition, the issue of start/end points does not occur. Dynamic Time Warping (DTW), Hidden Markov Model (HMM), Elman Recurrent Neural Network (ERNN) and Recognition Machine (RM) are used as recognition algorithms. Isolated recognition experiments reveal the following major observations: RM and HMM outperform other considered algorithms. Both the Elman neural network and DTW (even when it is an off-line algorithm) fail to obtain accurate recognition results. Even though HMM obtains slightly better results compared to RM, further analysis reveals that HMM does superficial analysis and overestimates unreliable and missing data, whereas RM rejects any recognition in these circumstances. HMM performs generally better with a small number of states (3,5) which does not provide comprehensive feedback for training purposes if a gesture has performed wrongly. In addition, it is observed that left to right topologies in HMM obtain better results. Similarly RM is biased towards making the frame predictions from left to right.

Continuous gesture recognition experiments are conducted on FDO_PT and FDO_CV sentences with various lengths (5, 10 and 20). In the continuous case, samples (sentence) contain sequential gestures and and transition data between different gestures. The start/end of gestures in sentences are not known in advance. Sentences are constructed from isolated samples from FDO_PT and FDO_CV datasets and transition data are created by a linear interpolation function which also accommodates a degree of random Gaussian noise. Constructing sentences in this way serves to accommodate control parameters (for instance d periodic control parameter in the W_Test dataset) over sentences in order to make more detailed and controlled analysis about the behaviour of recognition algorithms.

Two of the most successful recognition algorithms, HMM (3lrs1, 3 state, left to right 1 skip) and RM in isolated gesture recognition experiments are considered for continuous recognition experiments. In order to assess the recognition results, two criteria are considered: Sentence and Word (Gesture) rate. HMM and RM deploy the class models obtained from isolated gesture recognition without any modification. In the case of HMM, the additional class model (NoN_{Ges}) representing transitions is constructed out of all transition data from sentences. HMM is also trained using continuous sentence directly. This scheme fails to obtain reliable recognition results.

Continuous gesture recognition experiments reveal that RM outperforms HMM in terms of sentence recognition. RM, especially obtains remarkable results on FDO_PT. Reduced feature set (x and y) and multiple users deployed for data collection in the FDO_CV dataset cause the relatively lesser continuous recognition performance using RM and HMM for FDO_CV sentences. It is observed that while the word recognition

rate is approximately constant, the sentence recognition rate decreases as the sentence length increases due to wrong recognition (insertion error) of certain gestures (*Wave Off*, *Up*, *Affirmative* and *Engage*) in sentences during the transitional phase. These insertion errors also occur due to the volume limitation of the input device and spatial and temporal similarity of gestures.

Although HMM obtains good results on isolated gesture recognition, it does not perform as well on continuous recognition as RM (even with grammar knowledge being provided). HMM retains an approximate word recognition rate among various sentence length (5, 10, 20). HMM fails to recognize static gestures and especially non-defined, transition data (NoN_{Ges}). This indicates that HMM fails to represent (model) on a wide range of transition data NoN_{Ges}.

In order to test the temporal properties in the case of continuous gesture recognition, a periodic control parameter ($-0.5 \leq d \leq 1$) is also introduced. The periodic control parameter (d) stretches or compresses the samples in the dataset in terms of percentages. RM achieves reliable results when d is in the range of -0.2 and 0.5. Outwith this range, RM considers missing sentence and unreliable data and rejects recognition.

In addition, a multilayer perceptron neural network (MLPNN) is introduced for the *Frame Predictor* component of RM to validate modularity of RM. A new hybrid system RM/MLPNN is tested on the FDO_PT dataset for isolated gesture recognition. RM/MLPNN achieved reasonable recognition rates for dynamic gestures but in the case of static gesture MLPNN does not lead to the same overall recognition as frames in static gestures do not have enough distinctive qualities to approximate the index.

Chapter 7

Conclusion

This chapter highlights the achieved outcomes of the study and looks at further directions. In addition, a brief summary of the thesis will be represented.

The aim of this study is to develop an on-line algorithm to recognize continuous dynamic and static gestures of FDOs without explicitly indicating the start and end of the gestures in the context of a virtual training simulator. Gestures need to be recognized in a real time manner as data becomes incrementally available. In addition, the user should be able to recieve feedback in the case of a wrong performance of any gesture. Recognition of FDO gestures should be tolerant to noise and inter/intra personal spatial and temporal variances.

In order to tackle this problem, in this thesis, the Markovian process and the dynamic programming based algorithm, Recognition Machine (RM) have been proposed. The recognition machine is based on a generic pattern recognition framework consisting of nine interactive components. RM conceptually is an on-line template matching technique. The main idea behind the recognition algorithm is to exploit the sequential consistency of the input frames according to the class models by using a dynamic programming paradigm and the Markovian process. Sequential consistency or so-called *Score* (S) addresses the similarity between the incremental input data and the class models. *Scores* employ similarity factors (Θ) for each class with an on-line sequential decision process which involves some predictions. The prediction process is a probabilistic estimation of the index of frames (N) in each class (C) which are spatially closest to the input frame (X), given the most recently predicted frame index (V).

The recognition machine is validated over several parametric artificial and real world datasets by employing various isolated and continuous gesture recognition experiments. The results of RM are also compared with other popular recognition algorithms such as HMM, RNN and DTW. The outcome of these analyses reveals that RM has

greater potential to achieve the aim of this study compared to the other investigated algorithms.

7.1 Summary of Chapters

The outcome of each chapter is summarized in this section. The first chapter introduced the origins of the problem in the context of Flight Deck Officer training. The main contributions of the study are also listed there. The second chapter represents the formal definition of the problem in the context of temporal pattern recognition. Necessary notation and terms are introduced. The problem is defined as one of on-line temporal pattern recognition and its challenges are pointed out. The similarity of the problem with other related domains (speech and handwriting) are represented in order to benefit from the experience of these communities in the following chapters. The chapter also introduced some of the complexity and similarity analysis used in the thesis.

A comprehensive literature review on gesture recognition is given in the third chapter. The proposed recognition algorithm is structured in the light of the generic pattern recognition frameworks which comprises, sensor processing, feature analysing (extraction and selection), modelling and a recognition algorithm. These components of the generic pattern recognition framework are elaborated in detailed. Advantages and disadvantages of various techniques employed for recognition algorithms, data analysis and acquisition are pointed out in a way to be utilised in the implementation of the thesis. For example, in the case of sensor processing, it is noted that computer vision-based techniques are less cumbersome compared to tracker-based input devices. But on the other hand, the reliability and accuracy of vision-based data acquisition is limited compared to tracker-based methods. These advantages and disadvantages are observed in the implementation of these sensor techniques in the study. As a recognition algorithm along side NN, DTW and other algorithms (Decision trees. hybrid approaches), it is reported in literature that HMM is one of the most promising temporal recognition algorithms. Therefore a detailed analysis of HMM is presented comparing it with the proposed recognition algorithm RM. These analyses reveal the difference between RM (for example, RM approaches data more carefully for the detection of the start/end of gestures and missing wrong gestures) and that RM shares similar basic properties (such as the Markovian process) with HMM.

In this thesis, FDO gestures are investigated using two datasets (FDO_PT and FDO_CV). They differ only in the method used for data acquisition. The FDO_PT

dataset is based on a tracker input device, whereas FDO_CV is computer vision-based. These two datasets have common properties apart from number of samples, data acquisition methods, and the number of users used to perform the gestures. While FDO_PT deployed only one user (the author itself), FDO_CT deployed four users.

Template-based representation is implemented for modelling gestures and temporal classes. Although, this representation is not as popular as feature-based due to its limitations in representing large variances in samples, in view of its advantages regarding on-line and incremental recognition, this approach was chosen. Class models are summary representations of the training cycles in the form of templates which correspond to the trajectory of classes based on features with summary, compact statistical parameters (mean μ and statistical deviation σ). In fact, class construction is based on estimating these statistical parameters which represent the best underlying statistical distribution of training data at each time point of the channels. In this study it is assumed that features are independent of each other, and training data at any time index of a channel obeys normal statistical distribution. Therefore during class model construction, each channel is constructed independently of each other. Channel construction procedure includes: sensor processing, feature analysing, period estimation for the class, stretching/compression of data and sub-event alignment and statistical estimation of model parameters.

Features represent directly or indirectly spatial or temporal properties of classes. For example, in the case of the FDO_PT and FDO_CV datasets, various spatial features sets (raw, angular and grid) are considered. Since the grid-based feature set F_{Grid} achieved a better performance over the segmented the FDO_PT dataset with the proposed recognition algorithm, grid-based feature set F_{Grid} is used as the main feature set for FDO_PT and FDO_CV datasets. For temporal features, fuzzy gradient features (x') of a spatial feature (x) are used. The fuzzy gradient features contain the direction of consecutive spatial grid features, which indicates whether it increases (1), decreases (-1) or lies between the extremes [-1,1] in spatial grid feature space. For example, the size of the feature vector is 12 in the case of FDO_PT and 8 in the case of FDO_CV datasets.

Having constructed the class models, the thesis introduced various similarity and complexity analysis techniques for temporal datasets in order to obtain certain characteristics of datasets before any recognition algorithms were deployed. The following techniques were proposed: Entropy analysis for class and channel complexity, mutual information between samples and class models and noise signal ratio; Chi-Square, skewness and kurtosis analysis for the fitness of statistical parameters of class models;

Fisher linear discriminant analysis; principal component-based *EROS* analysis; intersection volume analysis (proposed by the author) and temporal analysis for variance in samples length and position of sub-events. These latest techniques focus on inter class similarities, and the similarity between samples and classes models. These analyses point out that the datasets deploying multiple users' data and fewer feature sets have more complexity, noise and intra similarity. For instance, the FDO_CV dataset is more challenging compared to the FDO_PT dataset. Continuous and isolated recognition experiments' results support the outcome of these analyses.

This study proposed the Recognition Machine (RM) to achieve the aim of the project that is, the recognition of FDO gestures in an on-line manner that incorporates automatic identification of the start/end points of gestures. RM conceptually is an on-line template matching technique. The main idea behind the proposed recognition algorithm is to exploit sequential consistency of the input frames according to class models by using a dynamic programming paradigm and the Markovian process. Sequential consistency or so-called *Score* (S) addresses the similarity between the incremental input data and the class models. *Scores* employ similarity factors (Θ) for each class with an on-line sequential decision process which involves some predictions. The prediction process is a probabilistic estimation of the index of frames (N) in each class (C) which are spatially closest to the input frame (X), given the most recently predicted frame index (V).

The recognition machine is implemented according to the classical pattern recognition framework [87]. The recognition machine (RM) has nine interacting components. The following is a summary workflow of the recognition machine: RM is fed by a sequence of input frames or input band B. The start/end of temporal classes in a band is not known in advance in the case of continuous recognition. Subsequent to acquiring data incrementally from the band ($b(t)$) at each discrete time t, data is pre-processed. Note that pre-processing utilises a limited amount of historical data only for smoothing purposes. Having pre-processed the data (x) is matched with all the channels of classes to obtain the channel degree of membership curves. In each class, the channel degree of membership curves are aggregated to obtain a final degree of membership curve (M), which represents the membership degree of x to the class models. In the frame predictor component, given the most recently predicted frame (V) and M, the next frame (N) is predicted. Then, in the following component *(score estimator)*, scores (S) are estimated based on the cumulative product of the similarity factors (Θ), which consists of the distance function (Ψ), and the membership degree of the predicted frames (M_N). In the final two components, some conditions are checked to see whether a recognition has emerged.

RM addresses the challenge of the gesture recognition problem such as on-line recognition, without a need for explicit marking of the start/end of gestures. The algorithm provides meaningful feedback in case of wrong recognition of gestures, and is tolerant to noise and intra/inter spatial and temporal personal variances.

In order to validate RM, several isolated and continuous gesture recognition experiments were conducted on various artificial and real world datasets apart from FDO_PT and FDO_CV. The results of RM are compared with some on-line and off-line recognition algorithm such HMM, DTW and RNN (Elman). Some of the important findings of these experiments include the following: RM and HMM outperform other recognition algorithms in isolated recognition experiments. RM obtains comparable results when compared to HMM. Elman neural network and DTW (even when it is an off-line algorithm) fail to obtain accurate recognition results. Even though HMM obtains slightly better results compared to RM, further analysis reveals that, since HMM uses only maximum likelihood criteria for recognition, it is superficial and overestimates on unreliable and missing data, whereas, RM rejects any recognition in these circumstances via some built-in heuristics. HMM performs generally better with small number states (3, 5) which do not provide comprehensive feedback for training purposes in case a gesture is performed wrongly. In addition, it is observed that, left to right topologies in HMM obtain better results. Similarly RM is biased towards making frame prediction from left to right.

In the case of continuous gesture recognition, only HMM and RM are deployed over various sentence lengths of FDO_PT and FDO_CV datasets, due to relatively better achievement of these recognition algorithms in isolated recognition experiments. In these experiments, RM outperforms HMM in terms of sentence and word recognition. RM, especially, obtains remarkable results on FDO_PT. Reduced feature set (x and y) and multiple users deployed for data collection in the FDO_CV dataset lead to a relatively poor recognition performance by RM and HMM for FDO_CV sentences. It is observed that while the word recognition rate is approximately constant, the sentence recognition rate decreases as the sentence length increases due to the wrong recognition (insertion error) of certain gestures (*Wave Off*, *Up*, *Affirmative* and *Engage*) in sentences during the transitional phase. This insertion error occurs due to the volume limitation of the input device and the spatial and temporal similarity of gestures. Although HMM obtains good results on isolated gesture recognition, it does not perform as well on continuous recognition as RM (even with grammar knowledge being provided). HMM retains an approximate word recognition rate among different sentence lengths (5, 10 and 20). HMM fails to recognize static gestures and especially non-defined, transition data (NoN_{Ges}). This indicates that HMM fails to represent (model)

a wide range of transition data NoN_{Ges}. Furthermore, in order to test the temporal properties in the continuous case, a periodic control parameter ($-0.5 \leq d \leq 1$) is also introduced. RM achieves reliable results when d is in range of -0.2 and 0.5. Outside this the range, RM considers a sentence to be missing and with unreliable data and rejects any recognition due to built-in control heuristics.

RM is a modular component-based system. In order to validate this, a multilayer perceptron neural network (MLPNN) was developed for the *Frame Predictor* component of RM. The new hybrid system RM/MLPNN was tested on the FDO_PT dataset for isolated gesture recognition. RM/MLPNN achieved reasonable recognition rates for dynamic gestures but in the case of static gestures RM/MLPNN did not achieve the same recognition level of recognition as RM or HMM alone.

7.2 Achievements & Outcomes of the Study

In this thesis we have proposed a gesture recognition algorithm that is suitable for training purposes.The dataset is sufficiently complex and contains static and dynamic gestures. The recognition algorithm (RM) has the following properties:

- Addresses the recognition of dynamic and static gestures.
- Provides isolated and continuous recognition.
- Recognitions are done in an on-line manner.
- Detects automatically the start/end points of gestures in continuous recognition.
- Provides timely feedback for training purposes.
- RM is a modular (component) based architecture, so different techniques can be deployed for components to obtain more efficient overall results.

The overall outcome of the thesis is that, *RM* achieves comparable results which are in agreement with HMM and DTW. Furthermore, the recognition machine provides more reliable and accurate recognition in the case of missing and noisy data. This algorithm also has additional advantage of providing timely feedback for training applications.

For continuous gestures, RM achieves better performance compared to HMM. HMM attempts to model the transitional and undefined data, whereas RM deploys heuristics to reject undefined gestures. The recognition machine addresses some common

limitations of these traditional algorithms and general temporal pattern recognition in the context of FDO training. The recognition algorithm is thus suited for on-line recognition.

In this thesis we have also presented a systematic way to analyse the inherent similarities and complexities of the gesture dataset. It has been found that grid-based normalized spatial features (F_{Grid}) are the most appropriate for reliable recognition of FDO gestures.

In the following paragraphs, the advantages and disadvantages of the proposed algorithm are compared with HMM and DTW.

RM exploits dynamic programming and the Markovian process. It addresses some limitations of existing related recognition algorithms such as Dynamic Time Warping (DTW) and the Hidden Markow Model (HMM). Specifically:

- RM intuitively deals with traditional issues of HMM such as training, decoding and topology. For topology, RM employs an ergodic architecture, which is biased to left to right with a larger number of states. For training and modelling the class, RM accumulates summary statistics with a template-based representation. RM intuitively addresses the decoding process by utilising local maxima on degree of membership curves and some dynamic programming schemes.

- HMM obtains best results in the case of a smaller number of states which do not provide meaningful feedback (observation sequence in decoding) for training. Whereas, RM employs a larger number of states (as it was concluded in [102]) to represent characteristics of gesture, therefore, the decoding process provides more meaningful feedback (observation sequence).

- Weak criteria to announce classification: The probability of models $P(O|\lambda)$ (where O is the observation vector and λ is HMM model), is the only criteria (maximum likelihood $P(O|\lambda)$) to announce a recognition in HMM. This criterion is weak in the case of on-line recognition in which the start and end points of a pattern is not known in advance. Whereas RM employs some heuristics during on-line recognition to prevent premature, unreliable recognition. This feature of RM is novel.

- HMM assumes that observations are managed with some underlying "*hidden*" states. Whereas, in the case of gesture recognition, states are more observable, therefore, RM represents gestures with a large number of *observable* states which is useful for meaningful decoding sequences.

- HMM makes a crucial, assumption, in the case of missing or incorrect data. Whereas RM approaches this situation more carefully, and it rejects any recognition in the case of incorrect and missing data. This is an important feature for training purposes, because RM can detect these mistakes during the performance of gestures in the training session and doing so, RM provides better training.

- HMM tends to model undefined, transition movements. Since these approaches have to model a large number of movements, their success is limited. Whereas RM deploys a heuristic-based rejection scheme to spot undefined, transition data between gestures.

- HMM and RM are based on the first order Markovian process. But RM can be easily extended to a high order Markovian process by utilising (aggregating) some historical degree of membership curves(M).

In addition to these, the proposed recognition algorithm has the following characteristics:

- Template-Based Representation: A comprehensive discussion the surrounding template-based approach is presented for the class modelling. But unlike the traditional template-based approach, in which the distance or similarity between input signals and templates are estimated in terms of Euclidean (or area, volume) distance, in this thesis, the templates are just used for representation. In other words, the similarity between templates and input signals in this thesis are estimated in terms of the distance between predicted consecutive indices. This serves to estimate the similarity between input signals and templates without knowing the start/end points of gestures, which are needed for euclidean distance estimation, in the case of continuous recognition [147].

- General Pattern Recognition Framework: In this thesis, a comprehensive and structured literature review is presented for a general pattern recognition framework and gesture recognition domain. The latest trends and techniques in data acquisition and preprocessing, class modelling/analysing and recognition algorithms for gesture recognition and temporal pattern recognition are discussed with their advantages and disadvantages. Based on the literature review, a general pattern recognition framework is used for the gesture recognition system.

- Features: It is observed that grid-based normalized spatial features (F_{Grid}) are better than angular spatial features ($F_{Angular}$). Similarly, *fuzzy gradient* temporal features obtain better results than gradient features.

- Data Acquisition: Computer vision and tracker-based sensors have their own disadvantages and advantages for data acquisition. It is observed that computer vision-based methods are less cumbersome for the user, but take more resources to extract features from raw data. On the other hand, for more reliable and real time operations, a tracker-based sensor is more appropriate, but it can be burden on the users.

- Variety of temporal datasets considered: In order to validate RM, several temporal datasets, (artificial and real world) were used besides FDO gestures. The author created two type of FDO datasets, the first of which is tracker-based (FDO_PT) and the second one is computer vision-based (FDO_CV). These datasets consist of enough complexity which can occur in real life. The FDO dataset can be downloaded from the following web address with a descriptive article and some MATLAB utility scripts to manipulate the dataset.

  http://personal.rmcs.cranfield.ac.uk/~turand

- Complexity and Similarity Analysis: The study presents a variety of complexity and similarity (temporal and spatial) analysis techniques for datasets. While some of these techniques are novel, some of them have been modified for temporal classes from classical static pattern classification by the author. These analyses point out that multi-user datasets contain more similarity and complexity. For example, FDO_CV and Perrotta have more variance compared to the FDO_PT and Yang datasets.

- Recognition Experiments: The study conducts several isolated and continuous experiments on several temporal datasets with various well-established techniques such as the Hidden Markov Model (HMM), Dynamic Time Warping (DTW) and Elman Neural Networks (ENN) besides the Recognition Machine.

- Isolated Recognition: The followings results have been are observed from the isolated recognition experiments on the dataset.

  - The recognition Machine obtains comparable results with HMM on the isolated gestures.
  - Although, HMM outperforms RM on some dataset, HMM lacks controlled recognition. HMM overestimates on unreliable and missing data, whereas RM rejects premature, missing and unreliable data recognition.

  - Unlike RM, HMMs with the small number of states perform better, although they produce less meaningful feedback for training in the case of error. RM utilises all frames (states) in the trajectory for class modelling. Therefore, RM accumulates better feedback information.
  - Left to Right HMM topologies obtain better results than ergodic topologies. Similarly, even though RM has an ergodic structure, its frame predictor component is biased towards making frame prediction operation from left to right.
  - The Elman Neural Network has the following drawbacks: It does not produce meaningful outcomes, its performance is low compared to other considered recognition algorithms and it takes a long time to train the network.
  - In addition to being off-line, DTW has also the lowest performance amongst the other recognition algorithms.

- Continuous Experiments: Based on the results of isolated recognition, for the continuous gesture recognition, only RM and HMM are considered. Primary results of the continuous recognition experiments are as follows:
  - RM outperforms HMM on FDO_PT and FDO_CV sentences with different sentence lengths [3, 5, 10 and 20]. In these experiments, while word recognition rate is approximately constant, sentence rate is degraded as the sentence length increases.
  - Transition analysis shows that some gestures such as *Wave Off*, *Up*, *Affirmative* and *Engage* cause most of the misclassification, due to their spatial similarity and limitations of input devices.
  - HMM fails to recognize transition data and static gestures. Even with grammar, HMM does not obtain reliable results as much as RM on the transition data.
  - Reduced feature set (x and y) and multiple users deployed for data collection cause the relatively lesser continuous recognition performance for FDO_CV sentences using RM and HMM.
  - RM retains its sentence and word recognition rate in the case of periodic temporal variance.

- Other recognition algorithms such as DTW and recurrent neural networks (RNN) have their own limitations for temporal pattern recognition tasks. For example,

in large RNN, chaotic behaviour emerges which makes it non-trivial to analyse RNN [111].

- Preliminary work on the hybrid system (RM/MLPNN) suggested that hybrid systems could provide a better solution to some issues of temporal pattern recognition (such as function approximation, density estimation, sequential decision) by combining well-established methods in the literature, which obtained reliable results on these issues.

7.3 Future Directions

A broad range of topics relating to gesture recognition and temporal pattern recognition have been presented in this thesis. However, every topics has been covered in the study. Based on the conclusion, outcome of the study and literature review, the following topics require further investigation:

- Hybrid Approach: This thesis proposed a multilayer perceptron neural network (MLPNN) for the *Frame Predictor* component of RM. The hybrid system RM/MLPNN was tested on the FDO_PT dataset for isolated gesture recognition. These experiments produced promising results. Therefore, in future work, a comprehensive study can be done to take advantage of this hybrid system. For example, in terms of the role of neural networks in the hybrid system:
  - MLPNN is just used for the prediction of indices. But MLPNN can be modified to provide membership degrees of input data to class, in addition to prediction [130].
  - Instead of employing MLPNN, other neural networks, such as radial basis functions (RBF) can be used for the *Frame Predictor* component. RBF is referred to as a universal function approximator, therefore, RBF could obtain better index prediction than MLPNN [8].
  - In this study, a feature vector is provided as part of an input vector to MLPNN. (ϑ of $\vartheta + L$ input vector). Since a prediction operation is based on the maxima on degree of membership curve, in further work, for better performance, instead of feature vector, degree of membership curves can be used.
  - In this thesis, RM/MLPNN is validated on isolated gestures. Further work with a hybrid approach should consider continuous gesture recognition too.

- Language Component: RM consists of a *language* component (see figure 5.3), which has not yet been taken into account in this thesis. For simplicity, it is assumed that each gesture has equal probability to appear in any part of the continuous gesture stream. But since FDO is a well-formed sign language, embedding linguistic knowledge into RM will boost the performance. As figure 5.3 shows, the *language* component of RM is linked to the *Score Estimator* component. The language component holds the prior language probabilities (P_{R_C}) for all gestures.

- Real world continuous datasets: In continuous gesture recognition experiments, RM is validated over the synthetic FDO_PT and FDO_CV datasets. As a future work, a real world multi-user continuous dataset should be collected in order to assess performance of RM.

- Timely Feedback Component: RM is able to provide timely feedback during the decoding process for training purposes. But RM does not yet consist of a component yet which utilises timely feedbacks. The feedback component will be responsible for holding the logs of training sessions for a user to provide feedback in case of wrong performance. In addition, the performance of users can be saved and assessed/monitored over a period of time.

Appendix A

Published Papers

In the following sections, published papers out of this thesis are presented.

- Deniz T. Sodiri [1] and Venkat V. S. S. Sastry; *On the Interpretation of Gestures arising in Flight Deck Officers Training*; SISO, 13th Conference on Behavior Representation in Modeling and Simulation; Virginia, USA, May, 2004 [119].

  Errata: Table 1 column four entries (V_2), see table 5.1 in Gesture Recognition Algorithm chapter. Note that above paper used a different heuristic for computing scores.

- Deniz T. Sodiri and Venkat V. S. S. Sastry; *Recognition Machine (RM) for On-line and Isolated Flight Deck Officer (FDO) Gestures*; IJIT, International Journal of Intelligent Technology, Volume 1, pages 138-145, 2006 [121].

- Deniz T. Sodiri and Venkat V. S. S. Sastry; *On-line Recognition of Isolated Gestures of Flight Deck Officers (FDO)*; Transactions on Engineering, Computing and Technology, Volume 13, pages 119-126, Budapest, 2006. [120].

[1] Author of the thesis (Deniz Turan) uses Deniz T. Sodiri as pen name in his publications.

On the Interpretation of Gestures arising in Flight Deck Officers Training

Deniz T Sodiri[1]
Venkat V S S Sastry
Applied Mathematics and Operational Research
Engineering Systems Department
Cranfield University, Royal Military College of Science
Shrivenham, SN6 8LA, U.K.
44-(0)1793 785315
d.turan@rmcs.cranfield.ac.uk, sastry@rmcs.cranfield.ac.uk

Keywords:
Static and Dynamic gestures, Online recognition algorithms, Template matching, Training in Virtual Environments

ABSTRACT: *This paper presents an algorithm for recognition of the real time static and dynamic gestures that arise in a virtual training system for Flight Deck Officer (FDO). Six distinct and commonly used gestures of FDO are considered in this paper. Since, FDO's gestures are arm based, a tracking system, Polhemus FASTRAK is used to acquire position of hands. The recognition system is based on templates and a novel matching/scoring technique. Gestures are modeled in the form of trajectory of angles between hand and suitably chosen local axes. Furthermore, auxiliary templates based on gradient of trajectories of these angles provide a better characterization of the gestures. Any gesture interpretation system has to address two problems – spatial variance and temporal variance. Spatial variance is addressed by constructing a database of gestures that include standard deviation of the sample at each point of the trajectory. Temporal variance of gestures, on the other hand, is addressed during the recognition stage which uses a special matching technique and relies on finding the points, that are equal or close to the input point rather than a sub string matching operation. Recognition algorithm uses the distance between two consecutive matched points to determine the similarity of the input data and gesture template. The matching technique also overcomes the starting/ending and repeatability/connectivity problems associated with gesture recognition. The present algorithm is able to achieve recognition of 98 % over a set of six commonly used gestures.*

1. Introduction

Gestures offer one of the most intuitive methods of interaction with the computer. These are particularly suited in many training applications. One of the training applications that is rich in gesture interactions is the domain of Flight Deck Officer (FDO) Training.

In the United Kingdom, FDOs are trained at the School of Flight Deck Operations (RNSFDO), RNAS Culdrose. RNSFDO comprises several sections, which carry out the training in the individual specialisations. In a trainee program, students are trained to give clear directions to an approaching helicopter that would enable it to land on the flight deck. The current FDO training simulator at RNAS Culdrose requires another person, typically an instructor, to fly the helicopter, in response to the signals given by the trainee. The primary aim of this study is to develop a system to recognize FDO's gestures as part of a training simulation program. Thus as the trainee waves the helicopter in, for example, the helicopter will move accordingly. This will require effective management of two-handed inputs and/or interactions in a virtual environment. A key component of such a system is the automatic recognition of gestures performed by the trainee.

The problem of gesture recognition can be simply stated as: *Given a set of gestures, classify a given trajectory of arms*. A gesture is characterized by a sequence of three-dimensional positions of both the arms, usually collected by a tracker device [12] over a period of time. In the current application, all the gestures take different lengths of time, and each gesture approximately lasts 3 – 5 seconds. A given exercise, for example, landing the helicopter consists of three to four distinct gestures to be performed. One of the tasks of a recognition system is to automatically determine the transition of gestures. Thus the task is to recognize a gesture dynamically, as the trajectory data

[1] Previous known as Deniz Turan.

is available incrementally. In this paper, we refer to this problem as *on-line recognition*.

The generic problem of gesture recognition has been studied in the literature with varying degrees of success, using neural networks [4], [8], hidden markov models [14], [16], dynamic time warping [2] and pattern recognition methods [1], [3], [5]. The problem of on-line recognition has received little attention. In this paper, the authors propose a heuristic algorithm based on pattern matching that is robust against both temporal and spatial variation in the gesture input.

The paper is organized as follows: In Section 1.1, we provide a brief background to human gestures in general. The gesture recognition problem is presented in Section 2, along with a formal definition. The process of data collection and the formulation of the *templates* is described in Section 3. The details of the proposed algorithm together with an illustrative example are given in Section 4. The results of the proposed algorithm and its performance, based on two experiments are presented in Section 5. The final section summarizes our observations, and the scope of the proposed algorithm.

1.1 Background

A gesture consists of intentional movement of arms for a period of time to convey a pre-assigned meaning. These gestures can be classified into two broad categories – static and dynamic. Static gestures or *poses* are those whose trajectories over a period of time remain the same, while dynamic gestures have trajectories that vary in time over the duration of the gesture.

McNeill [9], [10] and Kendon [7] define three phases of a dynamic gesture: preparations (pre-stroke), stroke, retraction (post-stroke) (See Figure 1). Furthermore, Quek proposed a set of rules (similar to Kendon's) to formalize how a dynamic gesture is performed [13]. This set includes the flowing six rules:

- Gestures are contained in movements that start with a slow initial move from the rest positions, continue with a phase with substantially increased speed (the stroke), and end by returning to the rest.
- The hand assumes a particular configurations during the stroke
- Slow motion between resting positions are not gestures.
- Hand gestures should be constrained within a certain volume or workspace
- Static hand gestures require a finite period of time to be recognized
- Repetitive gesture can be gestures.

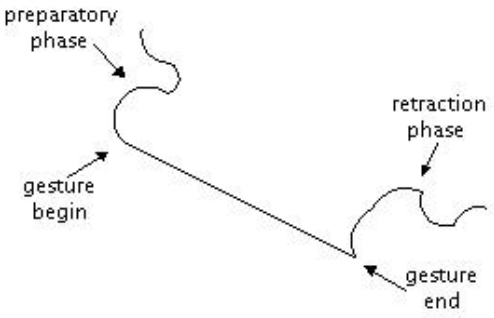


Figure 1 Three phases of a gesture (after [7]).

The gesture recognition system implemented in this paper is similar to that of a generic recognition system [11]. First, inputs are obtained through sensors in the form of raw data. Then, the raw data is processed to detect and extract attributes, features either in order to construct the gesture models or to be used in recognition algorithm. The gesture models are constructed in advance. Final step of the system is to recognize input data through gestures models by using recognition algorithm and predefined rules, language. Figure 2 illustrates this process.

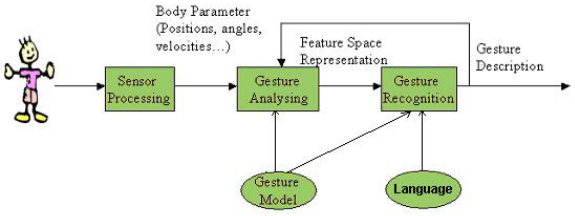


Figure 2 Components of a gesture recognition system.

A set of six gestures that are frequently used in FDO training are shown in Figure 3. Corresponding trajectories obtained by the tracking device are shown in Figure 4.

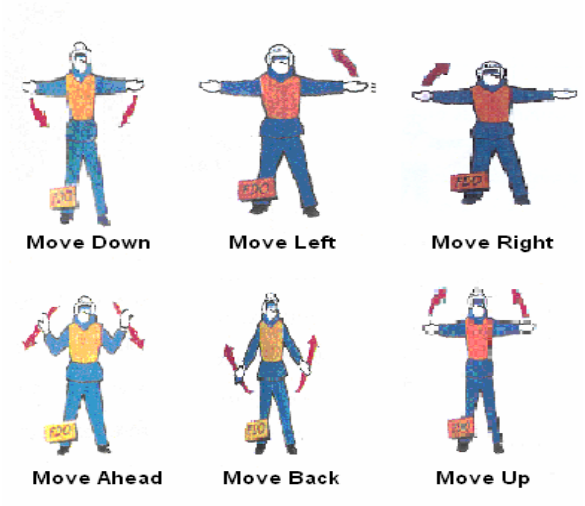


Figure 3 Commonly used gestures in FDO training.

2. The Problem

A formal definition of the gesture recognition is presented in this section, along with the notation used in this paper. The formal definition will also help us understand the subtleties of gesture recognition. The problem can be specified as 5-tuple *(C, P, G, U, D)* where:

- *C* is the set of gesture model templates with cardinality of N,

$$C = (G_1, G_2, \cdots, G_N)$$

- *P* is the set of period/length of each gesture,

$$P = (p_1, p_2, \cdots, p_N), \quad 0 < p_1 \cdots p_N \leq T$$

T is the length of an experiment. Period of each gesture is small compared to the length of the experiment (*T*).

- *G* is a gesture model and consist of units, indexes which indicates the time order.

$$G_i = \{U_{i,1}, U_{i,2}, \cdots, U_{i,p_i}\}, \quad 0 < i \leq N$$

- *U* is the feature vector (*1 x M*) that is used to construct a gesture model. A unit consists of attributes or component (*f*). Note that, in the remainder of the paper, attribute, component and feature are used interchangeably.

$$U = [f_1, f_{2,\ldots}, f_M]$$

- *D* is the historical set of incremental data (*d*). Using this data incrementally, we need to identify the gesture to which it belongs.

$$D = \{d_t\} \quad , d_t \in U \quad \text{for} \quad 0 < t \leq T$$

Although *D* contains all the historical data, often only the most recent data is used in practice.

Example

We clarify the notation with the help of a fictitious set of gestures. For illustration, the gestures are characterized by a single feature. This example is also used to illustrate the proposed matching technique.

$$\begin{aligned} C &= (G_1, G_2, G_3) \\ P &= (5, 9, 7) \\ G_1 &= \{-1, -1, 1, 1, 0\} \\ G_2 &= \{1, 1, -1, -1, 0, 0, 1, 1, 1\} \\ G_3 &= \{-1, 1, 1, 1, 0, 0, 0\} \\ f &\in (-1, 1, 0) \\ D &= \{-1, 1, 1, 0\}+ \end{aligned}$$

where the notation {…}+ indicates that the trajectory is repeated indefinitely. The above example includes three gestures with periods 5, 9 and 7 respectively. Each gesture contains the features whose elements are taken from the set (-1,1,0).

Note that the input data, *D* is shifted version of G_1 that is repeated indefinitely. It is expected that any matching algorithm should recover this gesture as G_1.

3. Construction of Gesture Templates

Recall that a gesture consists of a trajectory points for both left and right arm (See Figure 4). It is convenient to map this data to a feature space, in which each gesture can be represented as a vector or a template [6]. The feature space consists of three-dimensional angular displacements for each arm expressed in a local coordinate system (See Figure 5), and their gradient information. In order to construct a template for a gesture, each gesture is performed several times under varying conditions. All the data is collected using the same person.

Each point on a trajectory for each arm is mapped to a feature vector that consists of seven components – three angles, and their standard deviations and dominant gradients. Instead of representing gradient information for all the angles, it is sufficient to include the gradient information for the angles that vary most. This is particularly true, in the present application as most of the gestures are performed in a plane. Thus a feature vector consisting of 14 components represents each gesture. For example, angular features for *Move Left* gesture are shown in Figure 6 (for clarity, the gradient features are not shown).

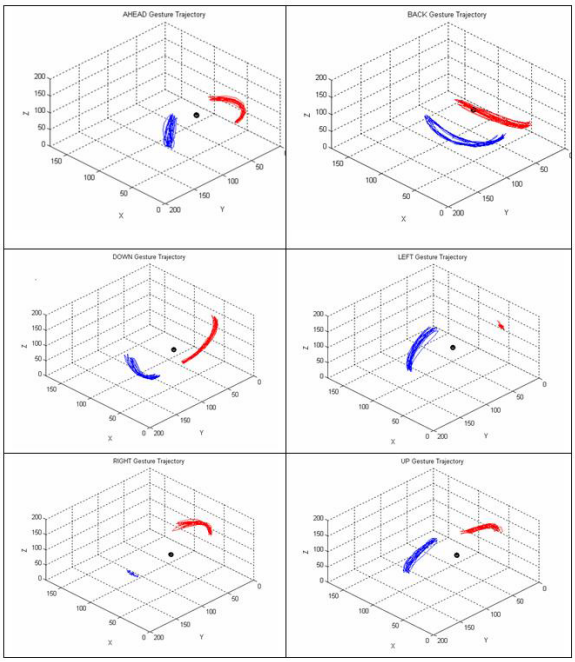


Figure 4 Trajectories of all six gestures.

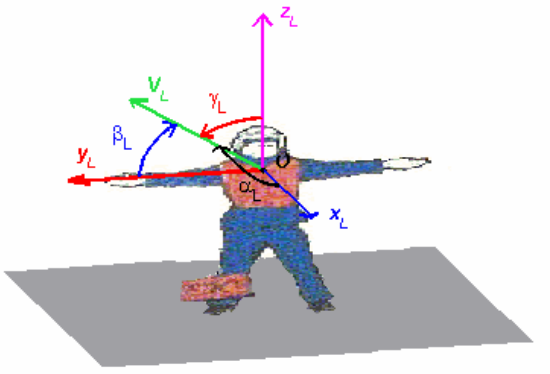


Figure 5 Local coordinate system.

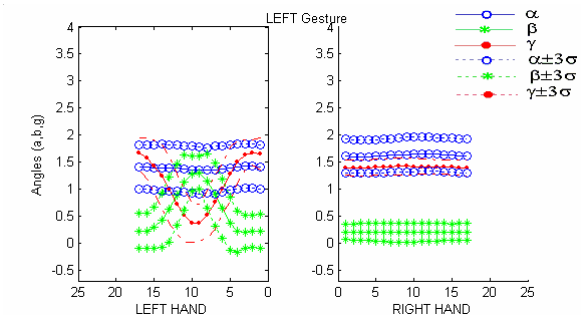


Figure 6 Move Left gesture. Note that the right hand posture remains static while the left hand moves up and down.

4. Proposed Algorithm

Prior to elaborating the recognition algorithm in detail, it would be helpful to motivate the algorithm with the example presented in Section 2.

Given a gesture template, G_t and an incremental test data *D*, our task is to find if *D* matches G_t. As it can be seen from the Figure 7-a, the test data *D*, is similar to the gesture template, G_t except for a small initial horizontal shift. This, in practice, accounts for the gesture starting a few second later. But both gestures convey the same meaning.

The proposed approach to solve the problem is briefly depicted in Figure 7, b-d. Let us assume, the first datum of *D*, d1 matches data of G_t at the points *n* and *k* (See Figure 7,b). If *d1* is assumed to be at the beginning of the gesture, *n* would be taken as the first selected point. Then, the next data, *d2*, matches the data of G_t at time *m* and *r* as shown in the Figure 4,c. Since *d2* succeeds *d1*, at least one of the matched points of G_t, *(m, r)* is also expected to be succeeding the previous selected point, *n*. Therefore, m is selected as the second next point. Similarly, for the next datum *d3*, which is the successor of *d2*, one of the matched points *(l, p)* is also expected to succeed the previous match *(m)*. Hence the next candidate match would be *(l)*.

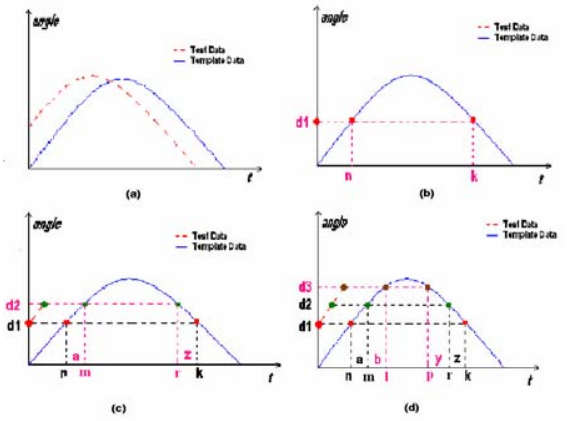


Figure 7 Illustration of matching technique. For visual clarity, the template and the input data are shown as continuous curves.

t	D	V_1	V_2	V_3	ω_1	ω_2	ω_3	S_1	S_2	S_3
0	-	0	0	0	0	0	0	0	0	0
1	-1	1	3	1	1	3	1	1	3	1
2	1	3	7	2	2	4	1	3	7	2
3	1	4	8	3	1	1	1	4	8	3
4	0	5	6	5	1	7	2	5	15	5
5	-1	1	3	1	1	6	3	6	21	8
6	1	3	7	2	2	4	1	8	25	9
7	1	4	8	3	1	1	1	9	26	10
8	0	5	6	5	1	7	2	10	33	12
9	-1	1	3	1	1	6	3	11	39	15
10	1	3	7	2	2	4	1	13	43	16
11	1	4	8	3	1	1	1	14	44	17
12	0	5	6	5	1	7	2	15	51	19
13	-1	1	3	1	1	6	3	16	57	22
14	1	3	7	2	2	4	1	18	61	23
15	1	4	8	3	1	1	1	19	62	24
16	0	5	6	5	1	7	2	20	69	26

Table 1 Computation of scores using the recognition algorithm.

In fact, the number of the matched points can be reduced by considering the gradient of the d_i and G_t. For example, when matching the second point, d2, since the gradient is increasing, ϕ_i, the point *r* can be eliminated because at the point *r* of G_t, the gradient is decreasing (ϕ_d). Using the gradient feature can reduce approximately half of the matched points. Thus it is advantageous to augment the templates with gradient features.

The main idea in the proposed algorithm is that, the similarity of the two signals can be expressed in terms of the widths of matched indices of two consecutive input data. In the present example, the matched indices are *n, m, l*; and the widths are *a := m-n*, and *b := l-m*. In fact, cumulative sum of the widths of matched indices thus provides an overall measure of similarity of the input signal to that of the template. Using this measure, a lesser value indicates more similarity. In the remainder of the paper, the cumulative sum of the time or index interval is referred to as Score and is used as a measure of similarity. Note that this definition preserves similarity under horizontal shifts, but not vertical translations. Effects of vertical translations are taken into account by introducing a bandwidth in the definition of the templates.

It is important to note that the matching operation is not a sub string matching. A sub string match would retrieve the location of the searched string. In the present matching operation, we are also concerned with elimination of start and end point of gestures and address the problem of temporal variance.

Note that, the values *a, b, y, z* are used for emphasizing that there may be a temporal variance during the gesture when it is performed. Performing the gesture slowly or fast may be one of the reasons for temporal variance. One of the most important advantages of the algorithm is to overcome the temporal (intra and inter personal variance and noise) issues by means of searching the possible matched points monotonically, incrementally and partially rather than matching the two signals point by point entirely and offline.

Another problem the algorithm addresses is that of the issue of start/end position of a gesture. The monotonic, incremental and partial matching enable us to dismiss this issue since algorithm is not based on the classical matching method which needs the start and end point of the signals to match point by point and offline.

Table 1 shows the details of the manual operation of the method, where V_i is denotes the matched time step or index of the gesture, G_i and ω_i is the distance between two consecutive selected indices for gesture G_i. The score S_i is the cumulative sum of ω_i. Consequently, a smaller value of S_i, indicates more similarity between the input data d and the features of G_i. From the table it is clearly seen that, G_2 is eliminated in the first few iterations of the algorithm. To clearly distinguish the gesture from the remaining two, the algorithm needs to iterate further. Empirically it can be shown, that the algorithm eventually converges on a unique gesture, and hence in the present example to G_1.

The above example is based on only a single attribute. In the present study gestures are more complex, and the recognition algorithm needs further refinement. The above matching algorithm can easily be extended to feature vectors. The key points of this matching process are summarized below:

- Preparation of feature vector: Input feature vector consists of angles and gradient of the angles.

- Possible Matched Angle and Gradient Points: Find set of all possible matched points for each attribute.
- Intersection of Matched Angle and Gradient Points: Since, multiple attributes are used to construct a gesture, intersections have to be done over the set of the matched indices to obtain a final set of matched points. Indices are matched up to a neighbourhood parameter, which can be varied to study the performance of the algorithm.
- Finding The Next Point: Choosing the next point from the intersected matched point and current point. Next point (*N*) is the point which is the most closest to the current point (*V*) in the final intersected set.
- Compute The Scores and Move: Compute the score, *S*, which is the difference between current point and the next point (S←*N*-*V*). And then, move to the next point (V←N).
- Assess The Scores: Analyze the current point , *V,* and score, *S*, to make a decision whether a gesture is recognized or not.

Further details of the algorithm are described in [15].

5. Results

In this Section, we consider two experiments. In the first experiment, the test input consists of trajectories for all the six gestures discussed in this paper. The second experiment is designed to investigate the effects of gesture transitions.

For convenience, the trajectories for each gesture are recorded to a file. This file, hereafter is referred to as Training Data File (TDF). To account for variations in starting/stopping of a gesture, each gesture is performed several times continuously. These gestures are also performed at various paces. All the data is collected using the same person.

The gesture input data is further partitioned into cycles. A cycle for a gesture refers to all the trajectories/features starting from the initial position until the gesture returns to its initial position. Note that of the six gestures considered in this paper, two of the gestures involve practically no motion in one of the arms (*Move Left and Move Right*). For these gestures, we use the dynamic part of the gesture to determine the cycle length.

5.1 Experiment 1

In Experiment 1, the test input data consists of several repetitions of all the six gestures. The number of repetitions of each gesture is shown in Table 2 (column 2). Note that a significant portion of the input data is used in the construction of the templates. The data used in this experiment in one significant aspect. That is, it contains gesture input that is not used in the construction of the templates. The number of these data samples is shown in column 3 of Table 2, and accounts for 10% of the sample size across the gestures. Note that the input data is processed further while constructing the templates. Thus the inclusion of non-training data provides a rigorous test for the robustness of the algorithm. As a result, we expect a small drop in the number of successful recognitions.

Gesture	**#Cycles**	**#Non-TD**	**No. Misclassified**	**% Correct classifications**
Ahead	217	32	1	99.5
Back	146	27	0	100
Down	157	16	0	100
Left	164	16	0	100
Right	169	25	0	100
Up	124	13	0	100

Table 2 Results of Experiment 1.

Table 1 shows the result of the experiment 1. The table shows the gesture, total number of the cycle for the gestures, the number of misclassifications, and the percentage of successful recognitions. As can be seen from Table 2, the algorithm is able to correctly classify all the gestures. However, the algorithm fails on one occasion. This misclassification is traced back to two situations – 1) non-training data for the *Ahead* gesture and 2) synchronization errors of two hands.

When the non-training data cycle (non-TD) is added to the training data, we no longer observe any misclassifications. This suggests that there is scope to improve upon our method of constructing templates. Thus we can conclude, that the algorithm is not only able to recover the trained examples, but also classify the unseen data correctly almost without any error.

5.2 Experiment 2

In the second experiment, the test data consists of combinations of gestures and non-gestures in a specified order (gesture cluster). This data set is more realistic than the first experiment, in that it represents a typical exercise in a training session. The aim of the experiment is to investigate the performance of the recognition algorithm and the models due to transitional effects of gestures.

The test data consists of gesture clusters. A gesture cluster may consist of gesture repetition as well as a single instance, and non-gestures. Table 3 contains three gesture clusters that are punctuated by non-gestures. Within a gesture cluster, gestures are normally performed continuously. Gesture transitions contain small pauses lasting approximately two seconds, which will be classified as non-gestures. The input gestures labeled as *NonGestures* are deliberate. In the first gesture cluster, for example, the *Down* gesture is performed three times, then the *Up* gesture is performed four times and so on until a non-gesture. Before performing each gesture cluster, the hands are stationed at the start point of the gesture. While constructing gesture clusters, the similarities of the gesture are taken into account to provide sufficiently challenging data set. For example, the third cluster contains *Down* and *Ahead* which are very similar. In other words, in order to make the test data sufficiently challenging, similar gesture is performed continuously and different number of times. Note also that the data set accounts for 45% of gesture transitions.

The test data consists of non-gesture movement as well, and the algorithm is expected to recognize them as 'non-gesture' or 'ambiguous gesture'. This accounts for meaningless motions performed by an FDO during a training session.

Table 3 shows the result of the second experiment. The first column represent the gesture cluster in order, the second one, total number of the gesture cycles in each gesture cluster and the last column represents the number of misclassifications. As Table 3 indicates, the result of the algorithm and models is successful as in the first experiment. The system is able to recognize the approximately 98% of the gestures. In general, the only misclassifications occurred during the transition from non-gesture to the starting point of a gesture. In spite of these issues, reasonable gesture recognition is obtained.

Gesture	**#Cycles**	**#Misclassifications**
Down	3	0
Up	4	0
Right	2	0
Left	3	0
Ahead	4	0
NonGesture	11	1
Down	1	1
Up	1	0
NonGesture	2	0
Back	2	0
Down	2	0
Up	3	0
Left	5	0
Down	2	0
Ahead	3	0
Right	3	0
Up	4	0
Back	6	0

Table 3 Results of Experiment 2.

6. Conclusions

The paper presents a complete gesture recognition system in which static and dynamic gestures are recognized using an on-line algorithm. Template based modeling, and ad-hoc algorithm based on a special matching technique and scoring are used to recognize the gestures. Dynamic and static gestures are modeled via angular templates, deviation of angles and gradient of the most fluctuating angular template. The particular definition of templates used in the present study addresses spatial variance in a neat way. The matching technique also addresses the problems of finding starting-ending point of a gesture; problems of temporal variance, and problems of transitions between gestures.

Results of two experiments are presented. On the test data (that is not used in the construction of the templates), the algorithm is able to recognize 98 % of the input gestures. The misclassifications are traced to transitions from one gesture to another gesture, particularly from a non-gesture to another gesture.
The performance of the algorithm on a larger set of 18 gestures is commensurate with the sample data set. Future studies will investigate the scope and performance of the algorithm as the size of the gestures is increased.

8. Appendix A

In this Appendix, we present results for an extended set of 18 gestures. The additional gestures are shown in Figure 8. In order to understand the spatial variation in the data set, we have computed the standard deviation at each time step across all the samples for a given gesture. For brevity, we have indicated the maximum and minimum of these standard deviations for angular attributes. This variation is shown in Table 5.

The results from the application of the algorithm are shown in Table 4. Note that the number of repetitions of each gesture is shown in column 1. Once again, the algorithm correctly identifies all 18 gestures. The average scores obtained when a gesture is presented to the algorithm are shown in Table 6. The results of Experiment 2 using the extended gesture set are

Recognition Machine (RM) for On-line and Isolated Flight Deck Officer (FDO) Gestures

Deniz T. Sodiri, and Venkat V S S Sastry

Abstract — The paper presents an on-line recognition machine (*RM*) for continuous/isolated, dynamic and static gestures that arise in Flight Deck Officer (FDO) training. *RM* is based on generic pattern recognition framework. Gestures are represented as templates using summary statistics. The proposed recognition algorithm exploits temporal and spatial characteristics of gestures via dynamic programming and Markovian process. The algorithm predicts corresponding index of incremental input data in the templates in an on-line mode. Accumulated consistency in the sequence of prediction provides a similarity measurement (Score) between input data and the templates. The algorithm provides an intuitive mechanism for automatic detection of start/end frames of continuous gestures. In the present paper, we consider isolated gestures. The performance of *RM* is evaluated using four datasets - artificial (W_TTest), hand motion (Yang) and FDO (tracker, vision-based). *RM* achieves comparable results which are in agreement with other on-line and off-line algorithms such as hidden Markov model (HMM) and dynamic time warping (DTW). The proposed algorithm has the additional advantage of providing timely feedback for training purposes.

Keywords — On-line Recognition Algorithm, Isolated Dynamic/Static Gesture Recognition, On-line Markovian/Dynamic Programming, Training in Virtual Environments.

I. Introduction

RECENT advances in technology have put computers at the centre of daily life. Yet, lack of naturalness in the interaction methods with computers still encumber users. From that perspective, gesture, one of most used means of the communication among humans, has been investigated for potential interaction scheme in some domains in the recent decades.

In the context of human computer interaction, a gesture is defined as "... expressive, meaningful, body motion -i.e., physical movement of the fingers, hands, arms, head, face or body with the intent to convey information or interact with the environment." [19].

The present study is motivated by a need to recognize automatically Flight Deck Officer (FDO) gestures for training purposes. FDOs are in charge of ensuring craft and maintaining operational status and readiness. For example, safe conduct of flight deck operations for helicopter such as launching and recovering on board are some of their responsibilities. This study aims to remove the role of the instructor, by automatically recognizing FDO's gestures to provide natural means to interact with the virtual environment during training sessions. In addition to that, a feedback has to be provided to the trainee about his/her performance for training purposes.

Gesture recognition problem is akin to temporal pattern recognition problem. It has common properties with other temporal pattern problems such as speech and hand writing recognition. For these problems and gesture recognition, a wide range of recognition techniques have been proposed with various success rate. Neural network [5], [20], [12], [21], dynamic time warping, [11], [4] , hidden markov model [10], [16], [13] and some other ad hoc methods [8] are among these techniques. But most of these efforts do not readily lend themselves to on-line recognition. During the last decade, Hidden Markov Models (HMM) and its variations, hybrid and extensions thereof, have attracted a huge attention. Subsequent to the development of HMM toolkits such as HTK [18], several applications have been developed in temporal pattern recognition domain.

In this paper, the authors propose an on-line recognition machine (*RM*) for dynamic and static gesture under a generic recognition framework. *RM* consists of classical pattern recognition components such as preprocessing, modelling/analysis, language and recognition algorithm. The characteristic features of the proposed *RM* are : its ability to address inter/intra personal spatial and temporal variance, ability to deal with both dynamic and static gestures in a continuous or segmented gesture streams, determine the start and the end of a gesture as part of recognition task and construct a base for additional feedbacks, assessments for training purposes. The recognition algorithm conceptually is an on-line template matching technique and it involves aspect of dynamic programming technique and Markovian process.

The remainder of the paper is organized as follows: A formal definition of problem and related issues are presented in Section 2. Then, an overview of the proposed recognition machine and its components are elaborated in Section 3. The components of *RM* are detailed under two subsections - gesture modelling/analysis and recognition algorithm. In Section 4, a comparison of the proposed algorithm with HMM and DTW and other possible techniques for the components are discussed. The performance of the proposed algorithm is evaluated using four data sets - artificial data set [9] , FDO data (tracker and visions-based) sets and hand motion data set ,Yang [6], in Section 5. In the last section, conclusion and future work are presented.

II. Definition of the Problem

The task at hand needs to address some of the following issues: Spatial and temporal variance; repeatability and con-

Deniz T Sodiri is a PhD student in Defence Academy of UK, Cranfield University, Swindon, UK (e-mail: d.turan@cranfield.ac.uk).

Dr. Venkat V S S Sastry is a senior lecturer and director of scientific computing in Defence Academy of UK, Cranfield University, Swindon, UK (e-mail:v.v.s.s.sastry@cranfield.ac.uk)

nectivity; start/end frame detection [13]. While spatial variance accommodates shape, rotational and translational variations in space, temporal variance accounts for velocity changes. In addition to these variations, in a continuous gesture stream, like in a sign language, consequently multiple repetition of gestures or transition from one gesture to another gesture, makes the recognition task non-trivial as it involves detection of completion of a gesture (segmentation). Specifying the start and end frame in advance or during performance is also a burden. It interferes with the naturalness of the interaction. On the other hand, an automatic prediction of start/end frames of gestures makes problem more challenging . Note that, for example, in speech recognition, silence is used as a delimator for start and end of a word. In gesture recognition, spatial and temporal properties of unintentional or undefined movement and genuine gestures are potentially similar. In addition to these, the problem is sensitive to environmental noise as well.

Taking into account these issues, the problem can be stated as a five-tuple *(C, L, H, F, B)*. C accounts for gesture models with cardinality of ϖ. Thus $C = (C_1, C_2, C_3, \cdots C_{\varpi},)$. Length or period of gesture models are represented with the set L ($L = (l_1, l_2, l_3 \cdots l_{\varpi})$). Each gesture may have different period. A class C_i, consists of η number of channels ($H_{i,j}$) each of which constructed with a sequence of a feature f_j from the feature or alphabet set ($F = \{f_1, f_2, f_3 \cdots f_\eta\}$). For convenience, features at a certain time is referred to as a frame in this article and sequence of frames determine a gesture. B is an η dimensional input bands or channels which consist of the historical set of incremental frames (b_t). Each cell or unit in the band, contains one frame. Since, the frames are obtained incrementally, at a time *t*, only the present and previous data on the band are accessible. Thus, $B = \{b_1, b_2, b_3, \cdots, b_t\}$.

Similar to Pavlovic's [14], a temporal class or gesture can be defined as follows :

A temporal class C_i, is a trajectory of frames in the form of channel templates ($H_{i,1\cdots\eta}$) in a η dimensional feature space F , over a defined time interval l_i.

In the present study, the gestures of interest are either static or dynamic. Static gestures are those that have certain poses or configuration where trajectories remain approximately same for the period (l_i). On the other hand, dynamic gestures are the motion whose trajectories vary spatially with time. Using the above notation, the problem can be stated as:

Given a sequence of input frames (and hence B) incrementally, develop an algorithm or a recognition machine (RM) to recognize the gestures to which it belongs.

III. Recognition Machine

Recognition machine is implemented according to the classical pattern recognition framework [14]. Figure 1 illustrates the components of RM. A brief outline of the recognition machine is as follows: The recognition machine (RM) has nine interacting components. RM is fed by a sequence of input frames or input band B, of which properties are defined in the problem statement. Subsequent to acquiring data incrementally from the band ($b(t)$) at each discrete time t, data is pre-processed. Then, pre-processed data (x), is matched with all

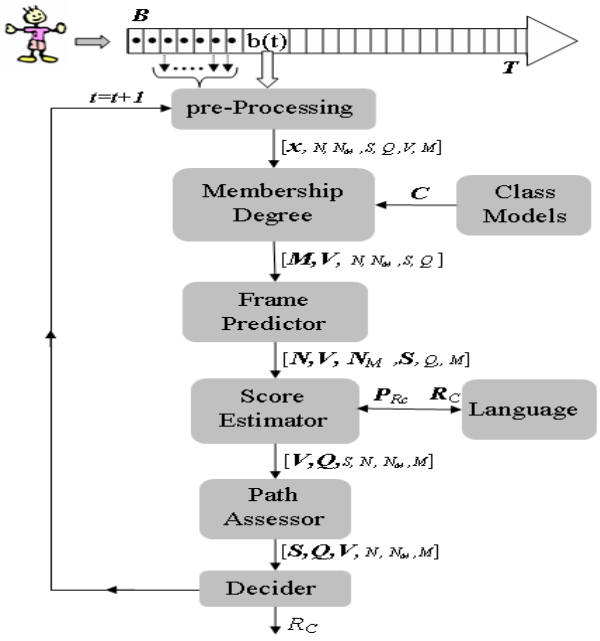


Fig. 1 Components and flow diagram of a recognition machine (C=Class Models; b, *X*=raw and processed current input frames; M=Membership degrees; V=Current Predicted Induces of Frames; N=Next Predicted Indexes; N_M=Membership Degree of Next Indexes; S=Scores; Q=Path Assessors; R_C=Recognized Class)

the channels of classes to obtain channel membership degree curves. In each class, channel membership degree curves are aggregated to obtain a final membership degree curve (M), which represents the membership degree of x to the class. In the frame predictor component, given the most recently predicted frame (V) and M , next frame (N) is predicted. Then, in the following component *(score estimator)*, scores (S) are estimated based on cumulative product of similarity factors (Θ), which consists of distance function (Ψ), and membership degree of predicted frames (M_N). In the final two components, some conditions are checked whether a recognition has emerged. The components of RM are further elaborated in the following two subsections: Analysis & Modelling and Recognition Algorithm. Note that, capital letters , such as V, M, N, S, M_N are used to denote a component related to all gestures. A subscript on these letters, V_i, M_i, N_i, M_{N_i} indicates the data related to the i^{th} class.

A. Analysis & Modelling

Analysing & Modelling part of RM is responsible for acquiring raw data from source, storing raw data on the input band (B), preprocessing, extracting feature and modelling classes. The main purpose of modelling stage is to build the templates for the gestures under consideration.

Recognition machine is fed by a η dimensional band (B). Typically, contents of B are obtained from source devices via input devices. Preprocessing component carries out smoothing, transformation and feature extraction tasks in that order. After smoothing the raw frames $b(t)$, necessary transformations are performed.

A template accumulates the trajectory of a class channel with two statistical parameters , mean (μ) and standard deviation (σ) at each discrete time step. It is assumed that at any discrete time, the underlying distribution is gaussian. The steps of constructing a template are described as follows: First step is to decide comprehensive and distinctive spatial and temporal feature vector(F). Due to temporal variance, training cycles have various lengths. Average length of all the training cycles of a class is used as the period of the class (l_i). Having estimated the period for the classes, then, all the training cycles are either stretched or compressed to the length of the period (L). In addition to that, sub events in the training cycles are aligned to occur at the same indexes during compression and stretching. Note that, these operations are performed only while constructing the templates. Finally, the aligned , stretched and compressed training cycles are used to construct the templates by using summary statistics (mean and standard deviation).

B. Recognition Algorithm

The recognition algorithm conceptually is an on-line template matching technique. The main idea behind recognition algorithm is to exploit sequential consistency of the input frames according to class models by using dynamic programming paradigm and Markovian process. Sequential consistency or so-called $Score$ (S) addresses similarity between the incremental input data and the class models. *Scores* employ similarity factors (Θ) for each class with an on-line sequential decision process which involves some predictions. The prediction process is a probabilistic estimation of the index of frames (N) in each class (C) which are spatially closest to the input frame (X) , given the most recently predicted frame index (V).

The following two metrics can be considered as similarity factors: A function of the distance ($\psi(.)$) between consecutive predicted frame index (N), and a membership degree of input frame to the predicted frames (M_N). The distance function ($\psi(.)$) utilizes the consistency along the sequence of predicted input frames index (N). A monotonic , steady incremental behaviour in the sequence of the predicted frame indexes points out consistency or similarity between the input frames and the class model of interest. In other words, small positive distances (Δ) between the consequent predicted frame indexes shows a possible recognition. A detailed and exemplified discussion of the distance function motivation can be found in [17]. In fact, the distance function is a type of radial basis function. Therefore, gaussian basis function ($e^{-\frac{\Delta^2}{2}}$) is used in this paper [1]. The similarity factors (distance function $\Psi(\Delta)$ and membership degree M_N) *score* of class C_i (S_i) are estimated as follows:

$$\begin{aligned} \Delta_{i,t} &= N_{i,t} - V_{i,t} \;\; ; \;\; \Psi(\Delta_{i,t}) = e^{-\frac{\Delta_{i,t}^2}{2}} \\ \Theta_{i,t} &= M_{N_{i,t}} \Psi(\Delta_{i,t}) = M_{N_{i,t}} e^{-\frac{\Delta_{i,t}^2}{2}} \\ S_i &= \prod_{t=1}^{T} \Theta_{i,t} \end{aligned} \quad (1)$$

Membership degree curves (M_i) estimation involves a partial on-line template matching operation ($M_i = P(X|C_i)$). It estimates the probabilities (M_i) of the input frame (X) belongs to the frames of each class model (C_i) in two stages. The first stage is a low level channel membership degree ($M_{i,j} = P(X_j|H_{i,j})$) estimation. The second phase is aggregation of channel membership degrees ($M_{i,1\cdots\eta}$) in order to obtain ultimate class membership degree (M_i). Membership degrees ($M_{i,j}$, M_i) are computed as follows:

$$M_i = \sqrt[\eta]{\prod_{j=1}^{\eta} M_{i,j}} \;\; ; \;\; M_{i,j} = e^{-\frac{(X-H_{\mu_{i,j}})^2}{2H_{\sigma_{i,j}}^2}} \quad (2)$$

where $H_{\mu_{i,j}}$ and $H_{\sigma_{i,j}}$ correspond to statistical mean and standard deviation parameters of the channels respectively. The parameter $M_{i,j}$ accommodates intra-membership degree redistribution. Intra redistribution regulates membership degrees among the indexes which are aligned during training phases because of temporal variances of sub events.

Frame predictor component predicts possible position of the input frame in the class templates given the membership degree curve (M_i) and most recently predicted frame index. Index of the local maxima (N_i) travels within the membership degree curve from beginning to end with a monotonic and increasing order, if the input data belongs to the classes. The input frame creates a local maxima in the membership degree curves wherever the frame is closer to the template frames. This characteristic of membership degree curve, namely position of the local maxima, serves to predict possible frame index. In the cases of multiple local maxima in the membership degree curves , nearest local maxima in the neighbourhood of the most recently predicted frame index is considered.

On-line prediction and piecewise matching operation paves way to resolve issues of temporal variance and identify *start/end* of a gesture. For each input frame, corresponding frames in the class templates are predicted. Therefore, these operations enable to detect start and end frame of gesture and adapt to temporal variances.

Even though , *score* (S) is one of the major measurements indicating similarities, it does not accommodate any information in itself what time or in what condition it is appropriate to declare a recognition. Order of predicted indexes or the path of observed indexes would help more accurate declaration. These operations are employed in the *Path Assessor* component. It prevents premature or wrong recognition and provides auxiliary information to the *decider* component, in order to evaluate all status and declare a recognition if one has emerged.

It is stated that in a consistent recognition, the predicted frame index N_i must be in an order, namely follow a monotonic increasing path from beginning to end within the membership degree curve (M_i). In this study, it is assumed that, M_i is consolidated by four consecutive part or milestones, Q_i=$\{q_{i,1}, q_{i,2}, q_{i,3}, q_{i,4}\}$ which are referred to as *path* in the rest of the paper. Each part occupies a quarter of class period ($0 < q_{i,1} < 0.25{*}l_i < q_{i,2} < 0.5{*}l_i < q_{i,3} < 0.75{*}l_i < q_{i,4} \leq q_{i,4}$). This component ensures that, all the parts are observed with a monotonic increasing order from $q_{i,1}$ to $q_{i,4}$. Note that,

$q_{i,4}$ is followed by q_1 for continuous recognition. If any jump occurs in the path , for example from $q_{i,1}$ to $q_{i,3}$ or $q_{i,4}$ rather than $q_{i,2}$, score and path will be reset ($S_i = 0, Q_i = q_{i,1}$).

In addition to these, it is also expected that, a sufficient N_i (a threshold, at least, 10 % of class period, $l_i/10$) has to be observed in each part to build a confidence for the observed path. Therefore, this component also holds the number of $N_i \neq V_i$ observations (path age QA) for each path part ($qa_{i,1}, qa_{i,2}, qa_{i,3}, qa_{i,4}$).

Having accumulated current status (path assessor, scores), now, it can be decided whether or not a recognition has emerged. Following conditions have to be met for an on-line recognition (R_C): 1- The path (Q_i) has to be in a sequential order in terms of the predicted frame indexes and N_i must be in the final part ($Q_i = q_{i,4}$). 2 - The duration in each part QA_i must be greater than a threshold eg. 10 % of class period. 3- S_i has to be maximized among the classes of which the paths include the final part ($Q_j = q_{i,4}$)

IV. Discussion

A class C_i can be thought of as a chain of L_i states (s_j), each of which consists of η channels. Approaching the template as a chain of states enable us to make the analogy between the proposed recognition algorithm and widely used algorithms such as Hidden Markov Model (HMM) and Dynamic Time Warping (DTW). HMM is a stochastic finite state automata, in which emission of observations and transitions between states are expressed in a probabilistic manner [2], [15]. DTW is an off-line template matching algorithm, in which time dimension is warped monotonically and increasingly in a window bandwidth , in order to minimize the distance between input and reference template. The proposed algorithm can be reduced to Hidden Markov Models as a special case. For example, the distance function (Ψ) and the membership degrees curves (M_N) approximately correspond the transition and the emission probabilities in HMM , respectively.

In the domain of gesture recognition for training purposes, the algorithm eliminates some issues of HMM such as training, decoding, evaluation [15]. Compared to speech, which is one of the main application area of HMM, a gesture trajectory is not as complex as speech. Therefore, unlike HMM, modelling of gesture data does not require *hidden* states which aims to represent unknown infrastructure. Gesture data or trajectories, roughly speaking, are well observable, unlike speech. Moreover, the proposed algorithm does not consist of training and modelling issues of HMM such as optimal number of states, topology, transition and emission probabilities . For example, in HMM, EM or Bauch-Welsher algorithm are used to estimate optimal transition probabilities, in a way to maximise the transition expectations. In this sense, the distance function directly employs the expectation which is that transition from a frame or state to the neighbourhood frames that are more probable than to the remote frames. Moreover, evaluation and decoding are run straight away in the proposed algorithm. Transparent decoding provides valuable feedback for training purposes and synthesis.

In addition to that, in on-line recognition, the proposed algorithm provides more control parameters (e.g. path assessors) to prevent premature or incorrect recognition, unlike HMM. Maximum likelihood criteria and some threshold mechanism are the only available methods when using HMM. It is worth noting that controlled recognition is critical for training and feedback. For example, in *Yang* and W_TTest2 experiments , it is observed that, while HMMs misrecognise some deformed and uncompleted gestures, the proposed algorithm rejects to any recognition , which is vital for a reliable training.

The proposed algorithm conceptually is a template matching technique in which time warping is employed in an on-line mode. In this sense, it is similar to dynamic time warping (DTW) apart from off-line mode. Recall that, DTWs make comparison between a reference and input template. But in the proposed algorithm, only an input frame X is compared with reference templates C_i. Moreover, in the proposed algorithm, since the distance operations are carried out over the membership degree curves (membership probabilities), the issue of common distance unit in DTW is eliminated.

The components of recognition algorithm have scope of further improvement. The task of some components can be carried out by other conventional algorithm. For example, the function of frame predictor component could be replaced by a function approximation algorithm such as RBN neural networks [1].

V. Experiments

In order to assess the performance of the proposed algorithm, four data sets are considered in this paper. The first data set is an artificial data set (W_TTest) which enables to perform parametric analysis. The remaining data sets come from real world applications involving user interactions in virtual environment (VE). The interaction gesture data set involves trajectories of hand motion while drawing shapes in a virtual environment. The final two data sets are related to FDO gestures which are gathered in two different ways , computer vision (FDO_CV) and tracker based (FDO_PT). Prior to conducting experiments, a PCA based similarity measurement (EROS) is applied to estimate intra disparity characterization of the data sets [22]. In this paper, we compare the performance of the proposed algorithm with HMM and DTW in an off-line fashion. But main emphasis is given to HMM in the experiments. Experiments are conducted in agreement with previously published studies [9], [6]. 10-fold cross validation scheme is used for training and testing.

DTW is implemented with 0.2 Sakoe-Chiba band windowing [3]. Class models of the recognition machine (C) are used as the reference templates in DTW and input templates are stretched or compressed to have identical length with the reference templates. HMM algorithm is applied using HTK toolkit [18]. Several configuration of states and topologies such as left to right (*lr*), left to right one skip(*lr1s*) and ergodic (*er*) are considered. EROS employs weighted Frobenious norms to the eigenvector and eigenvalues of principal components which are obtained from covariance matrices of temporal classes represented as matrices. In the evaluation part, EROS uses non

Fig. 2 Static and dynamic FDO gestures

parametric kNN neighbourhood scheme ($k = 1, 2, 3 \cdots 10$) to evaluate the disparity in data sets. Precision/Recall metrics in EROS accommodates proportion of k to the volume (recall) which consists of k number of samples of class of interest. High values of precision (% 100) indicates higher disparity in data set. Further information about EROS can be found in [22].

A. Synthetic data set_W TTest [9]

W_TTest is a parametic data set and consist of three classes A,B and C and each of which has three channels (α, β, γ). Period of classes are 100 time units. It must be noted that apart from a couple of frames, class A and B are identical to each other. In noisy circumstances, these distinctive frames could also disappear.

W_TTest addresses the following challenges: multiple channels, spatial variance, temporal variances in the form of periodic and sub event, gaussian noise and irrelevant channels. These challenges are controlled with following parameters : d, duration or periodic variance; c, variance in sub events' positions; h, variance in sub events' amplitude; g, noise level and *irrel*, irrelevant channels $A_\gamma, B_\gamma, C_\beta$ which seems to convey a message but in fact it is random and unrelated to the class. Apart from *irrel* , other parameters range in the interval of [0,1] where 0 indicates that the parameter of interest is *off*. *Irrel* parameter is either *on* or *off*. The noise is distributed uniformly and randomly in the data set. For a detailed definition of the dataset ,the interested reader is referred to [9]. Experiments are conducted over different values of noise levels $g = 0.1$ and $g = 0.2$ which are referred as W_TTest1 and W_TTest2 in the rest of paper. Other parameters are fixed as follows $h = d = 0.2, c = 0.1$, *irrel on*. Actually, W_TTest2 , due to high noise, accommodate unclassifiable samples (6-8 %) to check reliability of the algorithms. Both data set (W_TTest1, W_TTest2) consist of 1000 samples for each classes. Raw data is used as features in both experiments.

B. Gestures for Interaction in VE - Yang [6], [7]

Yang data set is a part of full body gesture data set comprising over 40 body motions [6], [7]. The gesture set consists of eight hand gestures of which is represented by three coordinates (x,y,z) at a given time. For each gesture, there are approximately 100 training samples. The quality of data set is very poor. Because of the shape of *circle*, *rectangle* and *triangle* gestures, there is a remarkable similarity in the data set. Each gesture is represented by following features: smoothed coordinate positions, their gradients, and angular velocity. Previous work on Yang data set achieves about 2.1 % recognition error [6].

C. Flight Deck Officer - Vision Based (FDO CV)

Computer Vision based gestures are collected via an average quality desktop web cam. Collected videos are pre-processed to extract the position of hands (x, y). Three different users performed the gestures. 18 out of over 40 FDO gestures are considered in the present paper (Figure 2, middle). These gestures accommodate all challenges which one would come cross during FDO's gesture recognition. Data set consist of four static gestures (Affirmative, Clean, Hold On, Negative), six dynamic gestures (Ahead, Back, Wave Off, Down ...) and eight hybrid gesture (Left, Right, Fire ...) , in which while one hand is static, the other hand is dynamic. The data set includes over 70 samples of each gesture. Each raw gesture is represented by stream of four coordinate data (x, y) for each hand. The coordinate data x, y and their gradients are used as feature vector.

D. Flight Deck Officer - Tracker Based (FDO PT)

Characteristics of this data set are similar to FDO_CV apart from a couple differences in the way data is collected, size of data set, and number of person performing the gestures. FDO_PT is collected via a tracker device (Polhemus FasTrak) of which two sensors acquire the position of hands in a three dimensional coordinate system (x, y, z). FDO_PT consists of 150 samples foreach class and these samples are collected only from a single person in different sessions. Similar to FDO_CV, each raw gesture is represented by stream of six coordinate data (x, y, z). But, for feature , coordinates of right and left hand (x, y, z) are transformed to angular features (α, β, γ) which correspond coordinate angles between axes and hand position in the local coordinate system. Hence, α, β, γ and their gradients are used as the feature vector.

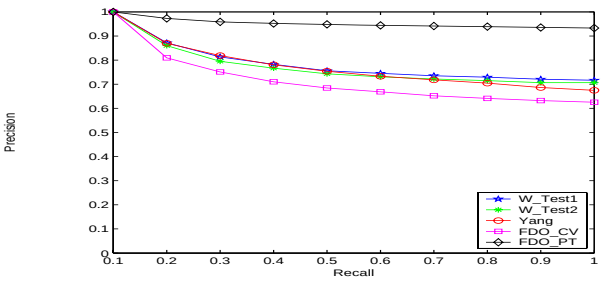


Fig. 3. Average Recall/Precision of data sets by using EROS ($k = 1, 2, 3, \cdots 10$)).

VI. Results

Prior to discussing recognition results, intra disparity of the datasets are analysed by EROS. For the sake of clarity and

TABLE I

HMM RECOGNITION ERROR RATES

	3lr	5lr	10lr	20lr	3lrs1	5lrs1	10lrs1	20lrs1	3er	5er	10er	20er
W_TTest1	0.6∓2.2	0.1∓0.5	0∓0	1.1∓3.6	1.4∓1.8	0.1∓0.5	2.3∓5.4	6.2∓11.9	18.2∓20.7	16.83∓25.7	17.1∓27	22.2∓38.5
W_TTest2	4.3∓3.7	3∓2.6	8.3∓7.6	14.1∓12.7	4.4∓3.7	2.9∓2.4	18.9∓25.2	19.8∓23.2	23∓23.9	24.8∓27	20.7∓19	16.3∓2.6
Yang	0∓0	0∓0	0∓0	0∓0	0∓0	0.3∓0.6	0.3∓0.6	0.2∓0.5	1.7∓1.4	1.3∓1.6	1.0∓1.6	1.6∓1.7
FDO_CV	1.2∓4.4	0.4∓2.0	0.4∓2.1	0.3∓1.8	1.2∓4.8	0.8∓3.8	0.4∓2.4	0.9∓4.1	1.6∓6.1	1.1∓3.7	1.9∓2.7	2.5∓4.2
FDO_PK	0.1∓0.3	0.1∓0.3	0∓0	0.2∓0.4	0∓0	0.1∓0.4	0.2∓0.5	0.2∓0.4	0∓0	0∓0	0.1∓0.3	0.2∓0.5

W_TTest1 ($g = 0.1$) W_TTest2 ($g = 0.2$), Yang and FDO recognition results when using HMM with different states (3,5,10, 20) and topologies , left to right (lr), left to right skip 1 (lrs1) and ergodic (er). 10 fold cross validation scheme is applied to all data sets.

TABLE II

RECOGNITION ERROR RATES (%)

	RM	HMM	DTW	EROS
W_TTest1	0.93∓0.43	0∓0	4.73∓4.05	21.32∓8.90
W_TTest2	7.83∓2.32	2.9∓2.47	14.33∓7.46	22.54∓9.25
Yang	0.86∓1.00	0∓0	27.08∓27.07	22.60∓9.99
FDO_CV	1.91∓1.47	0.3∓1.8	5.63∓14.71	28.25∓11.50
FDO_PT	0.09∓0.14	0∓0	0.03∓0.01	4.76∓2.05

Recognition Error Results in Percentages (%) for online RM and HMM , DTW and EROS. For HMM, best results of the table I is shown.

space, demonstration of intra class disparity of the datasets by EROS are omitted here. But the following observations are worth noting: There is remarkable similarity between class A and B in W_TTest1 and between *Rectangle* and *Triangle* gestures in Yang data set. Similar disparity results are also obtained for W_TTest2 data set. In FDOs, *Negative* and *Affirmative* gestures are similar due to common spatial and temporal properties. They are both static gestures and, except γ channel (z coordinate), other channels are same. Therefore, unique eigenvectors and eigenvalues are not formed for *Negative* and *Affirmative* gestures in EROS. Other gestures in FDOs are quite dissimilar. The figure 3 illustrates average cross disparity in all data sets in which the disparity in FDO_PT data set is higher compared to other data sets. It is worth noting that FDO_PT data set has higher disparity than FDO_CV. This difference is largely due to reduced number of channels in FDO_CV, combined with larger variation in number of users while collecting FDO_CV data set.

Table I shows HMM recognition error for all data sets. HMM experiments indicate that they perform better when state number is smaller (3,5, 10) and topology is left to right. It can be concluded that, left to right topology is more appropriate for gesture recognition task. Similarly, although RM is ergodic topology, its frame prediction component is biased to make prediction from left to right direction. Even though, HMM obtains comparable results as RM, during decoding of state sequence in HMM, small number of states, does not provide meaningful feedback which is critical for the training purposes.

Another important point of this study is that HMMs make strong assumptions during recognition decision. HMMs declare recognitions even in the cases where recognitions are impossible or unreliable due to high noise and missing data (table II). For example , in W_Test2 data set, because of high noise (g=0.2),in some cases (6-8 %), sub events emerge in the β channel of class B similar to class A, which make it impossible to segregate class A and B. Similarly, in Yang data set, becasue of high noise and missing data, some gestures barely can be classified by even a human. Even those , as the result tables indicate HMMs make over estimation and assign them to a class without considering the quality of the signal. In these circumstances, unlike HMMs, RM rejects to declare a defined class recognition and declares a non defined class recognition ($R_C = C_{NON}$). This advantage of RM is achieved by some heuristics along side with maximum likelihood criteria employed in the *path assessor* components.

Finally, the table II compares the recognition error (%) of the proposed algorithm (RM) with other algorithms. Best results of HMMs experiments from table I are shown in the table II. EROS column shows the average precision of cross disparity for all neighbourhoods ($k = 1, 2, 3 \cdots 10$) for the data set of interest.

The proposed algorithm (RM) achieves remarkable results compared to HMMs and other algorithm, considering that RM is an on-line algorithm and others are off-line, the performance are comparable. It is observed that in W_TTest2 data set, performance of RM is decreased because of high noise ($g = 0.2$), which deforms and diminishes the sub events of the classes.

VII. CONCLUSION

In this paper, we proposed an on-line recognition machine for gesture of FDO in the context of a training application. Recognition machine is based on the generic pattern recognition framework. Gestures are represented in a template form. Recognition algorithm is based on dynamic programming and markovian process and it conceptually implements an on-line template matching scheme. The algorithm predicts the index of an input frame in each class templates. Consistency in the sequence of prediction scores provides a merit for recognition. In addition, the prediction process paves way for automatic detection of start/end frames of gestures in a continuous stream by exploiting path heuristics.

The proposed algorithm (RM) is compared with HMM and DTW algorithm using a parametric artificial data set (W_Test) and three real word data sets (Yang, vision and tracker based FDO). Even though, RM is an on-line algorithm and uses limited historical data, it achieves comparable results on segmented data. Controlled declaration of recognition in the cases

of high noise and missing data provides an advantage over HMM. Moreover, RM provides more meaningful feedbacks (longer observed trajectory) by employing more observable states , unlike HMM which is generally successful on small configuration (3-5 states) and on hidden states. It is worth emphasizing that DTW are primarily designed for off-line recognition , while the RM algorithm has been designed to deal with continuous gestures. Preliminary results of continuous gesture recognition of RM are promising and RM achieves far better results compared to HMM, which will be presented in a future paper.

It is proposed to extend the present study for recognition of continuous gestures which forms part of gesture dialogues in the FDO training application.

REFERENCES

[1] Christopher M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1996.

[2] Herve Bourlardy and Samy Bengio. *The Handbook of Brain Theory and Neural Networks*, chapter Hidden Markov Models and other Finite State Automata for Sequence Processing. The MIT Press, second edition, 2002.

[3] E. Keogh C. A. Ratanamahatana. Everything you know about dynamic time warping is wrong. *Third Workshop on Mining Temporal and Sequential Data, in conjunction with the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.

[4] Andrea Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS'01)*, page 82. IEEE Computer Society, 2001.

[5] S. Sidney Fels and Geoffrey E. Hinton. Glove-Talk: A neural network interface between a data-glove and a speech synthesizer. *IEEE Transactions on Neural Networks*, 4(1):2–8, January 1993.

[6] Yang-Hee Nam Jane Koh. Full-body motion recognition using principal component based target reduction. In *KIPS(Korean Information Processing Society) Proceedings*, volume Vol. 11 , no.1, pages 873–876, Korea, May 2004.

[7] Yang-Hee Nam Jane Koh, Eun-Woo Lee. Full-body motion recognition using multi-phase target reduction method. *HCI 2004(Korean)*, 2004.

[8] Yangsheng Xu Jie Yang. Hidden markov model for gesture recognition. Technical Report CMU-RI-TR-94-10, The Robotics Institute, Carneige Melon University, 1994.

[9] M. W. Kadous. *Temporal Classification: Extending the Classification Paradigm to Multivariate Time Series.* PhD thesis, The University of New South Wales, School of Computer Science and Engineering, 2002.

[10] C. Lee and Y. Xu. Online, interactive learning of gestures for human/robot interfaces, 1996.

[11] H. Li and M. Greenspan. Continuous time-varying gesture segmentation by dynamic time warping of compound gesture models. 2005.

[12] Kouichi Murakami and Hitomi Taguchi. Gesture recognition using recurrent neural networks. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 237–242, New York, NY, USA, 1991. ACM Press.

[13] Y. Nam and K. Wohn. Recognition of space-time handgestures using hidden markov model, 1996.

[14] Vladimir Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695, 1997.

[15] Rabiner L. R. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. IEEE*, 77, (2):257–286, Feb 1989.

[16] Gerhard Rigoll, Andreas Kosmala, and Stefan Eickeler. High performance real-time gesture recognition using hidden markov models. In *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, pages 69–80, London, UK, 1998. Springer-Verlag.

[17] D T Sodiri and V V S S Sastry. On the interpretation of gestures arising in flight deck officers training. In *Proceedings of the Thirteenth Conference on Behaviour Representation in Modelling and Simulation*, 2004.

[18] Thomas Hain-Phil Woodland Steve Young, Gunnar Evermann. *The HTK Book, 3.2.1*. Cambridge Research Laboratory Ltd, 2002.

[19] M Turk. *Handbook of virtual environments: Design, implementation, and applications*, chapter Gesture recognition, pages 223–238. Mahwah, NJ: Lawrence Erlbaum Associates, Inc., 2002.

[20] P. Vamplew and A. Adams. Recognition and anticipation of hand motions using a recurrent neural network, 1995.

[21] Simei G. Wysoski, Marcus V. Lamar, Susumu Kuroyanagi, and Akira Iwata. A rotation invariant approach on static-gesture recognition using boundary histograms and neural networks.

[22] Kiyoung Yang and Cyrus Shahabi. A pca-based similarity measure for multivariate time series. In *Proceedings of the 2nd ACM international workshop on Multimedia databases*, pages 65–74. ACM Press, 2004.

Appendix B

FDO Gestures

This appendix is a chapter from a military training publication on FDO gestures, in order to illustrates the complete list of FDO gestures [149].

MARSHALLING SIGNALS

1. Introduction

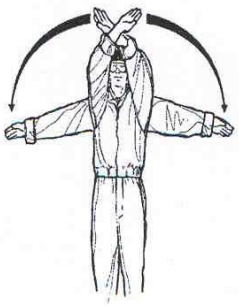
This chapter shows the hand signals available to the FDO/Director/Marshaller and Aircrew when engaged in routine operations on the flight deck. These signals are used on both single and multi spot ships and are applicable to fixed and rotary wing aircraft. Hand Signals that are common between Nations for use on flight decks are contained in APP 2 Vols 1 and 2.

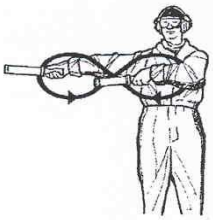
2. List of Signals

No	Description		No	Description
1	Hold On deck		21	FDO Engage Rotors
2	Wave Off		22	Pilot Ready for Shut Down
3	Fixed Wing Cancel Launch		23	FDO Shut Down
4	Fixed Wing Launch		24	FDO Droop Stops
5	Fire		25	Pilot Fold Main Rotor Blades
6	Abandon Aircraft		26	FDO Fold Main Rotor Blades
7	Pilot Affirmative / All Clear		27	Pilot Fold Tail Pylon Only
8	Pilot Negative NOT Clear		28	FDO Fold Tail Pylon Only
9	FDO Affirmative		29	Pilot Disconnect Ground Supply
10	FDO Negative		30	FDO Ground Supply to be Disconnected
11	Pilot Request start APU		31	FDO Connect Ground Power
12	FDO Start/Confirm APU Running		32	FDO Fuel System Pressure On
13	Pilot Request Start Engines		33	FDO Fuel System Pressure Off
14	FDO Start Engines		34	Ground Crew Fuel Spillage
15	Director/Marshaller U/C Doors		35	FDO Fuel Spillage
16	Pilot Spread Main Rotor Blades		36	FDO/Pilot Connect Telebrief
17	FDO Spread Main Rotor Blades		37	FDO Personnel Wish to Approach Aircraft
18	Pilot Spread Tail Pylon Only		38	FDO Personnel Approach the Aircraft
19	FDO Spread Tail Pylon Only		39	▸ Free Disk (weapon loading team)
20	Pilot Request to Engage Rotors		40	Pilot Wishes to Speak to Ground Crew ◂

41	Ground Crew wish to Speak to Pilot		68	FDO Down Locks Install
42	FDO Away Lashings		69	FDO Down Locks Remove
43	Pilot Ready For Lashings		70	FDO Tail Wheel/Nose Gear steering Unlock Disengage
44	FDO In Chocks & Lashings		71	FDO Tail Wheel Lock
45	FDO Remove Chocks		72	FDO Flaps Lower
46	FDO Insert Chocks		73	FDO Flaps Raise
47	Pilot Request to Swivel		74	FDO Engage Nose wheel Steering
48	FDO Disengage Harpoon ▸ Deck Lock ◂		75	FDO Commence Acceleration Checks
49	FDO Engage Harpoon ▸ Deck Lock ◂		76	FDO NO GO Vertical Take-Off
50	FDO Indicate Direction Of Departure		77	FDO/Director Wet No-Go VTO All Clear
51	FDO Come Up		78	FDO Nozzle Checks
52	FDO Move Down		79	FDO Up Winch/Cargo hook
53	FDO Hover		80	FDO Down Winch/Cargo Hook
54	FDO Move Left (Airborne Only)		81	FDO Load Has Not Been Released
55	FDO Move Right(Airborne Only)		82	FDO Release Load
56	FDO Clear		83	FDO Hook Up Load
57	FDO Indicate Landing Direction		84	FDO Cut Cable
58	FDO This Way		85	FDO Lower Wheels
59	FDO Proceed to Next Marshaller		86	FDO Raise AEW Radar Dome
60	FDO I have charge of your Aircraft		87	Merlin COMPWASH Parameters are correct
61	FDO SPOT Turn to Port		88	Weapons Safe
62	FDO Spot Turn to Starboard		89	F/W No Attempt to Fire Weapons
63	FDO Move Back		90	F/W Attempted to Fire Weapons
64	FDO Come Ahead		91	Type of Weapon GUN
65	FDO Slow Down		92	Type of Weapon MISSILE
66	FDO Stop		93	Type of Weapon BOMB
67	FDO Brakes		94	F/W I Require to Land

B. FDO GESTURES

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
1	FDO /Marshaller	Hold on deck (Mandatory signal)		
2	FDO /Marshaller	Wave-off (Mandatory signal)		
3	FDO	FIXED WING CANCEL LAUNCH Mandatory Signal "Raised Crossed Flags" The Red Flag is raised and crossed with the Green flag above the FDO's head.		Same as day signal with Red and Green wands
4	FDO	FIXED WING LAUNCH		Same as day signal with Green wand

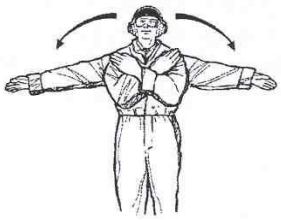
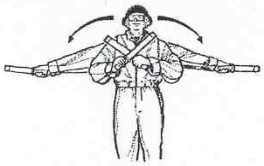
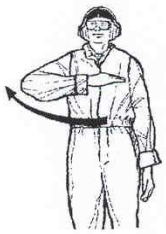
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
5	FDO /Marshaller	FIRE (Make rapid horizontal figure-of-eight motion at waist level with either arm, the other arm pointing at the source of the fire).		
6	FDO /Marshaller	ABANDON AIRCRAFT Simulate unfastening seat belt and shoulder straps and throwing them up and off		Same as day with wands held as extension of arm.
7	Pilot	Affirmative / All Clear.		As Day Signal with cockpit lighting
8	Pilot	Negative / NOT Clear		Same as Day with cockpit lighting

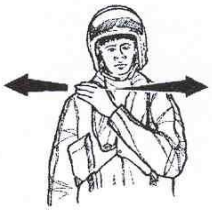
B. FDO GESTURES

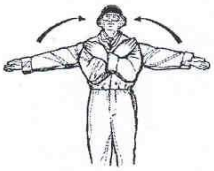
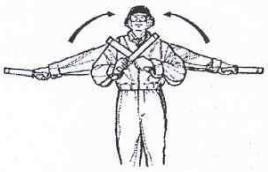
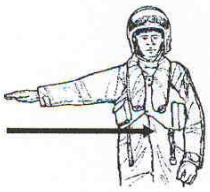
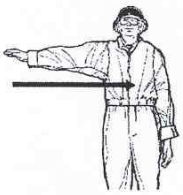
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
9	FDO /Marshaller	Affirmative		
10	FDO /Marshaller	Negative (not clear)		
11	Pilot (May be initiated by Ground Crew)	Request start APU		Cockpit Light On with same visual signal as day. Nav Lights to steady bright on starting APU
12	FDO /Marshaller	Clear to start APU. or Confirm APU Running *Note. extinguisher required*		Same as day using wands

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
13	Pilot	Request start engines. (Indicate No. of engine by raising No. of fingers) ***Merlin Notes***. *1. Merlin can start engines(x3) in any order* *2. Signal repeated using appropriate number of fingers until all engines have been started.*		Navigation lights - steady bright **MERLIN** Nav Lights Dim/Bright- 1 x for Number 1 engine 2 x for Number 2 engine 3 x for Number 3 engine
14	FDO /Marshaller	Start engines (indicate no.of engine by raising the no. of fingers, or by night, by pointing to engine with illuminated wand in right hand) *Note. MERLIN Same as day with addition of wands (indicate engine No by FLASHING right wand appropriate number of times.*		
15	Director / Marshaller	Harrier - Confirmation that nose undercarriage doors have closed post engine start. **Note.** If doors rail to close post start, it is mandatory for the director/marshaller to instruct the pilot to shut down.		As per day with wands used as extension of arms.
16	Pilot	Request to spread Main Rotor Blades *Note. MERLIN During normal operations will spread main rotor blades and tail pylon simultaneously in response to this request.*		Same as day with cockpit light on

B. FDO GESTURES

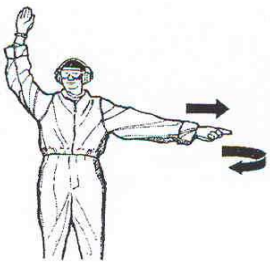
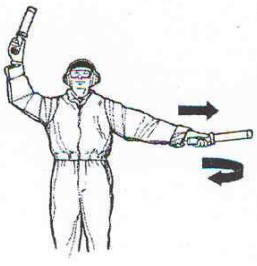
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
17	FDO /Marshaller	Spread Main Rotor Blades ***Note.*** *MERLIN During normal operations will spread main rotor blades and tail pylon simultaneously in response to this signal.*		
18	Pilot	Request to spread Tail Pylon ONLY		Same as day with cockpit light on
19	FDO /Marshaller	Clear to Spread Tail Pylon		Same as day signal with the addition of wands
20	Pilot	Request engage rotors.		Navigation lights to flashing dim.
21	FDO /Marshaller	Engage Rotors *Note.* *Check limits before signalling*		

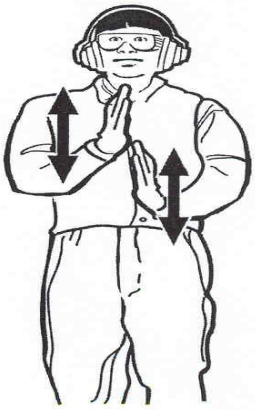
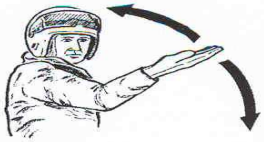
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
22	Pilot	Ready for shut-down. Merlin Ready to shut down Rotors and all 3 engines. ***Note.*** *Merlin will normally shut all 3 engines down simultaneously*		As for day, but with a torch
23	FDO /Marshaller	Shut-down. *Note.* ***Merlin*** *Droop stops are not visible.*		
24	FDO /Marshaller	Droop stops: Out. When helo starts to run-down, both thumbs raised above the head, pointing outwards. IN. When droop stops go in, both thumbs turned inwards.	 	 

No.	By Whom	Signal Meaning	Signal - Day	Signal - Night
25	Pilot	Request Fold Rotors		Same as Day with cockpit lighting
26	FDO /Marshaller	Clear to fold rotors. ***Note**. MERLIN During normal operations will fold main rotor blades and tail pylon simultaneously in response to this signal.*		
27	Pilot	Request to fold tail Pylon Only		Same as Day with cockpit light on
28	FDO /Marshaller	Clear to fold Tail Pylon		Same as day with Addition of wands
29	Pilot	Disconnect ground power supply.		Cockpit light on - signal as for day.

No.	By Whom	Signal Meaning	Signal - Day	Signal - Night
30	FDO /Marshaller	Ground Power Supply to be disconnected		
31	FDO /Marshaller	Connect Ground Power Supply.		
32	Ground Crew Member	Fuel System - Pressure On		
33	Ground Crew Member	Fuel System - Pressure Off		

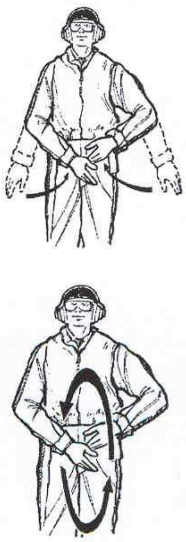
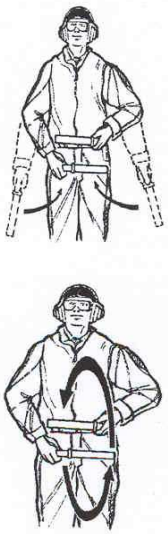
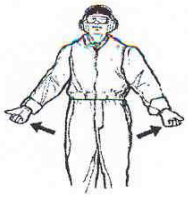
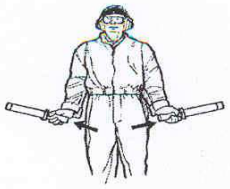
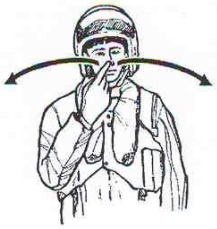
B. FDO GESTURES

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
34	Ground Crew Member	Fuel Spillage		
			Point to spillage, making a horizontal, circling motion with the hand/wand.	
35	FDO	Fuel Spillage	Repeat signal to acknowledge the spillage. Inform OOW/Command on Helo intercom.	
36	FDO /Marshaller /Pilot	Connect Telebrief. (Make T-signal at head-level).		
37	FDO /Marshaller	Personnel wish to approach the aircraft		
38	FDO/ Marshaller	Personnel Approach the Aircraft.		

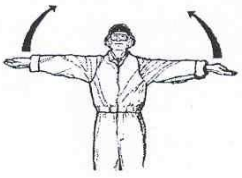
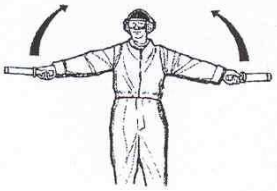
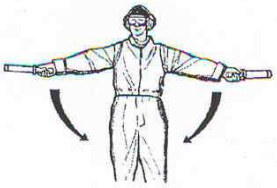
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
39	FDO/SMR/ Weapons Supervisor	Free Disc (Weapon Loading Teams ONLY) SMR/Weapons Supervisor to indicate to the aircrew 'Weapons Safe'. When weapons safe confirmed by aircrew SMR / Weapons Supervisor gives this signal to FDO who repeats the signal to Weapons Team and Aircrew. Free Disc ends on completion of weapon load/unload when the MF705D is signed.		
40	Pilot	Pilot wishes to speak to ground crew.	 Telebrief/radio call.	Two flashes of torch **outside** the window.
41	FDO /Marshaller	Ground crew wish to speak to pilot.	As above. Acknowledged, then 'Personnel approach aircraft' signal.	As above. Acknowledged, then 'Personnel approach aircraft' signal.

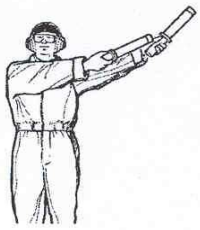
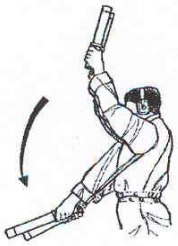
B. FDO GESTURES

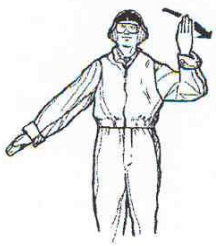
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
42	FDO /Marshaller	Ship on flying course - away lashings signal, passing relative wind if required. Lashing no's to muster in front of a/c under the disc. When all lashing no's correct, away lashings signal is given again. Lashing no's muster by FDO with the lashings held aloft.		 Lashings to be individually pointed out using wands when mustered by the FDO.
43	Pilot	Ready for lashings.		Navigation lights flashing dim.

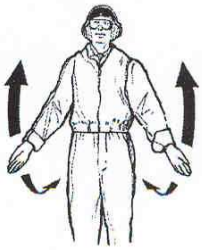
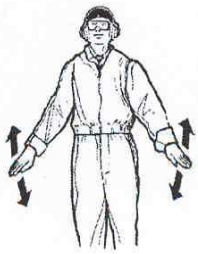
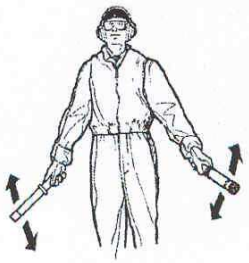
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
44	FDO /Marshaller	In chocks and lashings for shut-down. Lashings to be secured then lashing no's to muster in front of the aircraft, under the disc.		
45	FDO /Marshaller	CHOCKS REMOVE (Conforms to ICAO Signal).		
46	FDO /Marshaller	CHOCKS INSERT (Conforms to ICAO Signal).		
47	Pilot	Request to Swivel into the Relative Wind (Pilot Points to nose, then indicates direction)		N/A

B. FDO GESTURES

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
48	FDO /Marshaller	Disengage Harpoon ‣ Deck Lock ◂ . *Note. Merlin Downward Ident light on for visual check*		
49	FDO /Marshaller	Engage Harpoon ‣ Deck Lock ◂ .		
50	FDO /Marshaller	Indicates direction of departure		As day signal with the addition of wands
51	FDO /Marshaller	Come up		
52	FDO /Marshaller	Move down		

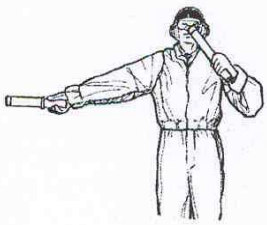
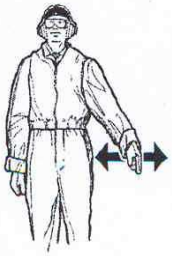
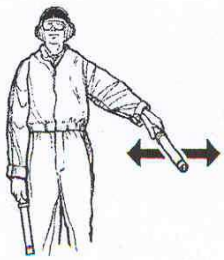
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
53	FDO / Marshaller	Hover		
54	FDO / Marshaller	Move LEFT (Airborne Aircraft Only)		
55	FDO / Marshaller	Move Right (Airborne Aircraft Only)		
56	FDO / Marshaller	Clear		
57	FDO / Marshaller	With back to relative wind, indicates the landing direction during silent operations. (Arms to move from vertical to horizontal in front of the body, three times).		

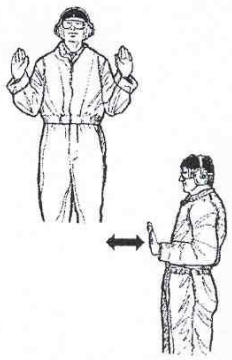
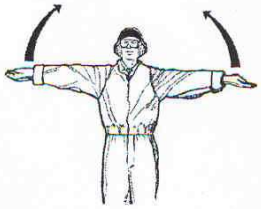
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
58	FDO /Marshaller	THIS WAY Ready to receive - you are clear to land.		
59	FDO /Marshaller	Proceed to NEXT Marshaller.		
60	FDO /Marshaller	I HAVE CHARGE OF YOUR AIRCRAFT		RED WAND 
61	FDO /Marshaller	SPOT Turn to Port (Applicable to both airborne and ground taxying aircraft).		

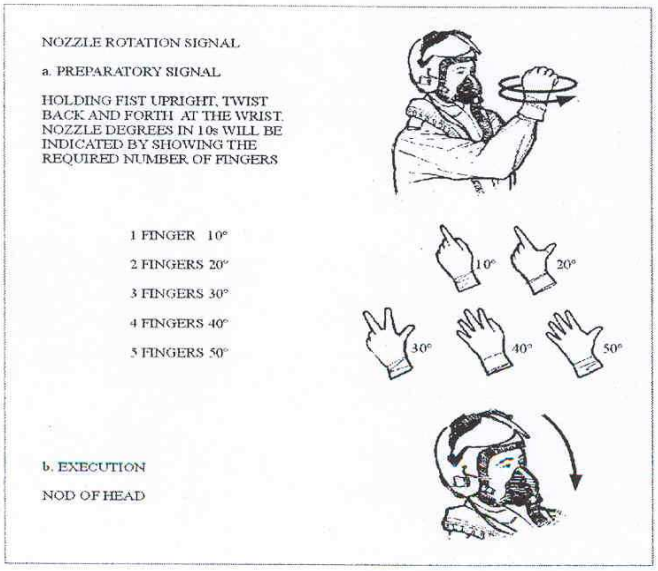
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
62	FDO / Marshaller	SPOT Turn to Starboard (Applicable to both airborne and ground taxying aircraft).		
63	FDO / Marshaller	Move back.		
64	FDO / Marshaller	Come Ahead		
65	FDO / Marshaller	SLOW DOWN (Conforms to ICAO signal)		

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
66	FDO /Marshaller	Stop.		
67	FDO /Marshaller	BRAKES. (Conforms to ICAO Signal)	ON DAY Arms above head, palms and fingers raised with palms toward aircraft, then fist closed.  OFF DAY Reverse of Above	ON NIGHT Arms above head then wands crossed.  OFF NIGHT Reverse of Above
68	FDO /Marshaller	DOWN LOCKS /UNDERCARRIAGE PINS/ARMAMENT PINS - INSTALL		Night similar except the right wand is placed against left forearm. The wand in the left hand is held vertical.

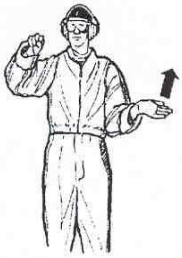
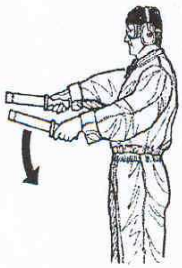
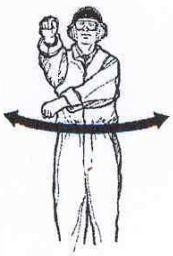
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
69	FDO /Marshaller	DOWN LOCKS /UNDERCARRIAGE PINS/ARMAMENT PINS - REMOVE.	With arms and hands in the "INSTALL DOWN LOCKS" position the right hand unclasps the left forearm. 	Similar to day signal except with the addition of wands.
70	FDO /Marshaller	Tail Wheel Unlock/ Disengage Nosegear steering		
71	FDO/ Marshaller	Lock tail wheel.		

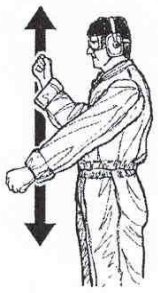
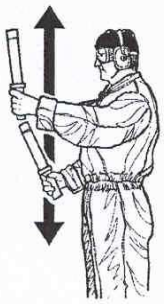
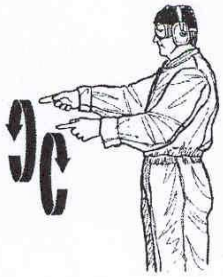
No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
72	FDO /Marshaller	FLAPS - Lower wing flaps.		
73	FDO /Marshaller	FLAPS - Raise Wing Flaps.		
74	FDO /Marshaller	ENGAGE NOSE WHEEL STEERING		
75	FDO /Marshaller	COMMENCE ACCELERATION CHECKS.		

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
76	FDO / Director	NO GO VERTICAL TAKE OFF		N/A
77	FDO / Director	Wet No-Go VTO All Clear		N/A
78	Pilot	NOZZLE Checks Harrier FA2/GR7		



B. FDO GESTURES

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
79	FDO /Marshaller	Up winch/cargo hook.		
80	FDO /Marshaller	Down winch/cargo hook.		
81	FDO /Marshaller	Load has not been released. (Make T-shape with the hands/wands).		
82	FDO /Marshaller	Release load.		

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
83	FDO /Marshaller	Hook Up Load		
84	FDO /Marshaller	Cut cable.		Same as day signal with the addition of wands, keeping the left hand wand pointing towards the aircraft.
85	FDO /Marshaller	Lower wheels. (Given when aircraft approaches with landing gear retracted. Marshaller gives signal by FRONT view of a cranking, circular motion of the hands).		Night same as day with wands held as extension of arms. **Note.** When EMCON policy allows, the FDO should accompany the signal with the radio call 'wheels, wheels, wheels'.
86	FDO /Marshaller	Raise AEW Radar Dome		As day signal with the addition of wands

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
87	Merlin Pilot/ Ground Crew	MERLIN COMPWASH Correct parameters have been achieved for the fluid to be "shot" into the engine. *Note.* *This signal is similar to and should NOT BE CONFUSED with that used by ARMED Sea Harrier / GR7's on landing.*		Same as Day with Cockpit Light On
88	Pilot	Weapons Switches are to Safe		Same as Day
89	F/W Pilot	On Landing NO ATTEMPT HAS BEEN MADE TO FIRE or RELEASE WEAPONS		Same as Day
90	F/W Pilot	On Landing FIRED or AN ATTEMPT HAS BEEN MADE TO FIRE or RELEASE WEAPONS *I*		Same as Day

No.	By Whom	Signal Meaning	Signal - Day	Signal-Night
91	F/W Pilot	Weapon Used GUN		Same as Day
92	F/W Pilot	Weapon Used MISSILE		Same as Day
93	F/W Pilot	Weapon Used BOMB		Same as Day
94	F/W Pilot	Require to Land	1. DESIRE TO LAND CONVENTIONALLY AS SOON AS POSSIBLE. MOVEMENT OF THE HAND, FLAT, PALM DOWNWARD, FROM ABOVE THE HEAD FORWARD AND DOWNWARD, FINISHING THE MOVEMENT IN A SIMULATED ROUND-OUT.  2. DESIRE TO LAND VERTICALLY AS SOON AS POSSIBLE. SAME AS ABOVE EXCEPT FINISHING THE MOVEMENT IN A VERTICAL MOTION. 	

Appendix C

Polhemus Fastrak Device

C.1 Components

The 3SPACE FASTRAK system includes a System Electronics Unit (SEU), a power supply, one receiver and one transmitter. You can expand the system's capabilities simply by adding up to three additional receivers. FASTRAK is also available as a board-level product for OEM/VARs.

Figure C.1: Tracker based Polhemus FasTrack

System Electronics Unit: Contains the hardware and software necessary to generate and sense the magnetic fields, compute position and orientation, and interface with the host computer via an RS-232.

Transmitter: The transmitter is a triad of electromagnetic coils, enclosed in a plastic shell that emits the magnetic fields. The transmitter is the system's reference frame

for receiver measurements.

Receiver: The receiver is a small triad of electromagnetic coils, enclosed in a plastic shell that detects the magnetic fields emitted by the transmitter. The receiver is a lightweight cube whose position and orientation are precisely measured as it is moved. The receiver is completely passive and highly reliable.

Stylus (Optional): The stylus is a pencil-like device that contains a triad of electromagnetic coils, and is used for digitising objects, drawing in three-dimensional space, or collecting contours of objects. The stylus is available in 3" or 8" lengths, with either a sharp or round nib.

3BALL (Optional): A triad of electromagnetic coils are housed in a billiard ball for use as a mouse to literally rotate the captured database, act as a light source for shading, or become the eye for perspective views.

C.2 Features & Specifications

Real Time: Virtually no latency. Digital Signal Processing (DSP) technology provides 4ms latency updated at 120 Hz. And data is transmitted to the host at up to 100K bytes/sec.

Improved Accuracy and Resolution: Accuracy of 0.03" RMS with a resolution of 0.0002 in./in. makes this the most precise device of its kind.

Range: Standard range is up to 10 feet. Coverage of up to 30 feet is possible with the optional LONG RANGER transmitter.

Multiple Receiver Operation: Permits measurement of up to 4 receivers on a single system and up to 16 receivers at a time, utilizing four multiplexed systems.

Reliable: From the pioneer in 3D position/orientation measuring devices, in business since 1970. Factory calibrated, never needs adjustment.

Multiple Output Formats: Position in Cartesian coordinates (inches or centimeters); orientation in direction cosines, Euler angles or Quaternions.

Position and Orientation Coverage:

The system will provide the specified performance when the receivers are within 30 inches of the transmitter.

Coverage of up to 10 feet is possible with slightly reduced performance.

Latency:

4 milliseconds.

Update Rate:

120 updates/seconds divided by the number of receivers.

Interface:

RS-232 with selectable baud rates up to 115.2K baud (optional RS-422).

Static Accuracy :

0.03” RMS for the X, Y, or Z position;

0.15 degrees RMS for receiver = orientation.

Resolution:

0.0002 inches per inch of transmitter and receiver separation;

0.025 degrees orientation.

Coverage:

Up to 10 feet with standard transmitter.

Coverage of up to 30 feet is possible with the optional LONG RANGER transmitter.

Multiple Systems:

Multiple systems can be frequency multiplexed with no change in update rate.

CRT Interference Rejection:

Provided by means of an external cable sensor.

Angular Coverage:

The receivers are all-attitude.

Operating Environment:

Large metallic objects, such as desks or cabinets, located near the transmitter or receiver, may adversely affect the performance of the system.

Operating Temperature:

10 C to 40 C at a relative humidity of 10

Physical Characteristics :

SEU -11.0” L x 11.4” W x 3.6” H

Power Supply -7.0” L x 3.7Y W x 2.2Y H

Transmitter - 2.3” L x 2.2” W x 2.2” H

Receiver - 0.9” L x 1.1” W x 0.6” H

Power Requirements:

25 W, 90-250 VAC, 38-65 Hz

Regulations:

Meets FCC, CSA, UL, and CE Requirements.

C.3 Usage of Polhemus FasTrak

Polhemus Fastrak is a tracking device, which returns six data per sensor and can use up to four sensor. The tracking device Fastrak has a transmitter, which creates a magnetic

field, and each sensors, which detect the strength and orientation of magnetic field. By using the strength and orientation of field, position and orientation of sensor can be computed. The data from each sensor are coordinates (x, y, z) and orientation angles (Azimuth, Elevation, Roll Euler).

Fastrak can be connected with computer either a RS232 serial port or an IEEE-488 parallel port. Through, commands can be sent to the device and data are received. The format of command depends on its function. Data can be received as a single record or continuous stream of records. Fastrak has a flexible communication configuration. On the Fastrak unit switches are used to set the basic communication parameters.(Baud Rate, parity, hardware handshaking, character width, RS-232/IEEE-488)

The structure of a Polhemus Fastrak record as follows. Each record starts with 3 bytes header followed by set of numerical value such as cartesian coordinate and orientation angles data, and then a carriage return/line feed. Numerical value can be in ASCII format or binary format according to device mode. In ASCII mode, numerical value is stored as ASCII string of which size is seven bytes one for sign 3 for the integer part, one for the dot and two for the fractional part. Size of a record is 47 in ASCII mode. If an error occurs, error code will be logged in third character. In other case value of third value is a space character. In Table C.1 structure of record is shown(A1:1 byte ASCII, S: sign,+ or -, and x: digit).

Byte Order	Identification	Format
0	Record Type	A1
1	Sensor Number	A1
2	System Error Code	A1
3	x Cartesian Coordinates	(Sxxx.xx)
10	y Cartesian Coordinates	(Sxxx.xx)
17	z Cartesian Coordinates	(Sxxx.xx)
24	az Orientation angle	(Sxxx.xx)
31	el Orientation angle	(Sxxx.xx)
38	roll Euler Orientation angle	(Sxxx.xx)
45	Return Line Feed	A1

Table C.1: Record Structure of Polhemus Fastrak Device

As mentioned in Fastrak specification, device has 0.03” RMS for the X, Y, or Z position and 0.15 degrees RMS for orientation. Fastrak’s update rate is 120 per second divided by the number of sensor. For example if there is one sensor, update rate will be 120, if there are two sensor, update rate will be 60 for each sensor. In other word, the number of sample in a second is always 120. Data will be unreliable if sensor is far more than 10 feet from transmitter.

C.4 Polhemus Fastrak Driver

RS232 serial port protocol is used to communicate with Fastrak the tracking device. Therefore, it is necessary to explain a little about RS232 serial port communication.

RS232 is popular standard communication protocol. RS stand for Recommended Standard and XXX indicates its version. RS232 devices can be plugged straight into computer's serial ports (COM1, COM2).

The concept behind serial communications is as follows, data is transferred from sender to receiver one bit at a time through a single line or circuit. The serial port takes 8, 16 or 32 parallel bits from a computer bus and converts it into an 8, 16 or 32 bit serial stream. The name serial communications comes from this fact; each bit of information is transferred in series from one location to another.

Communication speed and structure of the data are important to communicate for computer and external device. Serial ports can communicate at various word size and baud rate. Structure of data consist of parity bits for error checking and flow control operations. And this parameter must be set before using the serial port.

RS232 supports several other modes for a efficient and reliable communication. Buffering modes (canonical or raw) and blocking modes (blocking or non-blocking modes) are some of these modes. Buffering modes determines what time data, which is held in buffer, should be returned. In canonical mode, data is returned when a full line is received. In raw mode, each character is send as soon as it is received. Blocking modes determines what should be done when the data read from a device and data is unavailable. If the driver is set blocking mode, then driver blocks until a data is available. In the non-blocking case, driver return with a error which indicates no data is available.

Information given about RS232 is enough to explain how to Fastrak driver is programmed. Because of the record structure of the Fastrak is ended by a return line feed, canonical mode is implemented. After each 47 bytes, buffer is flushed and data is parsed. Time delaying is not acceptable in real time virtual environment systems. Instead of waiting the data, the system should deal with other tasks. Therefore, non-blocking mode is selected as blocking modes. Although, Fastrak supports up to 115.2K baud, 9600 baud is selected because of its reliable communication. The communication is set up as 8N1 namely 8bit,no parity, 1 stop-bit.

All the things explained till are so low-level RS232 serial port initialising step. The next step is to build high-level functions, which run the Fastrak properly. The steps preparing the Fastrak to ready to acquire the data are follows:

```
/* Baud rate, parity, stop bit, buffer-blocking mode*/
```

```
prepareSerialPort();
/*Send Carriage Return Key to initiliaze Fastrak*/
sendCarriageReturnKey();
/*Resetting takes about 13 seconds.*/
ResetFastrak();
/*Wait until first point is received*/
RequestPoint();
/*Set unit as centimetre*/
SetUnitCentemeter();
/*Setting acquiring Continuous Data*/
setContinous();
```

After preparing the Fastrak, the final thing is to implement how to acquire data. As mentioned earlier, the structure of data is fixed.(Please look Table C.1). By using non-blocking and canonical feature of communication, the data are acquired, parsed and returned back in form of following structure.

```
struct fastrak_data_struct{
float x, y, z;
float a, e, r;
int sensor;
}
```

C.5 Error Code Of Polhemus FastTrak

Symptom	Possible Solution
Fastrak Won't Communicate	Check Dipswitch Settings Check RS-232 Cable Check Communication Program Settings Check PC COM Port
Green Light Won't Stop Flashing	Download New Firmware
BIT Error a-c	Change Tuning Module Move Transmitter Away From Metal Replace Power Supply Brick
BIT Error D-F, J-L	Turn Off CRT-Based Displays Separate Receivers
BIT Error d-g	Move Receivers Away From CRT-Based Displays Separate Receivers
BIT Error k	Reduce Range
BIT Error m, x, y	Perform Following Command Sequence: "W", "Ctrl K", "Ctrl Y" (Resets System Defaults)
BIT Error s	Reduce Range
BIT Error t, u	Test with Compensation Turned Off (Send "d" Command)

Table C.2: Fastrak Error Code and Possible Solutions

Bibliography

[1] L. E. Baum and J. A. Egon. An inequality with applications to statistical estimation for probabilistic functions of a markov process and to a model for ecology. *Bull. Amer. Meteorology Soc.*, 73:360–363, 1967.

[2] L.E. Baum and G.R. Sell. Growth functions for transformations on manifolds. *Pacific J. Math.*, 27,(2):211–227, 1968.

[3] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[4] A. Y. Benbasat and J. A. Paradiso. An inertial measurement framework for gesture recognition and applications. In *GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction*, pages 9–20. Springer-Verlag, 2002.

[5] Y. Bengio. Markovian models for sequential data. Technical Report 1049, Dept. IRO, Universit'e de Montr'eal., 1996.

[6] J Bilmes. A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical Report ICSI-TR-97-021, University of Berkeley, 1997.

[7] J Bilmes. What HMMs can do? Technical Report UWEETR-2002-0003, Dept of EE, University of Washington, 2002.

[8] C. M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1996.

[9] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, August 2006.

[10] G. Bouchard and B. Triggs. The tradeoff between generative and discriminative classifiers. In *IASC International Symposium on Computational Statistics (COMPSTAT)*, pages 721–728, Prague, August 2004.

[11] H. Bourlard and N. Morgan. Hybrid HMM/ANN systems for speech recognition: Overview and new research directions. In *Summer School on Neural Networks*, pages 389–417, 1997.

[12] H. Bourlardy and S. Bengio. *The Handbook of Brain Theory and Neural Networks*, chapter Hidden Markov Models and other Finite State Automata for Sequence Processing, pages 528–533. The MIT Press, second edition, 2002.

[13] C. Cadoz. *Les ralits virtuelles*. Dominos, Flammarion, 1994.

[14] Y. Chen, W. Gao, and J. Ma. Hand gesture recognition based on decision tree. In *International Symposium on Chinese Spoken Language Processing*, pages 299–302, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 2000.

[15] S. Chu, E. Keogh, D. Hart, and M. Pazzani. Iterative deepening dynamic time warping for time series. In *Proc 2 SIAM International Conference on Data Mining., http://www.siam.org/meetings/sdm02/sdm02-12.pdf*, 2002. [last checked 06-June-2007].

[16] C. Cohen. *A Brief Overview of Gesture Recognition*. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/COHEN/gesture_overview.html, 1999. [Online resource; last checked 09-Apr-2007].

[17] A. Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS'01)*, pages 82–89. IEEE Computer Society, 2001.

[18] A. Corradini. Real-time gesture recognition by means of hybrid recognizers. In *GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction*, pages 34–46, London, UK, 2002. Springer-Verlag.

[19] A. Corradini and P. Cohen. Multimodal speech-gesture interface for hands-free painting on virtual paper using partial recurrent neural networks for gesture recognition. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, volume III, pages 2293–2298, 2002.

[20] A. Corradini and H. Gross. Camera-based gesture recognition for robot control. *IEEE Computer Society, Joint Conference on Neural Networks (IJCNN 2000)*, 04:133–138, 2000.

[21] P. Cosi. Hybrid HMM-NN architectures for connected digit recognition. In *IJCNN (5)*, pages 85–90, 2000.

[22] M. Craven, K. Curtis, B. Hayes-Gill, and C. Thursfield. A hybrid neural network/rule-based technique for on-line gesture and hand-written character recognition. In *Proceedings of the Fourth IEEE Int. Conf. on Electronics, Circuits and Systems, Cairo, Egypt, December 15-18 1997, Vol. 2, pp. 850-853.*, 1997.

[23] M. W. Craven and J. W. Shavlik. Using neural networks for data mining. *Future Gener. Comput. Syst.*, 13(2-3):211–229, 1997.

[24] J. J. de Oliveira Junior, J. M. de Carvalho, C. O. de A. Freitas, and R. Sabourin. Evaluating NN and HMM classifiers for handwritten word recognition. *SIBGRAPI '02: Proceedings of the 15th Brazilian Symposium on Computer Graphics and Image Processing*, pages 210–217, 2002.

[25] P. Dreuw, T. Deselaers, D. Rybach, D. Keysers, and H. Ney. Tracking using dynamic programming for appearance-based sign language recognition. In *7th International Conference on Automatic Face and Gesture Recognition, FG2006*, IEEE, pages 293–298, Southampton, April 2006.

[26] S. Dreyfus. Richard Bellman on the birth of dynamic programming. *Operations Research*, 50,1:48–51, 2002.

[27] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[28] Naval Education, Professional Development Training, and Technology Center. Aviation Boatswain's Mate H. Technical Report 0504-LP-026-4040, NAVEDTRA 14311, United State's Navy, 2001.

[29] S. Eickeler, A. Kosmala, and G. Rigoll. Hidden Markov Model Based Continuous Online Gesture Recognition. In *Int. Conference on Pattern Recognition (ICPR)*, pages 1206–1208, Brisbane, 1998.

[30] J. L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

[31] M. Eysenck and M. T. Keane. *Cognitive Psychology: A Student's Handbook*. Psychology Print, 4th edition, 2000.

[32] G. Fang and W. Gao. A SRN/HMM system for signer-independent continuous sign language recognition. *FGR '02: Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 312–317, 2002.

[33] S. S. Fels and G. E. Hinton. Glove-Talk: A neural network interface between a data-glove and a speech synthesizer. *IEEE Transactions on Neural Networks*, 4(1):2–8, January 1993.

[34] G. D. Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61,(3):268–278, 1973.

[35] R. Frank, N. Davey, and S. Hunt. Time series prediction and neural networks. *Journal of Intelligent and Robotic Systems*, 31:91–103, 2001.

[36] B. Gabrys and A. Bargiela. General fuzzy min-max neural network for clustering and classification. *Neural Networks, IEEE Transactions on*, 11(3):769–783, 2000.

[37] J. Ghosh and L. Deuser. Classification of spatio-temporal patterns with applications to recognition of sonar sequences. *Neural Representation of Temporal Patterns.*, pages 221–250, 1995.

[38] K. Goto and K. Keeni. On classification of alarms from network intrusion detection system using multi-layer feed-forward neural networks. In *Neural Networks and Computational Intelligence*, pages 163–168, 2003.

[39] R M. Gray. Vector quantization. *IEEE ASSP Mag*, pages 4–29, April 1984.

[40] D. J. Hand and K. Yu. Idiot's Bayes - not so stupid after all? *International Statistical Review*, 69(3):385–399, 2001.

[41] P. A. Harling. Gesture input using neural networks. Technical report, Department of Computer Science, University of York, 1993.

[42] J. Hawkins and S. Blakeslee. *On Intelligence*. Times Books, October 2004.

[43] G. Hinton. Supervised learning in multiayer neural networks. *The MIT Encyclopedia of the Cognitive Sciences*, pages 814–816.

[44] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[45] P. Hong, M. Turk, and T. Huang. Gesture modeling and recognition using finite state machines. In *Proc. Fourth IEEE International Conference and Gesture Recognition*, page 410, March 2000.

[46] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Network*, 2(5):359–366, 1989.

[47] T. S. Huang and V. I. Pavlovic. Hand gesture modeling, analysis, and synthesis. *Proc. of International Workshop on Automatic Face and Gesture Recognition (IWAFGR), Zurich, Switzerland*, pages 73–79, 1995.

[48] R. J. Hubbold, X. Dongbo, and S. Gibson. MAVERIK - The Manchester Virtual Environment Interface Kernal. In M. Göbel, J. David, P. Slavik, and J. J. van Wijk, editors, *Virtual Environments and Scientific Visualization '96*. Springer-Verlag Wien, 1996.

[49] C. Hummels and P. J. Stappers. Meaningful gestures for human computer interaction: Beyond hand postures. In *Proceedings of the 3rd. International Conference on Face & Gesture Recognition*, page 591. IEEE Computer Society, 1998.

[50] FastTrak Polhemus Inc. *3SPACE User's Manual*. http://www.polhemus.com/ftrakds.htm, 1998.

[51] F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23-1:67–72, 1975.

[52] A. K. Jain, R. P. W. Duin, and . Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[53] F. Jelinek. *Statistical methods for speech recognition*. MIT Press, 1997.

[54] B. Jin, A. R. Hurson, and L. L. Miller. Neural network-based decision support for incomplete database systems: Knowledge acquisition and performance analysis. In *Conference on Analysis of Neural Network Applications*, pages 62–75, 1991.

[55] M. I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pages 531–546, Hillsdale, 1986. (Amherst 1986), Erlbaum.

[56] M. I. Jordan. Serial order: A parallel, distributed processing approach. Technical Report Number:8694, Institute for Cognitive Science, University of California, 1986.

[57] M. W. Kadous. *Temporal Classification: Extending the Classification Paradigm to Multivariate Time Series.* PhD thesis, The University of New South Wales, School of Computer Science and Engineering, 2002.

[58] V. Kecman. *Learning and Soft Computing, Support Vector Machines, Neural Netwrorks and Fuzzy Logic Models.* The MIT Press, 1st edition, 2001.

[59] A. Kendon. An agenda for gesture studies. *Semiotic Review of Books*, 7, 3:8–12, 1996.

[60] E. Keogh and M. Pazzani. Derivative dynamic time warping. In *First SIAM Int. Conference on Data Mining. http://www.siam.org/meetings/sdm01/pdf/sdm01-01.pdf*, 2001. [last checked 06-June-2007].

[61] E. J. Keogh and M. J. Pazzani. Scaling up dynamic time warping for data mining applications. In *Knowledge Discovery and Data Mining*, pages 285–289, 2000.

[62] M. Kipp. The neural path to dialogue acts. In *European Conference on Artificial Intelligence*, pages 175–179, 1998.

[63] J. Koh, E. Lee, and Y. Nam. Full-body motion recognition using multi-phase target reduction method. *HCI 2004(Korean)*, 2004.

[64] J. Koh and Y. Nam. Full-body motion recognition using principal component based target reduction. In *KIPS(Korean Information Processing Society) Proceedings*, volume Vol. 11 , no.1, pages 873–876, Korea, May 2004.

[65] J. Kruskall and B. Liberman. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, chapter 4, The symmetric time warping algorithm: From continuous to discrete. CSLI Publication, 1983.

[66] R. J. Kuo and K. C. Xue. A decision support system for sales forecasting through fuzzy neural networks with asymmetric fuzzy weights. *Decis. Support Syst.*, 24(2):105–126, 1998.

[67] G. Kurtenbach and E. Hulteen. Gestures in human computer communication. In *The Art and Science of Interface Design*, pages 309–317. Addison-Wesley, 1990.

[68] C. Lee and Y. Xu. Online, interactive learning of gestures for human/robot interfaces. *IEEE Int. Conf. on Robotics and Automation*, pages 2982–2987, 1996.

[69] S. W. Lee. Automatic gesture recognition for intelligent human-robot interaction. In *FGR '06: Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition (FGR06)*, pages 645–650. IEEE Computer Society, 2006.

[70] T. Lee. *Automatic Recognition of Isolated Cantonese Syllables Using Neural Networks.* PhD thesis, The Chinese University of Hong Kong, May 1996.

[71] H. Li and M. Greenspan. Continuous time-varying gesture segmentation by dynamic time warping of compound gesture models. In *HAREM2005: International Workshop on Human Activity Recognition and Modelling*, pages 35–42, 2005.

[72] M. M. Lipschultz. *Theory and Problems of Differential Geometry.* Mc Graw Hill, 1969.

[73] E. Lleida and R.C. Rose. Utterance verification in continuous speech recognition: Decoding and training procedures. In *IEEE Transactions on Speech and Audio Proc.*, volume 8-2 of *IEEE*, pages 126–139, March 2000.

[74] S. Manganaris. *Supervised Classification with Temporal Data.* PhD thesis, Vanderbilt University, School of Engineering, Computer Science Department, December 1997.

[75] S. Marcel. Hand posture recognition in a body-face centered space. In *CHI '99: CHI '99 extended abstracts on Human factors in computing systems*, pages 302–303, New York, NY, USA, 1999. ACM Press.

[76] D. McNeill. So you think gestures are nonverbal. *Psychological Review*, 93,(3):350–371, 1985.

[77] D. McNeill. *Hand and Mind: What Gestures Reveals About Thought.* Chicago: University of Chicago Press, 1992.

[78] D. Medler. A brief history of connectionism. *Neural Computing Surveys*, 1(1):61–101, 1998.

[79] D. Michie, D. J. Spiegelhalter, and C. C. Taylor. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, 1994.

[80] R. Moddemeijer. On estimation of entropy and mutual information of continuous distributions. In I. T. Young et. al., editor, *EUSIPCO-86*, pages 1033–1035, Amsterdam (NL), September 2-5 1986. North-Holland.

[81] T. K. Moon. The expectation-maximization algorithm. *IEEE Signal Processing Mag*, 13,(6):47–60, 1996.

[82] K. Murakami and H. Taguchi. Gesture recognition using recurrent neural networks. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 237–242, New York, NY, USA, 1991. ACM Press.

[83] K. P. Murphy. *Dynamic Bayesian networks: representation, inference and learning.* PhD thesis, UC Berkeley, Computer Science Division, 2002.

[84] Y. Nam and K. Wohn. Recognition of space-time hand gestures using hidden markov model. In *ACM Symposium on Virtual Reality Software and Technology*, pages 51–58, 1996.

[85] J. Nespoulous, P. Perron, and A. R. Lecours. *The Biological Foundations of Gestures: Motor and Semiotic Aspects.* Lawrence Erlbaum Associates, 1986.

[86] J. Park and I. W. Sandberg. Approximation and radial-basis-function networks. *Neural Computation*, 5(2):305–316, 1993.

[87] V. Pavlovic, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695, 1997.

[88] R. D. Peacocke and D. H. Graf. An introduction to speech and speaker recognition. *Computer*, 23(8):26–33, 1990.

[89] A. Perrotta. Interpretation of hand gestures in HCI. Master's thesis, RMCS,Cranfield University, August 2005.

[90] U. T. Place. The role of the hand in the evolution of language. *Psycoloquy, Language Gesture (1)*, 11-07, 2000.

[91] A. B. Portiz. Hidden markov models: A guided tour. *IEEE Proceedings of the ICASSP*, pages 7–13, 1988.

[92] F. Quek. Toward a vision-based hand gesture interface. *Virtual Reality Software and Technology Conference*, pages 17–31, Aug 1994.

[93] F. Quek. Eyes in the interface. *Image and Vision Computing (IVC)*, 13(6):511–525, August 1995.

[94] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[95] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[96] L. Rabiner and B. Juang. An introduction to hidden markov models. *ASSP Magazine, IEEE*, 3, (1):4–16, Jan 1986.

[97] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77, (2):257–286, Feb 1989.

[98] L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson. Considerations in dynamic time warping algorithms for discrete word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26:572–582, December 1978.

[99] C. A. Ratanamahatana and E. Keogh. Everything you know about dynamic time warping is wrong. *Third Workshop on Mining Temporal and Sequential Data, in conjunction with the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 50–60, 2004.

[100] S. Reiter, B. Schuller, and G. Rigoll. A combined LSTM-RNN-HMM approach for meeting event segmentation and recognition. In *Proceedings of the 31st International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages II–393–II–396, Toulouse, France, May 2006.

[101] G. Rigoll, A. Kosmala, and S. Eickeler. High performance real-time gesture recognition using hidden markov models. In *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, pages 69–80, London, UK, 1998. Springer-Verlag.

[102] G. Rigoll, A. Kosmala, J. Rottland, and C. Neukirchen. A Comparison Between Continuous and Discrete Density Hidden Markov Models for Cursive Handwriting Recognition. In *Int. Conference on Pattern Recognition (ICPR)*, volume 2, pages 205–209, Vienna, 1996.

[103] I. Rish. An empirical study of the naive Bayes classifier. In *IJCAI-01 workshop on "Empirical Methods in AI"*, pages 41–46, 2001.

[104] T. S. Rognvaldsson. A simple trick for estimating the weight decay parameter. In *Neural Networks: Tricks of the Trade*, pages 71–92, London, UK, 1998. Springer-Verlag.

[105] J. W. Ruffner, D. E. Shiraev, P. A. Morey, J. E. Fulbrook, A. D. Struckhoff, and T. M. Franz. Automating hand signal recognition: Transforming helicopter signaling skills training. *The Interservice/Industry Training, Simulation and Education Conference (I/ITSEC)*, 2005.

[106] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations*, pages 318–362, 1986.

[107] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition, 2003.

[108] N. Saito and R. R. Coifman. *Local feature extraction and its applications using a library of bases*. PhD thesis, Yale University, Dept. of Mathematics, 1994.

[109] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. In A. Waibel and K.-F. Lee, editors, *Readings in Speech Recognition*, pages 159–165. Kaufmann, San Mateo, CA, 1990.

[110] A. M. Salem, M. M. Syiam, and A. F. Ayad. Unsupervised artificial neural networks for clustering of document collections. In *ICEIS (2)*, pages 383–392, 2004.

[111] U. Salihoglu. Chaos in small recurrent neural networks : theoretical and practical studies. Technical report, Universit'e Libre De Bruxelles , Facult'e Des Sciences, D'eparement D'informatique, 2004.

[112] A. Sandberg. Gesture recognition using neural networks. Master's thesis, Stockholm University, 1997.

[113] V. Scarcia. An Investigation into Gesture Interpretation for Flight Deck Officer Training. Master's thesis, RMCS,Cranfield University, 2001.

[114] C. Shahabi, L. Kaghazian, S. Mehta, A. Ghoting, G. Shanbhag, and M. L. McLaughlin. *Touch in Virtual Environments: Haptics and the Design of Interactive Systems*, chapter Understanding of User Behavior in Immersive Environments, pages 238–259. Prentice Hall, 2001.

[115] S. C. Shapiro. Artificial intelligence. In D. Hemmendinger A. Ralston, E. D. Reilly, editor, *Encyclopedia of Computer Science*, pages 89–93. Van Nostrand Reinhold, New York.

[116] H. F. Silverman and D. P. Morgan. The application of dynamic programming to connected speech recognition. *IEEE ASSP Mag*, 61,(3):7–25, 1990.

[117] D. T. Sodiri. Gesture Interpretation in Virtual Environment Flight Deck Officer Training. Technical Report AMOR 2002/5, AMOR/ESD, RMCS, Cranfield University, 2002.

[118] D. T. Sodiri. Real Time Dynamic Gesture Recognition. Technical Report AMOR 2004/1, AMOR/ESD, RMCS, Cranfield University, 2004.

[119] D. T. Sodiri and V. V. S. S. Sastry. On the interpretation of gestures arising in flight deck officers training. In *Proceedings of the Thirteenth Conference on Behaviour Representation in Modelling and Simulation, Virginia, USA*, 2004.

[120] D. T. Sodiri and V. V. S. S. Sastry;. On-line recognition of isolated gestures of Flight Deck Officers (FDO). In *Transactions on Engineering, Computing and Technology, Budapest, Hungary*, volume 13, pages 119–126, 2006.

[121] D. T. Sodiri and V. V. S. S. Sastry;. Recognition Machine (RM) for on-line and isolated Flight Deck Officer (FDO) gestures. *IJIT, International Journal of Intelligent Technology*, 13:138–145, 2006.

[122] T. Starner and A. Pentland. Visual recognition of american sign language using hidden markov models. In *Int'l Workshop Automatic Face and Gesture Recognition*, pages 189–195, 1995.

[123] T. Starner, J. Weaver, and A. Pentland. A wearable computer based American Sign Language recognizer. *Lecture Notes in Computer Science*, 1458:84–96, 1998.

[124] M. H. Stone. The theory of representations for boolean algebras. *Transactions of the American Mathematical Society*, 40:37–111, 1937.

[125] D. J. Sturman, D. Zeltzer, and S. Pieper. Hands-on interaction with virtual environments. In *Proceedings of the 2nd annual ACM SIGGRAPH symposium on User interface software and technology*, pages 19–24. ACM Press, 1989.

[126] S. B. Suresh. Flight Deck Officer-Gesture Interpretation. Master's thesis, RMCS,Cranfield University, 2000.

[127] K. Symeonidis. Hand gesture recognition using neural networks. Master's thesis, University of Surrey, Centre for Vision, Speech and Signal Processing, August 2000.

[128] G. Thimm and E. Fiesler. Neural network pruning and pruning parameters. In T. Furuhashi, editor, *The 1st Workshop on Soft Computing*, Furo-cho, Chikusa-ku, Nagoya 464-01, Japan, August 1996. Dept. of Information Electronics Nagoya University.

[129] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura. Speech parameter generation algorithms for HMM-based speech synthesis. *Proc. ICASSP*, 3, (1):1315–1318, June 2000.

[130] E. Trentin and R. Cattoni. Learning perception for indoor robot navigation with a hybrid hidden markov model/recurrent neural networks approach. *Connect. Sci.*, 11(3-4):243–265, 1999.

[131] J. Triesch and C. von der Malsburg. Robust classification of hand postures against complex backgrounds. In *FG '96: Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition (FG '96)*, page 170, Washington, DC, USA, 1996. IEEE Computer Society.

[132] A. Trott. An Investigation into the Use of Virtual Reality for Naval Flight Deck Operations Training. Master's thesis, RMCS,Cranfield University, 1999.

[133] M. Turk. *Handbook of virtual environments: Design, implementation, and applications*, chapter Gesture recognition, pages 223–238. Mahwah, NJ: Lawrence Erlbaum Associates, Inc., 2002.

[134] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, pp. 586-591, June 1991.*

[135] P. Vamplew. Recognition of sign language gestures using neural networks. In *The First European Conference on Disability, Virtual Reality and Associated Technologies, http://www.icdvrat.reading.ac.uk/1996/papers/1996_04.pdf*, 1996. [Online resource; last checked 06-June-2007].

[136] P. Vamplew. *Recognition of Sign Language Using Neural Networks.* PhD thesis, University of Tasmania, 1996.

[137] P. Vamplew and A. Adams. Recognition and anticipation of hand motions using a recurrent neural network. In *Proceedings of IEEE International Conference on Neural Networks, vol 3 pages:2904-2907*, 1995.

[138] A. T. Vemuri and M. M. Polycarpou. Neural-network-based robust fault diagnosis in robotic systems. *IEEE Transactions on Neural Networks*, 8(6):1410–1420, November 1997.

[139] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Information Theory*, IT-13:260–269, 1967.

[140] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang. Phoneme recognition using time-delay neural networks. In A. Waibel and K.-F. Lee, editors, *Readings in Speech Recognition*, pages 393–404. Kaufmann, San Mateo, CA, 1990.

[141] D. Wang. Temporal pattern processing. *The handbook of brain theory and neural networks*, pages 967–971, 1998.

[142] T. Wang, N. Zheng, Y. Li, Y. Xu, and H. Shum. Learning kernel-based HMMs for dynamic sequence synthesis. *Graph. Models*, 65(4):206–221, 2003.

[143] R. Watson. A survey of gesture recognition techniques. Technical Report TCD-CS-93-11, Department of Computer Science, Trinity College, Dublin, 1993.

[144] P. Werbos. *Beyond regression: new tools for prediction and analysis in the behavioural sciences*. PhD thesis, Harvard University, 1974.

[145] T. Wessels and C. W. Omlin. Refining hidden markov models with recurrent neural networks. In *IJCNN '00: Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)-Volume 2*, page 2271, Washington, DC, USA, 2000. IEEE Computer Society.

[146] T. Westeyn, H. Brashear, A. Atrash, and T. Starner. Georgia Tech Gesture Toolkit: Supporting experiments in gesture recognition. In *International Conference on Perceptive and Multimodal User Interfaces, ICMI*, pages 85–92, 2003.

[147] G. M. White. Dynamic programming, the Viterbi algorithm, and low cost speech recognition. *in Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 413–417, 1978.

[148] J. Wilpon, L. R. Rabiner, C. Lee, and E. R. Goldman. Automatic recognition of keywords in unconstrained speech using hidden markov models. In *IEEE Transactions on Acoustics, Speech, and Signal Proc.*, volume 38:11, pages 1870–1878, 1990.

[149] Personal Communication with Ministry of Defence of United Kingdom. Marshalling Signals, BRd 766 part 2 chapt 18, Military Training Publication. It can be obtained by request from www.defenceimages.mod.uk.

[150] Personal Communication with Shingakusha Co. Ltd. Images are obtained from http://www.sing.co.jp, 2006.

[151] S. G. Wysoski, M. V. Lamar, S. Kuroyanagi, and A. Iwata. A rotation invariant approach on static-gesture recognition using boundary histograms and neural networks. In *Neural Information Processing, ICONIP 2002*, pages 2137–2141, 2002.

[152] L. Xiao-Hui and C. Chin-Seng. Rejection of non-meaningful activities. In *FGR '06: Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition (FGR06)*, pages 189–196, Washington, DC, USA, 2006. IEEE Computer Society.

[153] H. Yang, A-Y. Park, and S-W. Lee. Robust spotting of key gestures from whole body motion sequence. In *FGR '06: Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition (FGR06)*, pages 231–236. IEEE Computer Society, 2006.

[154] J. Yang and Y. Xu. Hidden markov model for gesture recognition. Technical Report CMU-RI-TR-94-10, The Robotics Institute, Carneige Melon University, 1994.

[155] K. Yang and C. Shahabi. A PCA-based similarity measure for multivariate time series. In *Proceedings of the 2nd ACM international workshop on Multimedia databases*, pages 65–74. ACM Press, 2004.

[156] M. Yang and N. Ahuja. Recognizing hand gesture using motion trajectories. In *Conference on Computer Vision and Pattern Recognition*, pages 1466–1472, 1999.

[157] S. Young, G. Evermann, T. Hain, and P. Woodland. *The HTK Book, 3.2.1.* Cambridge Research Laboratory Ltd, On-line Manual from http://htk.eng.cam.ac.uk, 2002.

[158] X. Zeng and T. R. Martinez. A noise filtering method using neural networks. In *Proceedings of the International Workshop of Soft Computing Techniques in Instrumentation, Measurement and Related Applications*, pages 26–31, 2003.

[159] Y. Zeng. Dynamic time warping digit recognizer. Master's thesis, Department of Electrical and Computer Engineering,Mississippi State University, November 2000.