HAMID ISAKHANI

BIOMIMETIC VISION-BASED COLLISION AVOIDANCE SYSTEM FOR MAVs

MSc by Research Thesis

School of Aerospace, Transport and Manufacturing

CRANFIELD UNIVERSITY

School of Aerospace, Transport and Manufacturing

MSc by Research Thesis

Academic Year 2016-2017

HAMID ISAKHANI

BIOMIMETIC VISION-BASED COLLISION AVOIDANCE SYSTEM FOR MAVs

Supervisors:

Prof. Nabil Aouf

Dr. James F. Whidborne

May, 2017

بسم الله الرحمن الرحيم

“In the Name of God, the Most Gracious, the Most Merciful”.

ACKNOWLEDGEMENTS

I hereby express my sincere gratitude to my mentor and supervisor, Prof. Nabil Aouf for facilitating this research opportunity with ample fortitude, and granting plethora of guidance and support throughout the course. Also, I would like to extend my tribute to Dr. James F. Whidborne, my co-supervisor for accommodating me in his gratifying research group and contributing boundless assistance to clinch this thesis.

I gratefully thank my technical adviser, Dr. Odysseas Kechagias-Stamatis for his prudent reviews, proposals, and instructions to motivate, develop, and improve my notions all through this research project.

Ultimately, I would like to honour my parents, Azam and Mansour Isakhani for being the principal elements of commencement and conclusion of all my life endeavours.

ABSTRACT

This thesis proposes a secondary collision avoidance algorithm for micro aerial vehicles based on luminance-difference processing exhibited by the Lobula Giant Movement Detector (LGMD), a wide-field visual neuron located in the lobula layer of a *locust*'s nervous system. In particular, we address the design, modulation, hardware implementation, and testing of a computationally simple yet robust collision avoidance algorithm based on the novel concept of quadfurcated luminance-difference processing (QLDP). Micro and Nano class of unmanned robots are the primary target applications of this algorithm, however, it could also be implemented on advanced robots as a fail-safe redundant system. The algorithm proposed in this thesis addresses some of the major detection challenges such as, obstacle proximity, collision threat potentiality, and contrast correction within the robot's field of view, to establish and generate a precise yet simple collision-free motor control command in real-time. Additionally, it has proven effective in detecting edges independent of background or obstacle colour, size, and contour. To achieve this, the proposed QLDP essentially executes a series of image enhancement and edge detection algorithms to estimate collision threat-level (spike) which further determines if the robot's field of view must be dissected into four quarters where each quadrant's response is analysed and interpreted against the others to determine the most secure path. Ultimately, the computation load and the performance of the model is assessed against an eclectic set of off-line as well as real-time real-world collision scenarios in order to validate the proposed model's asserted capability to avoid obstacles at more than 670 mm prior to collision (real-world), moving at 1.2 ms^{-1} with a successful avoidance rate of 90% processing at an extreme frequency of 120 Hz, that is much superior compared to the results reported in the contemporary related literature to the best of our knowledge.

Contents

List of Figures

List of Tables

Nomenclature

Symbols

F	Thrust force
g	Acceleration due to gravity
L	Length
m	Mass
r	Propeller radius
k	Rotor blade configuration
δ	Proximity factor
ϕ	Bank angle
ψ	Yaw angle
γ	Flight path angle
λ	Scaling factor
Ω_n	Motor speed
$\dot{\phi}$	Roll rate
$\dot{\psi}$	Yaw rate
ρ	Mitigation factor
τ	Torque
C_{D0}	Minimum drag coefficient
x, y, z	Aircraft centre of gravity coordinates in earth axis

Acronyms

AHE	Adaptive Histogram Equalisation
AOA	Angle of Attack
APM	Ardu-Pilot Mega

CAS	Collision Avoidance System
CDS	Collision Detection System
CPU	Central Processing Unit
DCMD	Descending Contralateral Movement Detector
DDA	Detect, Decide and Act
DDD	Dull Dirty Dangerous
DOF	Degree of Freedom
EMD	Elementary Motion Detector
FCU	Flight Control Unit
FFI	Feed Forward Inhibition
FLC	Fuzzy Logic Control
FOV	Field of View
FPGA	Field Programmable Gate Array
GCS	Ground Control Station
GPS	Global Positioning System
GPU	Graphics Processing Unit
GUI	Graphic User Interface
HLP	High Level Processor
HMI	Human Machine Interface
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
LFTC	Large-Field Tangential Cells
LGMD	Lobula Giant Movement Detector
LIDAR	Light Detection and Ranging
LLP	Low Level Processor
LMC	Large Monopolar Cell
LiPo	Lithium Polymer
LQR	Linear Quadratic Regulator
LTP	Local Trajectory Planner
LVF	Long Visual Fibre
MAV	Micro Air Vehicle
MIDCAS	Mid Air Collision Avoidance System
MOSFET	Metal–Oxide–Semiconductor Field-Effect Transistor
MPC	Model Predictive Control
NAS	National Airspace System

OODA	Observe, Orient, Decide, and Act
PCAS	Portable Collision Avoidance System
PRM	Probabilistic Road Maps
PTAM	Parallel Tracking and Mapping
QLDP	Quadfurcated Luminance-Difference Processing
RAM	Random Access Memory
RHC	Receding Horizon Control
ROS	Robot Operating System
RPA	Remotely Piloted Aircraft
RRTs	Rapidly-Exploring Random Trees
SAA	Sense and Avoid
SAAFT	Sense and Avoid Flight Tests
SPI	Serial Peripheral Interface
SVF	Short Visual Fibres
SVO	Semi-direct Visual Odometry
TCAS	Traffic Alert and Collision Avoidance System
TVLQR	Time Varying Linear Quadratic Regulator
UAS	Unmanned Aircraft Systems
UAV	Unmanned Aerial Vehicle
VFR	Visual Flight Rules
VINS	Vision-aided Inertial Navigation Systems
VMC	Visual Meteorological Conditions
VSLAM	Visual Simultaneous Localization and Mapping

1

Introduction

"Those who are inspired by a model other than Nature,
a mistress above all masters, are labouring in vain."
– Leonardo da Vinci

The manoeuvring capability of autonomous flying robots are not yet comparable to that of insects in a complex cluttered environment. Inspired by this observation, this research presents the development of a reactive autonomous collision avoidance algorithm for aerial and ground robots based on insect neuronal vision architecture. To achieve a fully autonomous navigation capability, a micro/nano robot requires an efficient, brisk, and proactive navigation system despite its limited hardware resources due to weight and energy constraints. In order to address such rather contradictory challenges, our approach is inspired by flying insects, which possess effective flight controllability in complex environments despite their limited weight (payload) and minuscule nervous system (processing power).

1.1 Motivation

Unmanned autonomous system's (UAS) existence solely depends on its superiority over manned systems with respect to cost, maintainability, and expendability. A system is designed from the outset to perform a particular role, thus the designer must decide the nature and type of system most suitable to perform the role and, in particular, whether the role may be better achieved with a manned or an unmanned solution, hence, it is futile to conclude that a UAS always has an advantage or disadvantage over its counterpart manned system.

An old military dictum links the use of UAS to roles which are dull, dirty or dangerous (Austin 2010). Nevertheless, for an extensive assessment, we must consider the economical, covert, diplomatic, research and environmentally critical roles as the prominent practical aspects of a UAS. Although in most applications a UAS proves preferable, the unfledged automation technology causes their efficiency and robustness to be weak when compared to their manned counterpart. Hence, the hankering to improve the robustness and reliability of unmanned systems has expanded incredibly over the recent years. The boundaries of technological development in the field of Artificial Intelligence (AI) is not limited to mimicking of human capabilities but also to mitigate the prevailing pilot errors and deficiencies in manual systems. One of the most arduous elements of the machine-autonomy development is formulating a robust collision avoidance algorithm to prevent catastrophic mission failure due to a dynamic/static obstacle collision.

Inspired by nature to address the aforementioned challenge in this research, a simple yet reliable collision avoidance system is designed based on an insect's vision architecture which has been proven to possess a simple anatomy thus eliminating the computational complexity of the system. Insects' elementary techniques to orientate in space and accomplish complex tasks such as obstacle avoidance, escape from predators and landing, with such limited neuronal resources provides sufficient evidence to bolster the desire to explore this area of research.

Hence, the fundamental source of motivation for this research is the simple yet agile architecture of an insect (*locust*) nervous system which lays the foundation for the development of this computationally simple yet effective collision avoidance algorithm to detect obstacle edges and establish collision-threat-level within the vehicle's field of view.

1.2 Aims and Objectives

This research aims to develop a collision avoidance algorithm that is inspired by an insect's vision-based collision detection system. This algorithm is designed to serve as a secondary reactive collision avoidance for micro and nano class of UAVs, and as an auxiliary collision avoidance for Remotely Piloted Aircraft (RPAs), or autonomous UAVs in order to enable collision-free flight in GPS-denied complex cluttered environments. Hence to achieve this, the following objectives have been formulated:

1. The design and development of a computationally efficient algorithm to detect complex obstacle/background edges biologically-inspired by a *locust* visual neuron known as

Lobula Giant Movement Detector (LGMD).

2. Modulation and adaptation of the above computational model for aerial/ground robot applications considering primary constraint such as limited payload delivery (processing power, sensor technology).

3. Interpretation of the collision alarms (spikes generated by the collision detection model) in terms of flight control commands based on an indigenous quartet image correlation processing called QLDP.

4. Testing and performance assessment of the finalised algorithm implemented on a fabricated ground robot to validate the claimed capabilities for real-world real-time applications.

1.3 Research Challenges

One of the most compelling challenges in development of an autonomous navigation system for micro robots is establishing a consistent design trade-off between computational power, size and cost. In autonomous systems, integration of processing units and their respective sensors within a confined micro sized block compromises a substantial amount of functionality. However, biological systems on the other hand, overcome size constraints without compromising performance by eliminating unsuccessful individuals through more than 350 million years of evolution. Similarly, this thesis is inspired by the basic computational model of an evolved visual neuron to address an engineering problem (autonomous navigation) with bare minimum processing resources. In particular, we are keen to understand the biological mechanisms bolstering a vision-based collision detection and avoidance with an ultimate goal of designing an autonomous aerial robotic system that can generate safe manoeuvres through a GPS-denied indoor cluttered environment.

Vision-based navigation is a challenging task that necessitates real-time processing of an input dataset that is acquired from an imaging sensor. In many cases, conventional processing units are implemented in autonomous navigation. However, the requirement of an agile analogue to digital converter (ADC) to communicate the data from a serial digital processor to a high-resolution sensor (like CMOS) convolves the system architecture, that further necessitates an advanced algorithm capable of real-time image-data processing in order to generate flight control commands. A system that is required to operate on a limited computation power, as in an autonomous micro aerial vehicle, the particular task of fast data transfer and computation is the primary constraint. However, an insect visual system where the micro sized sensors and processing neurons are carefully arranged in a dense architecture, introduces an exquisite alternative concept for both sensory data acquisition and processing. This system has evolved over millions of years to provide us with an extremely advanced neuronal architecture that can be implemented in autonomous navigation algorithm design.

1.4 Proposed Methodology

Flying insects are the most promising model systems to address the optimum computation demands for vision-based navigation tasks, such as obstacle avoidance. They inspire development of faster and economical algorithms that eliminates the need for computation extensive methodologies such as Lucas-Kanade optic-flow model, or expensive devices and hardware technologies such as 3D laser range finders.

Flies perform brisk manoeuvres to avoid a predator or an obstacle within a fraction of a second. Reactive avoidance decisions are generated and interpreted as evasive manoeuvres within only a few milliseconds in fly's brain (limited computation resources) (Armstrong & van Hemert 2009). As such, the *locust* of *Acrididae*, a predominant family of grasshoppers possess similar brisk reactions using their wide-field visual neuron located in the Lobula layer of their nervous system that essentially executes a series of edge detection processes to detect obstacles or predators across its field of view. Inspired by these processes, an algorithm named Quadfurcated Luminance-Difference Processing (QLDP) is designed and implemented on a robot to address one of the most challenging robotic-navigation problems that is reactive collision avoidance.

1.5 Tools and Software Requirements

Design, analysis, modulation and implementation of the proposed algorithm are comparatively economical, hence, its experimental validation does not necessitate exceptional simulation tools and software. The software requirement for design and development of the algorithm is,

1. MathWorks MATLAB Simulink with the following toolboxes:

    - Computer Vision System
    - Image Acquisition

- Image Processing
- Embedded Coder
- MATLAB Coder
- Control System Toolbox

And the system implementation requires the following hardware,

1. Ground robot (with Ackermann steering).
2. Micro-controller development board (Arduino Nano3)
3. CMOS sensor (USB camera)
4. 9g Servo motor and a Motor shield
5. Ground Station PC

1.6 Thesis Organization and Contributions

The structure of this thesis constitutes 7 chapters including the introduction chapter that defines the fundamental concept of the project followed by,

- Chapter 2, delivers insights in a detailed literature review of nature-inspired vision-based UAV collision avoidance systems, and the contemporary related work.
- Chapter 3, introduces target research platforms for our proposed algorithm that involves flight control-command generation and dynamics of a quadrotor UAV, and fabrication of a ground robot used in chapter 6 for validation purposes.
- Chapter 4, describes the anatomy of a *locust* LGMD neuron and its computational model proposed in associated research that inspires the fundamental design of QLDP introduced in this chapter.
- Chapter 5, proposes the modulation, adaptations and modifications required to achieve the apt collision avoidance model, further visualising layer-by-layer processing of the QLDP.
- Chapter 6, presents the implementation and validation of the QLDP assessment results for both off-line as well as real-world real-time applications.

- Chapter 7, discusses the conclusion and summary of results, the model limitations, and the scope for future work.

The most important contribution of this thesis being an optimised algorithm to execute autonomous collision avoidance onboard micro and nano robots is submitted for publication as the following article,

- Hamid Isakhani, Nabil Aouf, Odysseas Kechagias-Stamatis and James F. Whidborne, "A Quadfurcated Visual Collision Avoidance System for an Autonomous Micro Aerial Vehicle", IEEE Transactions on Robotics, submitted, February, 2017.

2

Literature Review

"The real danger is not that computers will begin to think like men,
but that men will begin to think like computers."
– Sydney Harris

2.1 Introduction

This thesis aims to present a solution to the challenging task of collision-free indoor micro UAV flying through complex cluttered environment with limited resources onboard, this task remains a fundamental research inquest as its applicability extends to the entire field of robotics (ground, aerial, marine), control theory, and computer vision. The system introduced here, fuses an indigenous control command generator based on a quartet image correlation processing, with an insect-inspired perception to control and manoeuvre a micro sized robot towards a collision-free path.

In this research, a ground robot (with Ackermann steering) is used to implement and validate the developed algorithm, this platform is fabricated to save time and resources (constraints) during the preparation of this thesis. The robot exhibits dynamics similar to a four-wheeled vehicle thus only 3 degrees of freedom necessitating adaptation of the algorithm to bifurcate the field of view (explained in the following chapters) instead of quadfurcation.

The core of this data processing algorithm is highly efficient and is based on an insect-inspired visual perception due to the fact that insects possess an extremely simple yet effective neural network actively processing information and generating brisk responses at high rates making them the most suitable solution for aerial robot navigation in GPS-denied unknown environments.

2.2 Vision-Based Collision Avoidance Systems

Collision avoidance is a multi-tier mechanism at the core of automotive and aviation safety where the key tiers are See and Avoid (SAA) capabilities (Mcfadyen & Mejias 2015) accounting for crucial challenges in autonomous aerial navigations. Pilot's vision, a fail-safe feature, is considered to be one of the fundamental characteristics required during flight to avoid an abrupt catastrophe when all the primary and secondary communication, navigation and collision avoidance systems fail. In short, it is a form of decentralised agile collision avoidance system in which the pilot identifies and avoids any unplanned static or dynamic collision threat. This relies solely on the pilot's visual system, knowledge, skills, experience and composure to conduct regulatory procedures accurately. Thus the function of SAA is to ensure a particular type of CAS constrained by pilot behaviour and potential, further agglomerating a subset of objectives including Observe, Orient, Decide and Act (OODA) or Detect, Decide and Act (DAA) (Hutchings et al. 2007), which are the subsystems of a generic see and avoid unit where seeing or detection function involves the acquisition of visual cues and identifying any potential collision threat, but avoidance on the other hand, involves the decision making and execution in order to effectively respond to the threat. Figure 2.1 illustrates a collision avoidance system based on a generic human vision.

However, with all its complexity and astuteness, a human-supervised collision avoidance system exhibits a number of deficiencies which must be addressed within an emulated autonomous model. Hutchings et al. (2007) investigated the pilot reaction time to collision threat and found that an average human pilot requires 40.4 seconds to initiate a rescue manoeuvre from the moment a potential collision threat is noticed, further considering other prominent deficiencies of a pilot controlled CAS, we can conclude,

- Prolonged response time with a limited line of sight and blind spots
- Narrow field of view with a constricted response frequency
- Necessitating optimum brightness and contrast ratio

Hence, emulation of a vertebrate's vision system is proven inconsistent and to overcome the afore mentioned shortcomings, simpler yet effective vision architectures of invertebrates

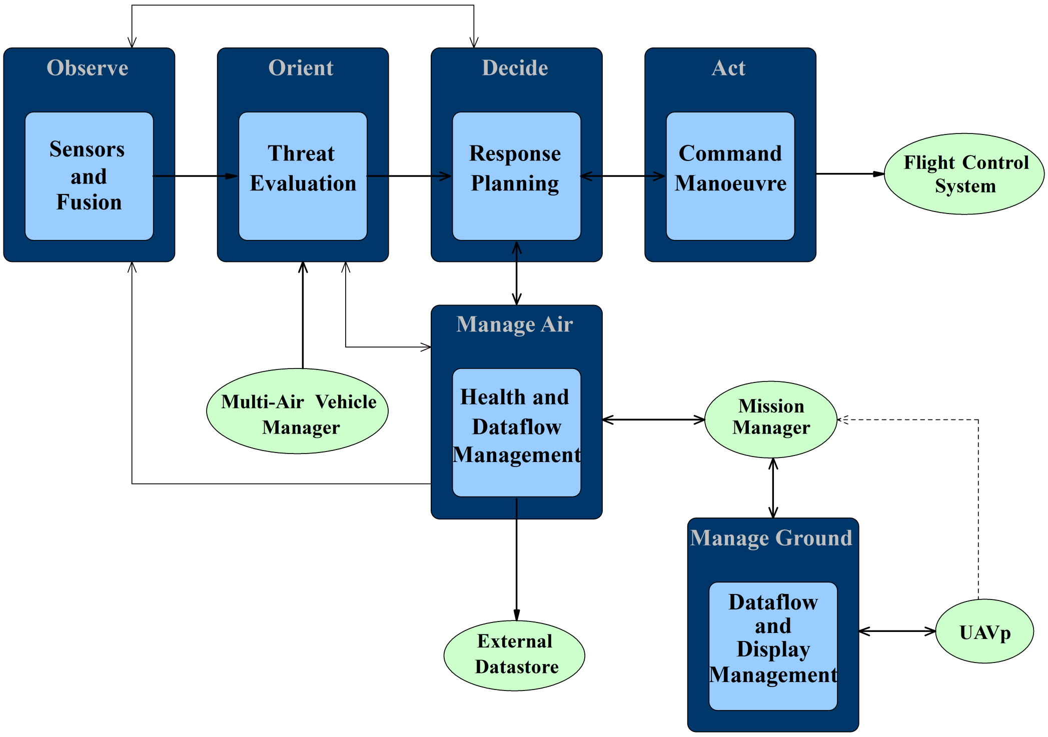


Figure 2.1: See and Avoid Architecture (from Hutchings et al. (2007))

must be decoded and implemented, for example, the model of a *locust (Locusta migratoria)* visual system presented in Rind & Bramwell (1996) (detailed demonstration in the following sections). Similarly, potential insects such as *Condylostylus* have proven to possess brisk reflexes with an average response time of 5 milliseconds which is $50\times$ faster compared to an average reaction time of a human being (Sourakov 2011).

2.3 Invertebrate's Vision

Several distinct optic lobe architectures have been identified in the animal kingdom, where the vertebrates possess only one of these designs and invertebrates feature all ten, ranging from simple arrangement of elementary photoreceptors, to advanced and sophisticated compound eyes bolstered by a convoluted range of vision-based behaviours. Further, invertebrates can recognise high-speed obstacles as they possess an impeccable set of light sensors, a wide spectrum of colour perceivers, and an ingenious polarised-light cue detectors (Wolf et al. 1980). The basic principles involved in visual data acquisition and processing are necessarily identical across the entire animal kingdom, involving neural plots and plans, some of which have inspired successful neuromorphic designs for robotic navigation (Horridge 1980). Hence invertebrate vision architecture, insect in particular, is a comprehensive fusion of our contemporary knowledge regarding how they perceive their surrounding environment, their processing principles, and how visual perception is implemented in their daily struggle to survive.

In flies, visual cues are processed initially by the photoreceptors that detect light, followed by successive computations performed by the optic lobes concealed within each eye as shown in Figure 2.2 (Eichner 2011). The photoreceptors detect illuminations, hyperpolarize and communicate luminance information to parallel processing pools in the primary neuropil (lamina). Further, the detection circuit located in the secondary neuropil (medulla) receives majority of input signals from lamina monopolar cells, the medulla. Here, a circuit picks up the information to perform detection of motion. The tertiary neuropil (lobula complex) consists of lobula and lobula-plate which receives the signals from previous layer to detect motion direction using the Large-Field Tangential Cells (LFTCs) via Elementary Motion Detectors (EMDs). LFTCs spatially integrate the output of presynaptic local circuits (Single & Borst 1998).

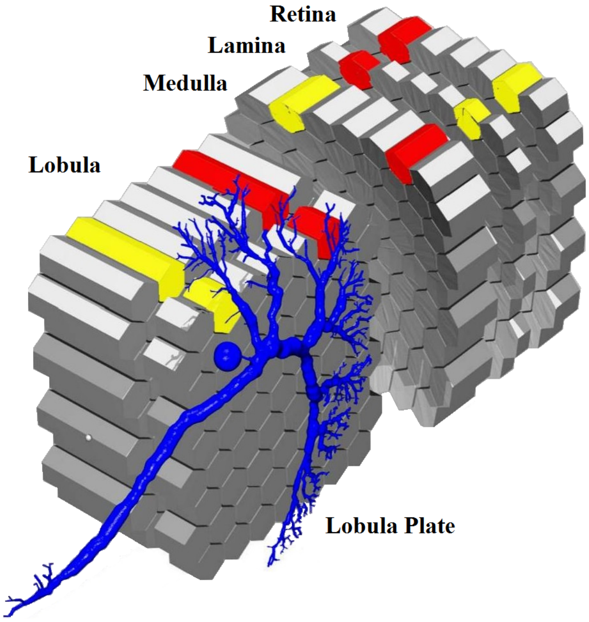


Figure 2.2: Compound Eye Optic Lobe (from Eichner (2011))

This neural network is implemented symmetrically on both sides of the insect's nervous system involving a comprehensive heterolateral contact among the lobula plates. In addition to the elementary connections within lobula plates, LFTCs synapse onto the descending neurons and neck muscle motor neurons to provide control command signals to neck, legs, and flight manoeuvring motor neurons (Strausfeld & Gronenberg 1990). However, to make this literature more specific, in the following section we review the *locust* wide-field visual neuron that decisively detects motion (Lobula Giant Movement Detector), and the neural network's physiological behaviour which has been decoded and implemented in computer vision by scientists over the past two decades.

2.4 Lobula Giant Movement Detector

A *locust* visual neuron consists of an active neural network called the Lobula Giant Movement Detector (LGMD) that responds selectively to an approaching object on collision course. LGMD facilitates flights in vast and dense swarms without colliding, simultaneously escaping an approaching predator (Cuadri et al. 2005). It is a reciprocally paired motion sensitive neuron that responds robustly to images of objects approaching on a collision course by integrating input signals from the photoreceptors. More than two decades ago, Rind & Bramwell (1996) pioneered the decoded computational model of an LGMD , illustrated in Figure 2.3.

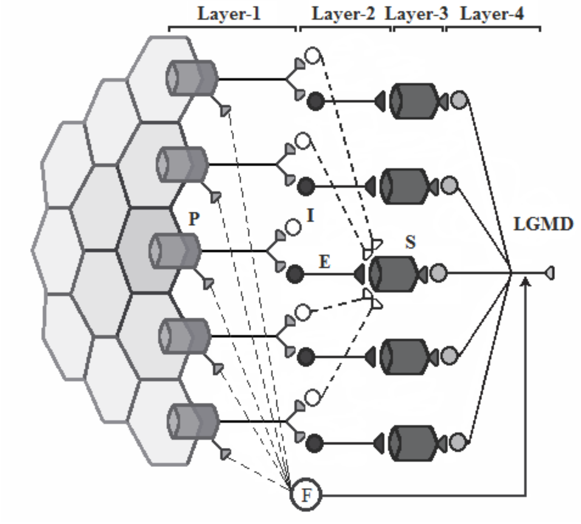


Figure 2.3: LGMD Neural Network (from Rind & Bramwell (1996))

This model has continued to evolve ever since, with the contemporary version being implemented in ground robot vision and autonomous cars involving eclectic set of adaptations and modifications with respect to their applied field. One of the outstanding features of LGMD is the neuron's capability to detect direction of an obstacle's motion involving approach, translation, and recession (Blanchard et al. 2000), which is achieved by the integration of a feed-forward inhibition, and ON/OFF channels. However, greater robustness, precision, and computational simplicity is achieved by enhancing these connections with complementary modules that emulate the exact intricate behaviour of an effective insect vision suitable for UAS applications, which ultimately inspires the core design of the data processing demonstrated in this thesis.

2.5 LGMD Optimisation Modules

To develop an optimised computer vision algorithm, the previous model flaws must be enumerated and the techniques to overcome those challenges, investigated. Continuous attempts are being made to contribute towards modification of the LGMD, including the pioneers of the original model, Rind et al. whose work in 2014 claimed that, apart from the conventional DCMD (Descending Contralateral Movement Detector) and LGMD, there are additional neurons in the *locust*'s optic lobe that respond to looming stimuli and have been implicated in triggering evasive responses, naming the additional neural network, LGMD2 (Sztarker & Rind 2014), further they provide evidences to prove that the two share many key features including selective responses to looming stimuli, but neither its role in overall behaviour nor its postsynaptic target neurons have been studied in detail yet, offering a potential subject of research for neuromorphic engineers.

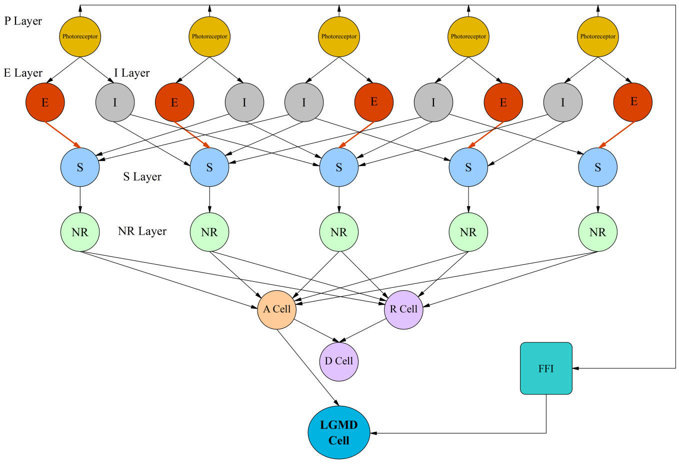


Figure 2.4: LGMD Model from Silva et al. (2014)

In Silva et al. (2014), a modified LGMD architecture is proposed, integrating two previous LGMD models to implement features such as noise immunity introduced in Yue & Rind (2006) and motion direction detection demonstrated in Meng et al. (2009). The optimised model is validated against a set of eclectic test cases (visual stimuli), where a successful filtering of isolated excitations is performed to prevent the perturbations from contributing to the excitation of the LGMD cell. Silva goes on to demonstrate that the neural architecture introduced in Yue & Rind (2006) was ineffective when tested for receding obstacle scenarios whereas the model in Meng et al. (2009) detected obstacle motion direction in depth. However, the latter is not immune to signal noise, which could cause faulty collision alarms in case of a perturbed scenario. Hence, fusing the advantages of the two models, a modified model is achieved that can distinguish an approaching from a receding object while ignoring any noise in the input signal. Figure 2.4 illustrates Silva et al.'s modified LGMD model.

Cuadri et al. (2005), among others, introduced two complementary modules called the Attention Focusing Mechanism and Topological Feature Estimator to increase original model's versatility and efficiency, where the first module aims to optimise the use of computational power, by limiting the processing core to focus only on the frame zones that exhibit maximum activity at a given time step, and the second module aims to derive further information about the contemporary alarming status, particularly making a quick categorisation of the looming object. Similarly, modules such as 'Contrast Corrector' module for operation in darkness (night vision), a 'Proximity Detection' module to differentiate between a large faraway object and a small one close to the robot, and an 'Image Stabilisation' module to eliminate the vibration induced noise (airborne UAV), are to be introduced in this thesis to attain further system robustness. It can be noted that the models in Yue & Rind (2006) and Meng et al. (2009) which are integrated in Silva et al. (2014), were already modified models of the original LGMD architecture which implies that models based on insect vision are continuously evolving and yielding wider research focus.

2.6 Associated Research

Collision avoidance is an intricate and vital block within an autonomous system for which, eclectic hardware and software solutions are being tested and studied in order to develop a reliable and robust autonomous system capable of generating collision-free control commands independent of human supervision and control. The most relevant research in this field includes,

2.6.1 Advanced Unmanned Aerial Vehicles

Limited payload delivery of the smaller UAVs hinders the development of their autonomous control systems, though certain algorithms implemented on micro-sized processors deliver a reasonably efficient collision-free manoeuvring capability (Floreano et al. 2011), but the substantial hardware advancements are emerging in development of large UAVs (<20kg) bolstered by their massive payload capacity accommodating sophisticated sensors and computers to perform fully autonomous take-off and landing (Johnson et al. 2005, Saripalli et al. 2002), stabilisation and localisation (Kanade et al. 2004), collision avoidance (Scherer 2007), and aerobatic flight (Gavrilets 2002). Hence, the software developments for micro class of UAVs must be pushed further to compensate for hardware deficiencies.

2.6.2 Vision Based Techniques

Smaller category unmanned aircrafts deliver a very limited payload which in turn limits the researchers' available technological solutions for collision sensing and avoidance, as an example, Planar laser range finders (LIDAR) are much heavier than the available payload on a small category unmanned aircraft making it unfit to invest on. Other active range finders also, such as the Microsoft Kinect 2 are incapable of operating outdoors, limiting their applications to indoor navigations, hence the most feasible solution remaining imminent is, lightweight cameras and computer vision algorithms.

Monocular Vision

The most outstanding advantages of monocular vision technology specifically for UAV applications includes, its simplicity, reduced weight and cost, justifying the researchers' increasing focus on this technique. Michels et al. (2004) designed an obstacle avoidance learning algorithm for a remote controlled car navigating autonomously at $5ms^{-1}$ in a complex cluttered environment. Processing images at 7 Hz, the rate of fatal errors ($\sim 2\%$) were relatively satisfactory. However, Dey et al. (2015) presented a much advanced solution, demonstrating flights over 2km in cluttered wooded environments using a fast non-linear regression (machine learning approach) to perform monocular depth estimation in real-time. Nevertheless, all afore mentioned algorithms are computationally expensive (compatible only on small and medium sized UAVs) thus inadequate for micro aerial robotic applications.

Stereo Vision

There is an intense focus on stereo vision technology applied in UAV navigations. For obstacle detection, Byrne et al. (2006) illustrate the working of augmented stereo vision on embedded flight systems. To reduce false-positive detections from correspondence, they implement an image segmentation technique which executes at 10 Hz maximum, which is not a desirable capacity for a redundant collision avoidance system. Hrabar et al. (2005) introduced an additional technique in which, both optical flow and stereo vision were fused to deliver an exceptionally robust navigation performance of 100% success rate. Yang & Pollefeys (2003), among others, eliminated the need for per-frame rectification to increase computation efficiency and maximize frame rate, and also used a graphics processing unit (GPUs) to process their stereo vision algorithm with further precision and speed. Autonomous waypoint navigation and an obstacle detection using stereo vision integrated within a full suite of Inertial Measurement Unit (IMU) and vision-based state estimation is introduced in Meier et al. (2012). Nevertheless, Goldberg & Matthies (2011) present a methodology performing 46 Hz stereo vision processing that is much superior compared to the former method's 15 Hz processing speed. However, the most recent developments in stereo vision algorithms include Barry (2016) high-speed autonomous obstacle avoidance algorithm onboard a small fixed-wing UAV flying at $14ms^{-1}$ in a wooded cluttered environment, avoiding collisions with trees by processing captured stereo images at 120 Hz in real-time. Also in micro-sized category, Wagter (2014) introduced a 4 g ultra-light flapping UAV equipped with a 168 MHz embedded processor executing a line-based stereo algorithm at up to 40 Hz with an input resolution of 128×32 pixels. Among others, Honegger (2012), Honegger et al. (2014), Oleynikova et al. (2015) achieved successful processing of 376×240 pixel images at 127 Hz using optical flow matching on a small Field Programmable Gate Array (FPGA) facilitating real-time obstacle avoidance at 5 ms^{-1}.

Optic Flow

Optic flow is an exceptionally efficient vision-based solution to achieve stable flight, take-off, landing and performing successful collision avoidance (Barber et al. 2005, Beyeler et al. 2009*a*,*b*, Zufferey et al. 2008). As a result of their computational simplicity, it is possible to achieve reduction in size, power, weight and an astonishing processing speed of around 4500 Hz (40×30 pixels) using dedicated sensors. Although optic flow algorithms are even commercialised as a result of their efficiency, yet they remain futile as their high framerate processing comes at the cost of computation load which makes them less attractive when implemented on an extremely low power processors aboard micro UAVs.

Visual Simultaneous Localization and Mapping (VSLAM)

VSLAM has gained ground in robotic navigation over the past decade particularly in mapping-based navigation where the robot builds a map of its surrounding environment to perceive its situation and position subsequently manoeuvring safely in that environment (Davison 2007, Harris & Pike 1988, Sim et al. 2005), the VSLAM based algorithms are successfully tested on UAVs as well (Kim & Sukkarieh 2003), however, in these algorithms, the map built could be dubious despite exploiting loads of computational power. Although Lee et al. (2011) introduced a plane constraint in their algorithm to minimise the required processing power, this limited their navigation system to operate only in structured environments.

Klein & Murray (2007) introduced another variation of this technique called the Parallel Tracking and Mapping (PTAM). This technique also aims at reducing the computational load by authorising the mapping function to execute at lower framerate while the tracking runs at higher processing speeds, nevertheless, the algorithm fails to reduce computational load in case of a large map tracking.

Visual Odometry

Early applications of this technique include ground vehicle navigations at a reasonable framerate accompanied by a comparatively low error rate (Nistér et al. 2004). Visual odometry is particularly advantageous in GPS-denied environments where the vehicle's position and orientation must be tracked, Scaramuzza & Fraundorfer (2011) presented a tutorial on visual odometry summarising current scope of the field. Additionally, Forster et al. (2014) demonstrate the latest advancement of this technique called the Semi-direct Visual Odometry (SVO) where a downward facing monocular camera is used to map and track features in real-time, however these techniques are not yet validated on small high speed UAVs in real-world scenarios.

2.6.3 Stabilization and Localization

Stabilising flying robots using vision-based techniques (Roberts et al. 2003) is not a new problem statement in engineering since the techniques mentioned in previous sections such as optical flow, visual odometry, angular-rate-based technique (Srinivasan 2004), Image-IMU data fusion technique (Engel et al. 2012, Sa 2013), and bio-inspired methodology (Netter & Franceschini 2002) have continuously strived to address the UAV stabilisation and localisation challenges, evolving over the time. Following sections explain vision-based flight stabilisation further in detail.

Vision-aided Inertial Navigation Systems (VINS)

Feature's location estimates in a captured image is predicted by a single unit that integrates monocular images with the position estimates derived from Inertial Measurement Unit (IMU) sensor of the onboard Inertial Navigation System (INS) (Chowdhary et al. 2013, O'Sullivan et al. 2013, Weiss 2012). Recent developments in VINS include Shen et al. (2013) dual system featuring a fast-monocular VINS processing at 25 Hz and a slow-stereo VINS operating at 1 Hz to generate stable state-estimates for a slow moving quadrotor. In parallel, Li et al. (2013), Li & Mourikis (2013) introduced visual INS capable of tracking within 0.5 to 0.8% of the travelled distance for significant distances using an on-line calibration technique on a commodity cellphone.

2.6.4 Trajectory Planning

Optimum trajectory planning algorithms remain a major challenge in development of autonomous UAVs due to manoeuvring constraints, these algorithms must be simplified exhaustively to reduce further load on the onboard processor which must execute parallel computations in real-time. These challenges are addressed in Langelaan & Rock (2005), which introduces a fair trade-off between the computational power and performance by using a potential field method to simplify the system where obstacles are considered to be electrical charges imparting a force on the UAV, however, they are prone to errors such as, speculating dynamic obstacles (other UAVs) as points, ignoring contact geometries, system ceasing in local minimums relying on redundancy to rescue the flight.

Shooting and Collocation Methods

To determine an optimal path for the UAV given a cost function, an exhaustive search over the state and action space of the vehicle could be considered (Diehl et al. 2006), although this method is not suited for high-speed flight applications due to its computation costs, Diehl et al. (2006) do not perform a complete simulation, but instead simulate the system forward (usually a small amount) to find an optimal trajectory (Bock & Plitt 1984). Collocation methods on the other hand, force the optimiser to find both a feasible state and action set, simultaneously minimising a cost function (Stryk 1993).

Probabilistic Roadmaps (PRMs)

The trajectory generation in Probabilistic Roadmaps (PRMs) is achieved in two sessions, initially the off-line session generates paths that transit a substantial portion of the state space, after which the PRM commences the on-line operation by processing a quick optimisation in order to connect the initial state to its graph of trajectories, the on-line session further runs another simple optimisation to branch from the existing paths to the final state (Kavraki 1996) increasing the processing speed by connecting the states that are far apart using its existing highways. Nevertheless, this methodology is yet, computationally expensive and arduous to integrate if directed routes are not mandated (Svestka & Overmars 1997).

Rapidly-Exploring Random Trees (RRTs)

LaValle (1998) introduced a solution for a broad class of path planning problems, known as the RRTs, specifically designed to handle nonholonomic constraints (including dynamics) and high degrees of freedom. The RRTs randomly explore state space through small extended operations that add to an existing tree. Further, this technique has experienced extensive advancements, as in Karaman & Frazzoli (2010) RRT* methodology, that derives asymptotic optimality by rewiring an RRTs' internal structure.

Trajectory Libraries

This methodology helps to generate a sequence of pre-built library of small trajectories to form a consistent protracted path. Each pre-built library may contain small trajectories of a certain flight direction which are chosen individually at each time-step. Atkeson (1993) introduced a trajectory planning technique to reduce the computational load of dynamic programming, by using second order local trajectory optimisation, which was later used in a learning context for planning with humanoids (Atkeson & Morimoto 2003), but when the technique was implemented on marble-maze navigation, the implementation challenges highlighted were, trajectory generation, choice of distance function, and dependency on model performance (Stolle & Atkeson 2006).

Frazzoli et al. (2000) among others, demonstrate a computationally efficient and highly stable control system based on trajectory libraries, in which, the planning is done for manoeuvres and trim trajectories individually where motion plans are the former and stable regions of state space are the latter trajectories. Similarly, Bachrach uses bundles of trajectories to map the environment in addition to using them for control system (Bachrach 2013). However, recent developments of this technique aim to solve the challenging task of safe transitioning between trajectories in libraries which can be achieved by ensuring that the next controller is capable of stabilising the ending state of the previous controller (Majumdar & Tedrake 2012), developing on this concept, Majumdar & Tedrake (2016) recently presented a fail-safe obstacle avoidance system.

Differentially Flat Systems

In a differentially flat system, it is possible to find a set of flat outputs from which the full state and input vectors can be determined without integration (LaValle 2006, Sira-Ramirez & Agrawal 2004), this technique helps to reduce the computational load of trajectory planning algorithms.

Mathematically, given a system of the form, $\dot{x} = f(x, u)$ with $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$, one can find flat outputs $y = h(x, u, \dot{u}, \ddot{u}, ..., u^{(k)})$ such that the following g and g' exist as: $x = g(y, \dot{y}, \ddot{y}, ..., y^{(j)})$ and $u = g'(y, \dot{y}, \ddot{y}, ..., y^{(j)})$.

Prominent applications of this methodology includes Alturbeh & Whidborne (2014), Cowling et al. (2001, 2007), Mellinger & Kumar (2011) introducing findings on minimum snap trajectories where the positions x, y, z and the yaw angle ψ are considered as the flat outputs, further disabling the control over pitch and roll, that in turn makes the knife-edge manoeuvres (cluttered environment flight) almost impossible without switching between controllers. Hence, not recommended for underactuated plants such as quadrotors since they require all the state variables to be specified within the system.

2.6.5 System Identification

To implement a powerful feedback control technique, system dynamics identification is necessary, which is generally performed in three parts, namely, model identification, followed by determination of flat plate dynamics, and parameter selection. For the first task, the identification model presented in Sobolic (2009) is reliable, but for the second task, flat plate dynamics from Cory & Tedrake (2008), Tedrake (2009) could be implemented, and for the final task of parameter selection, there exists a plethora of techniques discussed in Tobenkin (2014), and as an example the grey-box technique introduced in Ljung (2007).

2.6.6 Feedback Control Algorithms

No flight controller system is complete without a robust feedback control algorithm, since external disturbance, modelling errors, wind gusts etc., cause plant's deviation from the planned trajectory irrespective of how optimum it might be. To explore the full potentials of our plant, real-time feedback controls are necessary onboard. There exists a vast literature on the feedback control algorithms to be considered while designing a non-linear system, explained in the following sections.

Linear Quadratic Regulators

LQR is one of the eminent feedback control techniques to deliver optimal controls with a linear dynamic model and quadratic cost (Lewis & Syrmos 1995). Many of the control tasks mentioned in previous sections (Abbeel et al. 2006, Barry 2012, 2014, Cory 2010, Ritz 2012), either implemented directly or variations of LQR, with Time Varying Linear Quadratic Regulator (TVLQR) being one such variation, it involves taking Taylor approximations along an open-loop trajectory to generate a linear time-varying dynamics model along the trajectory (Tedrake 2009), however, optimal control can be applied, even on non-linear systems about known trajectories using the same Taylor approximations.

Model Predictive Control

Model Predictive Control (MPC) make direct use of an explicit and separately identifiable model. Robustness (Bemporad & Morari 1999) and autonomy (Garcia et al. 1989) are the salient features of these controllers since they can yield high performance, independent of expert intervention for long periods of time. To generate optimal input for the given system with a reference trajectory, an optimisation program (model class and robustness requirements define the type of this program) must be executed. After the optimisation program the system's response is observed for the first input values, the simulation and optimisation process are repeated at time $t + 1$ (Garcia et al. 1989). Another major advantage of MPCs is their agility which is applicable for linear form of MPCs (Wang & Boyd 2008). Nevertheless, as suggested in Singh & Fuller (2001), linearisation of the non-linear MPCs could simplify these systems as well.

Robust Control Methods

One of the robust control methods based on system uncertainty is the H-infinity method that characterises worst-case scenario using H_∞ norm for a system whose uncertainty is given as, $||G(s)||_\infty = \max_\omega \bar{\sigma}(G(j\omega))$, where $\bar{\sigma}$ denotes the largest singular value of the frequency response and $G(j\omega)$ describes the system's frequency response (Bates & Postlethwaite 2002). To build a linear control system that is stable under bounded uncertainty, the system's most vulnerable portions of the maximum frequency response must be characterised on the largest singular value, and if the system is found to be stable to a bounded uncertainty, then it can be concluded that the entire system shall possess fairly stable frequency response to uncertainty.

Gain Scheduling

Gain scheduling is a process of creating a hybrid of multiple controllers (same or different) with specific desired characteristics fused to form a single versatile system. In such a hybrid system, the 'scheduling variable' decides the gain values for each controller within the system, hence a slow moving scheduling variable is preferred (Shamma & Cloutier 1993). Depending on the individual characteristics and complexity of the fused controllers, a system with significant flexibility, robustness and a rich feedback control can be achieved.

2.6.7 Flight in Cluttered Environment

Flying small UAVs in a complex cluttered environment is not yet possible, an early work on autonomous flight through forests was introduced in Langelaan & Rock (2005), which demonstrates experiments on a ground vehicle in an artificial forest since their vision-based SLAM algorithm (without range estimates) consumed significant computation power to build maps, eventually making it infeasible for flying robot applications (Langelaan 2007, 2006). Further advanced research such as Roberts (2012) vision-based range estimation overcomes the earlier shortcomings by using range estimates to predict tree distances focusing computation on the closest trees, however, their algorithm is limited to tractable forests since they consider trees as perfectly vertically oriented objects. More recent work such as Dey et al. (2015), present flight data for their vision-based algorithm that uses the captured monocular features such as Budgeted near-optimal feature selection and fast non-linear regression to extract depth cues.

2.6.8 Neuromorphic Engineering

The neuro-biological architecture of various living creatures has been studied over the past decades to find bioinspired solutions to the prevailing computation problems, one such example is the study of chemotaxis in ants (Webb 1996), demonstrated by a Lego robot fitted with a pair of gas sensors and an elementary processor performing an ant-like trailing behaviour with the help of a small network of neurons. However, the architecture implemented by Webb was not based on identified neurons as in LGMD models. Other bioinspired robots, have also implemented abstract models of neural architectures, for instance, Lambrinos et al. (1997) introduce a polarised light compass possessed by a desert ant called *Cataglyphis*. The compass has been implemented on a ground robot (Sahabot) where three light polariser sensors inspired by the polarisation-opponent interneurons of a cricket's optic lobe provide the input to the compass. However, due to ambiguity of the neural network, consecutive computations were carried out mathematically. On the other hand, fly motion detection abilities have been implicitly applied in robotic navigations based on Reichardt & Egelhaaf (1988) correlation-type elementary motion detector that is inspired by fly optic lobe, cloned using neuromorphic engineering techniques. Furthermore, the control algorithms based on optic flow methods also manoeuvre robots through complex environments successfully. However, these techniques succumb to the challenging requirements of this research such as, knife-edge manoeuvres, complex backgrounds, underactuation, irregular contrast, and limited payload delivery.

2.7 Summary

It can be concluded that the proposed QLDP based on LGMD luminance-difference computing feature to detect and avoid static obstacles during a high speed flight through complex GPS-denied environments is one of the highly apt solutions available for Micro/Nano class of robots due to its computational simplicity yet robust brisk manoeuvre generation satisfying our research objectives. This chapter provides evidences to bolster the preference of implementing a biologically inspired vision-based collision avoidance system on a flying robot to achieve a fail-safe mission accomplishment. Several salient accomplishments of the proposed system briefly includes,

- Operation in GPS-denied environments
- Obstacle direction detection using motion cues
- Distant obstacle detection using proximity estimator
- Noise/blur immunity and high vibration resistance
- Compact, economical (no active sensors) and low power rating
- Computational simplicity satisfying micro robotic constraints
- Indigenous reactive controller facilitating brisk knife-edge manoeuvres
- High frequency processing (120Hz) facilitating high-speed flights

3

Research Platform

> *"The human condition is not perfect. We are not perfect specimens, any of us. We're not robots."*
> – Michael Ovitz

A comprehensive research is always performed on a reliable platform and validated against a set of well established metrics. To accomplish this goal, this chapter presents a test plant that is standardised regardless of the type of collision avoidance model implemented. This system features a genuinely fabricated ground robot with dynamics similar to that of a four-wheeled vehicle capable of validating our algorithm in a near-exact real-world application. Prior to platform implementation, the collision avoidance model is calibrated through simulations and off-line testing to minimise errors before risking damage to hardware components. The robot is also capable of storing the data collected from collision tests such that further analysis and modulations could be performed on the proposed algorithm. To conclude, the platform designed satisfies the following requirements:

1. Standardisation of the controls and dynamics involved

2. Simple setup and calibration of onboard collision avoidance algorithm

3. Perform laboratory tests and simulations

4. Conduct field tests for real-world scenarios

5. Collect and store test data, system response, and performance

6. Autonomous waypoint assignment and trajectory generation capability

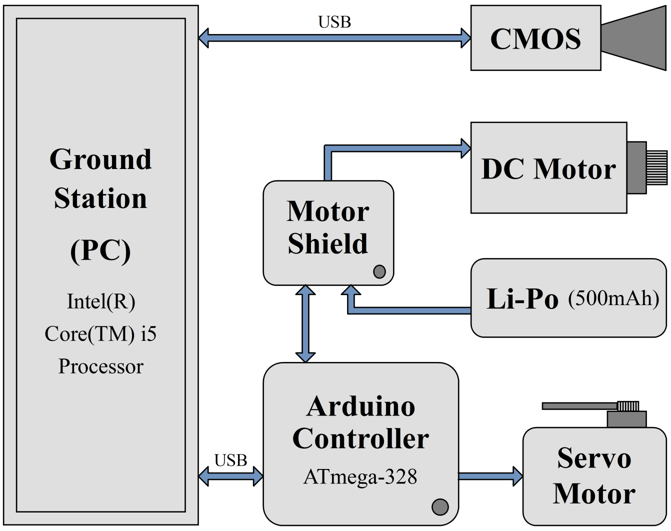


Figure 3.1: Ground Testing Platform Architecture

Figure 3.1 illustrates the testing platform circuit architecture which comprises of mainly two parts being, the ground station (PC running the proposed algorithm in MATLAB) and the robot communicating through a universal serial bus connection. A brief description of the proposed experimental setup involves a CMOS sensor capturing digital images of robot's forward field of view which are transmitted to the ground station for processing in real-time and thus generation of avoidance motor control commands. Further, the generated control commands are transmitted to the microcontroller development board onboard the robot to drive the motor shield and a servo motor steering and manoeuvring the robot safely through obstacles. However, this thesis also proposes an aerial testing platform architecture shown in Figure 3.2 for UAV applications for future development of this research. The architecture involves an advanced UAV control system with two layers of processors (high and low level) thus implementing the proposed collision avoidance algorithm as an auxiliary fail-safe system onboard this robot. However, micro and nano UAVs with elementary computation power being the primary application target of this thesis may exploit this system to acquire collision-free autonomous manoeuvring capabilities efficiently. To setup the proposed platform for model implementation and performance validation, certain fundamental hardware and software components are required, which are ex-

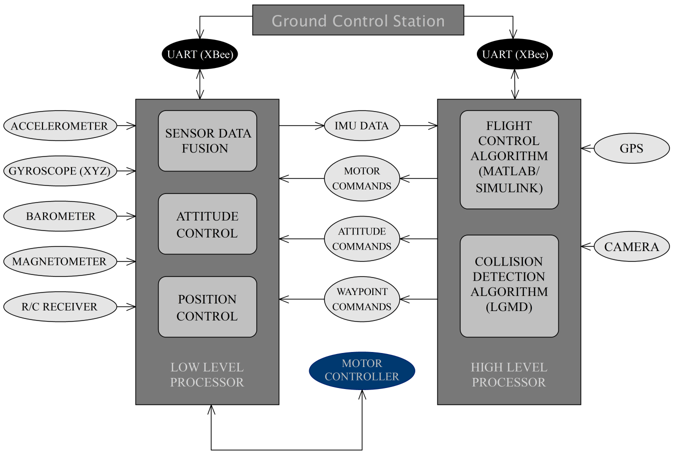


Figure 3.2: Aerial Testing Platform Architecture

plained in the following sections.

3.1 Tools and Components

According to standard research practice, validation of any proposed algorithm must be conducted practically to record, assess, and compare its performance for real-world scenarios involving several hardware and software systems operating in real-time, namely;

- Ground Control Station
- Robotic Platform

3.1.1 Ground Control Station

The most important hardware consideration is the Ground Control Station (GCS) which executes the processing and supervises the core of operation; It must simultaneously visualise and record robot operation data, while generating trajectory autonomously. This section presents the hardware and its role involved in the GCS. For a GCS, the most significant component is the field computer which is essentially the core of the GCS responsible for communicating, guiding and navigating robots according to visual inputs it receives. Therefore, a lightweight, transportable, yet high specification mobile PC (laptop) is preferable for this purpose.

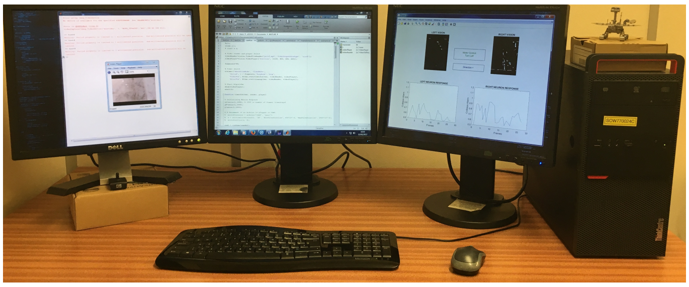


Figure 3.3: Ground Control Station (PC)

In this thesis, a normal PC with an Intel Core i5-6500 processor and 8 GB of RAM shown in Figure 3.3 is used. The test platform is a micro ground robot capable of operating indoors. The GCS can operate on a Windows, MAC, or a Linux operating system to execute MathWorks MATLAB software, however, Linux is preferred to accommodate Robot Operating System (ROS). Here, MATLAB is used as the platform to develop the QLDP due to, (a) user-friendliness and ease of use, (b) wider applications, (c) easier debugging capability, and (d) no memory management requirements. Further, the GCS computer requires XBee telemetry module (recommended) to establish wireless communication with the robots in real-time. The XBee modules are lightweight

Arduino-based radio communication boards featuring point-to-point, point-to-multipoint, and multipoint-to-multipoint communication facilitating control over multiple robots from a single GCS. Ultimately, the GCS computer must be loaded with the test-field map data for visualisation and way-point assignment purposes. Nevertheless, in this thesis universal serial bus connection is used and the implementation of wireless communication is suggested for future work.

3.1.2 Ground Robotic Platform

The proposed model is implemented on a 3-DOF ground robot designed and fabricated at the Unmanned Autonomous Systems Laboratory (UASL), Cranfield University, to meet the specific requirements of the experimentation discussed in Chapter 6. It exhibits the necessary agility to accomplish successful collision avoidance using a DC motor, 9g servo motor controlling steering, Arduino Nano development board interfacing software and hardware components, motor shield, and a 2-Megapixel CMOS sensor capturing input data.

The Atmel ATmega328 8-bit AVR micro-controller with a maximum of 20 MHz operating frequency built into an Arduino development board is implemented to interface the robot with the proposed algorithm developed in MATLAB. Image frames captured by the CMOS sensor are transmitted through a USB cable to the ground station where the images are processed and motor control commands generated. These commands are transmitted through the micro-controller to the servo and motor shield to control the robot's locomotion. In this platform, the conventional robotic differential steering was substituted with an Ackermann steering to replicate a near-exact real-world four wheel vehicle dynamics. This poses greater manoeuvring challenges as a result of underactuation. The schematic illustration of the assembled robot is shown in Figure 3.4.

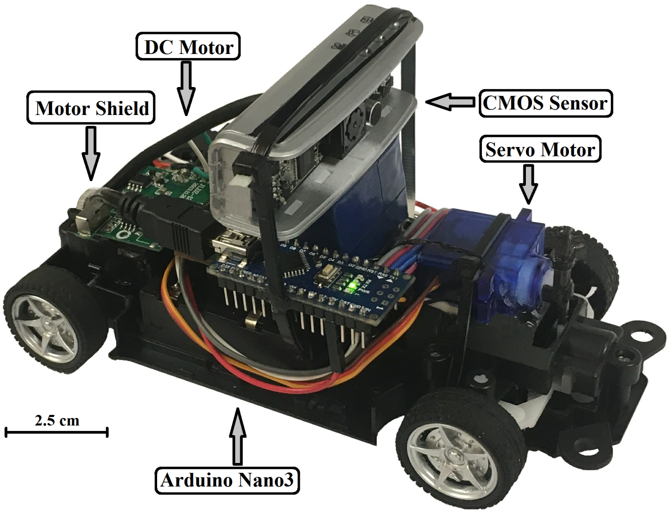


Figure 3.4: Ground Robot (with Ackermann Steering)

In this thesis, a standard motor control algorithm is designed to meet the requirements of every individual test explained in Chapter 6, which is fused with the proposed QLDP facilitating stable autonomous navigation through obstacles to reach the assigned waypoints. Large input data acquisition frequencies (∼120Hz) necessitates not only a fast processing algorithm, but also an efficient, precise, and an extremely agile control generation block within the collision avoidance algorithm since even a minute error or delay may cause a catastrophic instability and collision. Complete technical specifications of the test platform proposed to be assembled for the experimentation are listed in Table 3.1.

Table 3.1: Platform Technical Specification

Dimensions	12x5x4.5cm
Maximum Total Weight	150g
Battery	500mAh LiPo
Motors	2x12W
Telemetry System (recommended)	XBee 2.4GHz
Input Data Capture	2-Megapixel HP CMOS Sensor
Field of View (FOV)	90°
Data Acquisition Frequency	30Hz
Vehicle linear velocity	0.2-2.0 ms^{-1}
Input Data Resolution	2D RGB 320x240 Pixels
Central Processing Unit	Intel(R) Core(TM) i5-6500
CPU Clock frequency	3.20 GHz
Random Access Memory	8.00 GB
Operating System	MS Windows 7 (64-bit)

For testing and system validation purposes, smaller platforms are preferred to minimize the time to test new algorithms since the required material and test-epochs are reduced drastically. Further, experiments can be conducted indoors, or even on a workbench without the need for any legal testing authorisations. The microcontroller used on the robotic platform generates signal pulses with a simple servo write command that essentially interprets the algorithm's response (spiking alarm) in terms of servo motor's shaft position controlling the Ackermann steering. Shaft position ranges from 0° to 18°, where the former is extreme right steering and the latter, extreme left. Dynamics involved in control and manoeuvring of a ground robot are fairly simple due to fewer degrees of freedom and governing equations of motion. However, aerial robots exhibit much complex dynamics and control discussed in the following section.

3.1.3 Aerial Robot Dynamics

Since the application of the collision avoidance algorithm proposed in this thesis primarily extends to aerial robots, therefore for future work purposes, it is necessary to briefly discuss the dynamics and control of a quadrotor that can be considered as a robot capable of quasi-stationary or hover flight similar to conventional helicopters. This platform consists of four rotors configured in a symmetrical cross pattern about its centre. Each rotor consists of a propeller blade directly attached to a motor located at the extremity of each arm. The platform centre houses the avionics and payload including telemetry links, autopilot and cameras.

The rotors are arranged in counter-rotating pairs such that the front and rear rotate counter-clockwise and the left and right rotate clockwise. The propellers have a fixed pitch and their axes of rotation are parallel to each other and the vertical axis. A vertical thrust and a horizontal drag acting through the centre of each rotor is thus produced at each arm. As a result, a torque is also produced about each axis from the platform centre. The net force and torque on the quadrotor results in translational and rotational movement. Therefore, it can be concluded that the motors are the only platform actuators simplifying dynamics to a fair extent. At a low-level, the control inputs required are variations in motor speed Ω_n for $n = 1, 2, 3, 4$ rotors. The thrust force F_n from each rotor depends on the blade configuration and physical parameters (k), and is directly proportional to the square of the motor speed such that,

$$F_n = k\Omega_n^2 \tag{3.1}$$

The torque τ_n produced at each motor depends, in part, on the propeller drag force $\bar{F_n}$ and propeller radius r_p, but can be simplified as

$$\tau_n = r_p\bar{F_n} = k_\tau\Omega_n^2 \tag{3.2}$$

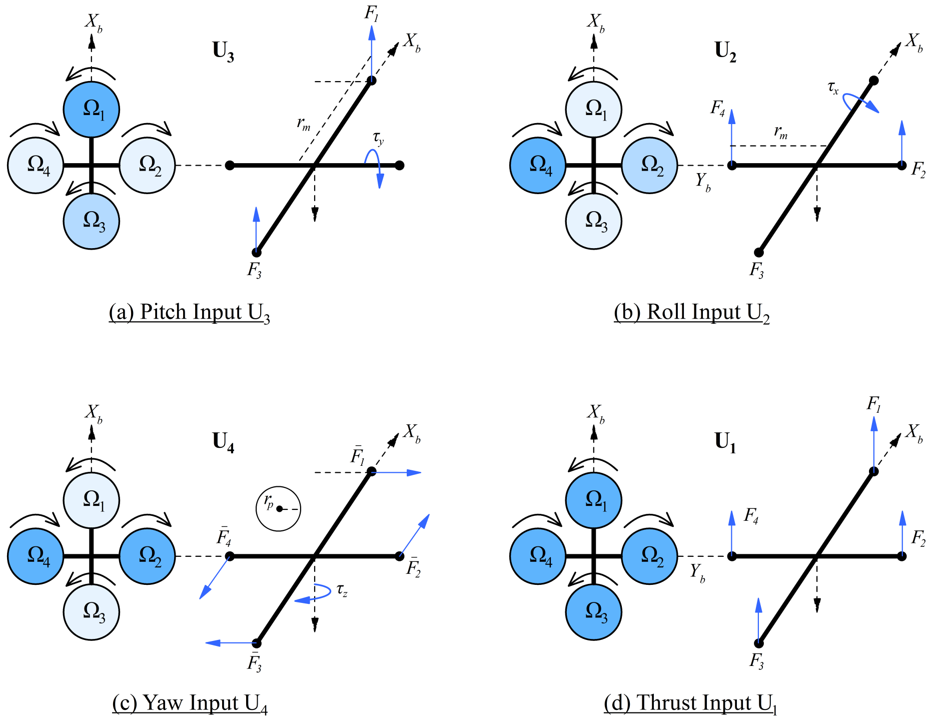
At a higher level, a mixture of rotor speed variations can be used to construct four separate control inputs U_i for $i = 1, 2, 3, 4$. The input controls consist of a force U_1 and three input torques U_2, U_3 and U_4. Although the platform is still under-actuated, the controls provide a more direct correspondence to changes in roll, pitch, yaw and vertical acceleration which is more useful for controlling position and attitude. A set of simplified diagrams depicting the various forces and torques acting on the platform resulting from changes in control input is given in Figure 3.5. The rotors are labelled 1-4 starting at the front rotor and moving clockwise through to the left rotor.

If all motor speeds are identical such that $\Omega_1 = \Omega_2 = \Omega_3 = \Omega_4$, a net upward force F_T

is produced. The net torque $\tau_c = 0$ as the effects from each rotor cancel due to the counter rotating arrangement. If the speeds are decreased or increased by the same amount and the pitch and roll angles are zero, the platform will descend, hover or ascend depending on the relative magnitude of F_T compared to the platform weight mg. For all other attitudes, a lateral and longitudinal force will also be induced causing movement in the horizontal plane. Setting $U_1 = F_T$ ensures the magnitude of the thrust can be directly controlled such that,

$$U_1 = F_1 + F_2 + F_3 + F_4 \tag{3.3}$$

$$U_1 = k(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \tag{3.4}$$



(a) Pitch Input U_3

(b) Roll Input U_2

(c) Yaw Input U_4

(d) Thrust Input U_1

Figure 3.5: Quadrotor Control Inputs (from Mcfadyen (2015))

The orientation of the thrust vector is controlled by changing the platform attitude by inducing a non-zero torque. Increasing the right motor speed and decreasing the left motor speed by the same amount $\Delta\Omega$ results in a positive torque τ_x about the lateral axis whilst retaining the overall thrust such that $F_T = F_1 + (F_2 + \Delta F) + F_3 + (F_4 - \Delta F)$. An acceleration in roll angle is thus induced. Setting $U_2 = \tau_x$

$$U_2 = r_m(F_4 - F_2) \tag{3.5}$$

$$U_2 = r_m k(\Omega_4^2 - \Omega_2^2) \tag{3.6}$$

Increasing the front motor speed and decreasing the rear motor speed by the same amount $\Delta\Omega$ results in a positive torque τ_y about the longitudinal axis whilst retaining the overall thrust such that $F_T = (F_1 + \Delta F) + F_2 + (F_3 - \Delta F) + F_4$. An acceleration in pitch angle is thus induced. Setting $U_3 = \tau_y$

$$U_3 = r_m(F_3 - F_1) \tag{3.7}$$

$$U_3 = r_m k(\Omega_3^2 - \Omega_1^2) \tag{3.8}$$

The torque produced by each motor is required to maintain the propellers spin and provide the necessary thrust force. It also induces a drag force $\bar{F}_n$ perpendicular to the associated thrust force such that each rotor contributes some torque about the body vertical axis. Increasing the right and left motor speed and decreasing the front and rear motor speed by the same amount $\Delta\Omega$ results in a positive torque τ_z about the vertical axis whilst retaining the overall thrust such that $F_T = F_1 + (F_2 + 2\Delta F) + F_3 + (F_4 - 2\Delta F)$. An acceleration in yaw angle is thus induced. Setting $U_4 = \tau_z$

$$U_4 = r_p(\bar{F}_2 + \bar{F}_4 - \bar{F}_1 - \bar{F}_3) \tag{3.9}$$

$$U_4 = k_\tau(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \tag{3.10}$$

The equations describing the control inputs can then represented in matrix form as

$$\begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{pmatrix} = \begin{pmatrix} k & k & k & k \\ 0 & -r_m k & 0 & r_m k \\ -r_m k & 0 & r_m k & 0 \\ -k_\tau & k_\tau & -k_\tau & k_\tau \end{pmatrix} \begin{pmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{pmatrix} \tag{3.11}$$

The model for the quadrotor control inputs derived above has been simplified. The simplifications include the following well-founded assumptions:

- The effects of blade flapping resulting from deformation of the propeller moving at high velocity are neglected.

- Each rotor is identical. The associated drag force on each propeller is assumed to act at the tip of the blades to produce the rotor torque. The associated propeller velocity is such that an approximately constant thrust is induced.

- The platform is symmetrical about the zx and zy planes.
- The principles of conservation of energy apply to each rotor such that all input energy is transferred to a thrust and drag force.

3.2 Summary

This chapter aims at introducing a research test platform that is implemented in Chapter 6 to validate the asserted capabilities of the proposed collision avoidance algorithm. Every required software and hardware components of the test platform are described briefly along with the dynamics and control theory involved.

Last section of the chapter provides a general description of an aerial robot (quadrotor) control principle in order to review an overall understanding of such robotic control inputs which can be embedded in the proposed algorithm to generate apt flight control commands as suggested in future work Chapter 7.

4

Biomimetic Vision Architecture

> *"I see insect level behaviour as a noble goal for artificial intelligence practitioners. I believe it is closer to the ultimate right track than are the higher level goals now being pursued."*
>
> – Rodney A. Brooks

4.1 Introduction

It has been over four centuries since the fascinating structure of a compound eye was unveiled using the earliest microscopes, captivating the concern of scientists from different fields of study across the world. Today an extensive range of research is conducted on insect vision, stretching from the study of genetics using the visual mutants of *Drosophila* through electrophysiological investigation of the properties of many visual neurons to the modelling and simulation of visual circuits and functions.

This chapter describes the inspiring biological structure and principles of flying insect vision with a focus on Lobula Giant Movement Detector (LGMD), a wide-field visual neuron located in the Lobula layer of the *locust's* nervous system which inspires the QLDP model proposed in this thesis. Salient developments in LGMD neural network aggregated over the years along with the original model pioneered by Rind & Bramwell (1996), are also presented in the following sections. Electrophysiological decoding of the LGMD was accomplished with the ultimate aim of designing a simple computational model performing the near-exact biological behaviour of a *locust* visual neuron to obtain a set of processing layers sequenced to deliver an efficient yet robust obstacle detection algorithm. An overview of this chapter can be summarised as follows,

- Anatomy and biological behaviour (operation-principles) of invertebrates (flying insects) visual neuron.
- Detailed study on elements of a *locust* visual neuron, focusing on the photoreceptor cells of the compound eye, processing in the lamina region of the optic lobe, and spiking responses of the LGMD neuron.
- Techniques used to design neural architectures, including single cell models, large neural networks and neuromorphic engineering.
- Computational modelling of the LGMD neurobiological model using published electrophysiological study reports.

4.2 Locust Visual Neuron Anatomy

Insect visual systems in general, including several species of *locust* (notably *Locusta migratoria and Schistocerca gregaria*), have been studied extensively and many similarities in anatomy and neuronal responses have been found. This section depicted from Blanchard (1998), focuses mainly on the visual processing performed by an insect movement detection system that is the *locust* LGMD. The *locust* visual system comprises two compound eyes, one located on each side of the head, with three neuropiles (Lamina, Medulla and the Lobula) behind each eye. Known collectively as the optic lobe, these neuropiles process the visual information as it passes from the photoreceptors to the brain (the protocerebrum). Figure 4.1 shows the structure of the optic lobe behind one eye. In Locusta,

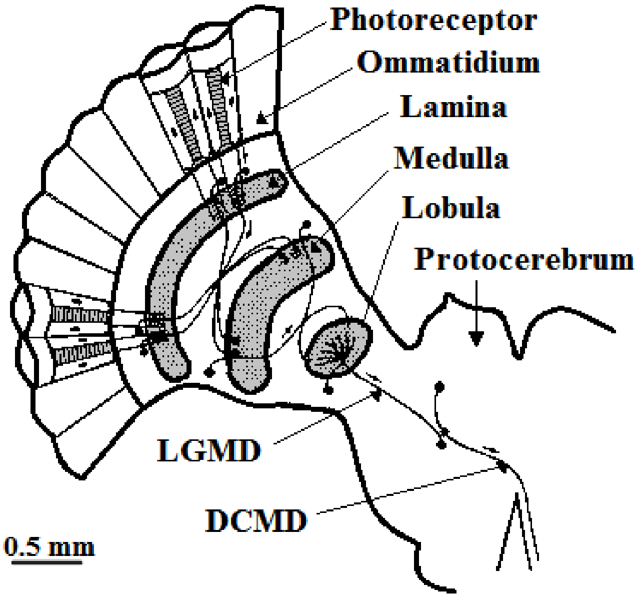


Figure 4.1: Locust Visual System. (from Rind & Simmons (1999))

each compound eye constitutes an estimated 8500 ommatidia (Shaw 1978) packed into a hexagonal array measuring approximately 3x2 mm (Wilson et al. 1978). The eyes use simple apposition optics where each ommatidium samples a discrete point in space (Nilsson 1989). The spatial resolution (acuity) is determined by both the angular separation of the ommatidia and their receptive field (acceptance angle). The angular separation is not uniform: there is an acute zone at the front of the eyes looking ahead of the animal (Horridge 1978) while in a region at the dorsal rim of the eyes the ommatidial lenses are smaller and fused (Eggers & Gewecke 1993). Structural changes within the ommatidia (Tunstall & Horridge 1967, Horridge et al. 1981) increase the acceptance angle as the ambient light intensity falls (Wilson 1975). The increase in acceptance angle allows light from a wider area to be captured by the ommatidium, improving sensitivity at low light levels (Williams 1983). Within each ommatidium there are eight photoreceptor (retinula) cells which combine to form a fused rhabdom with their microvilli (Wilson et al. 1978). Six of the photoreceptors (R1-6, also known as short visual fibres (SVFs) because their synaptic terminals are in the lamina) contribute to the rhabdom along the full length of the ommatidium while the remaining two photoreceptors (R7-8, or long visual fibres (LVFs) which have synaptic terminals in the medulla), once thought to be second order cells (Scholes 1965), contribute a small amount in the proximal third of the rhabdom. The rhabdom is the photosensitive structure within the ommatidium, with rhodopsin molecules embedded in the microvillar membranes of the photoreceptors (Williams 1983). At the proximal end of an ommatidium the photoreceptor axons form into a bundle which passes through the basement membrane and projects to the first optic neuropile, the lamina (Meinertzhagen 1976). The neurons in the lamina are grouped into cartridges and there is a precise retinotopic mapping from the ommatidia to these cartridges which preserves the spatial information of the visual image.

There are many anatomical similarities between the neurons identified in the *locust* lamina and those in other insects (Nowel & Shelton 1981, James & Osorio 1996, Strausfeld 1976, Shaw 1984). The terminals of the six SVFs are found in the cartridge: the axons of the LVFs pass through the lamina and project to the medulla. There are six monopolar cells, two of which, M1 and M2, have thickened axons and dendrites confined within the cartridge. These correspond to the large monopolar cells (LMCs) found in flies which receive extensive synaptic input from the six SVF terminals (Nicol & Meinertzhagen 1982). Twelve cell types have been observed in flies (Strausfeld & Campos-Ortega 1977), including the monopolar cells, amacrine cells whose dendrites project across many cartridges and efferent cells from the medulla: it is reasonable to assume that similar cells are present in the *locust*. A number of glial cell processes are found around the cartridges which electrically isolate neighbouring cartridges (Shaw 1984). In addition the *locust* lamina is isolated from the eye and the optic lobe by glial cells and trachea which fill the extracellular space, forming diffusion barriers (Shaw 1978). As with the lamina the second optic neuropile, the medulla, has a retinotopic arrangement of neuronal cartridges. However the visual information is reversed horizontally by the projections from the lamina which cross at the first optic chiasma.

The anatomy of medulla cartridge in the *locust* is not clearly identified, but the estimates suggest that each cartridge contains at least 40 distinct cell types (Osorio 1992) and a few of these cell types have been identified (O'Carroll et al. 1992, James & Osorio 1996). The anatomy of the fly medulla has been studied in greater detail and many more cell types have been found and identified (Strausfeld 1976). The third optic neuropile, the lobula, has a coarse retinotopic arrangement with the positions of the cartridges reverting to their original positions after the second optic chiasma. However, many of the neurons found in the lobula have dendritic trees which cover large areas of the neuropile and hence the visual field. These wide-field neurons can be identified reliably in different individuals from their anatomy. A variety of neurons of this type has been identified in the *locust* (Rind 1987) including the LGMD (O'Shea & Williams 1974), the neuron at the heart of our collision avoidance algorithm. In addition to the flow of information down the optic lobe from the eye to the brain there are neurons which project in the opposite direction. All of the optic neuropiles receive inputs from the brain and the contralateral eye via efferent neurons with large axonal arborizations which cover large areas of the visual field (Gewecke & Hou 1993). Other neurons project centrifugally between regions of the optic lobe (Strausfeld 1976, Shaw 1984).

4.3 Lobula Giant Movement Detector

The LGMD neuron of the *locust* optic lobe has been studied for many years and its responses and functions have been the subject of controversy constantly. The LGMD is a unique, identified, wide-field neuron in the lobula neuropile behind each eye of a *locust*. The anatomy of the LGMD is shown in Figure 4.2. It has three dendritic subfields, a large dendritic fan which extends across the full width of the lobula and two smaller subfields protruding from the base of the

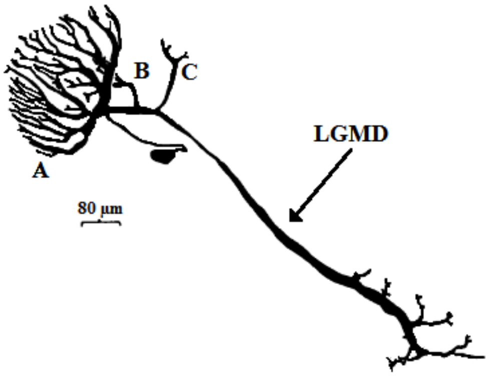


Figure 4.2: LGMD Anatomy (from O'Shea & Williams (1974))

fan (O'Shea & Williams 1974). Its axon projects to the brain where synapses are made with the DCMD neuron (O'Shea et al. 1974). In turn the axon of the DCMD projects down the contralateral nerve cord to form synapses with interneurons and motorneurons in the thoracic ganglia (Simmons 1980).

Initially the DCMD was studied owing to the ease with which recordings could be made with extracellular electrodes from the nerve cord, where spikes from the DCMD are easily identified due to their large amplitude (Rowell 1971). A later study using intracellular recordings established a one-to-one correspondence between spikes in the DCMD and the LGMD (O'Shea et al. 1974), which led to the proposal that the synapse between the two cells was electrical (O'Shea & Rowell 1975). This was disproved by a detailed examination of the properties of the synapse which revealed a short transmission delay and a slight gain, both of which are characteristics of a chemical synapse (Rind 1984). The one-to-one correspondence between spikes in these neurons, which persists up to frequencies of 400Hz (Rind 1984), allows the responses of the LGMD to be inferred from those of the DCMD.

An early experiment revealed responses in the nerve cord, probably from the DCMD, to increases and decreases of light intensity (ON and OFF) and to movement of illuminated objects (Burtt & Catton 1952). Subsequent experiments using striped stimuli caused a controversy by hinting that the acuity of the visual system was significantly better than expected from the optics of the eye, but these results were later attributed to artefacts in the stimuli (Horridge 1975). Experiments where small stimuli moved horizontally or vertically produced a short burst of spikes in the LGMD/DCMD. This response habituated if the movement was repeated and showed that the receptive fields of the cells covered the whole field-of-view of the eye. The idea that the neurons were tuned to detect novel movements of small objects was proposed but, despite the suggestion that they may play a role in initiating escape behaviour via the thoracic ganglia, the value to the animal of this non-directional movement detection was questioned (Rowell 1971).

Nevertheless due to a lack of functional reference, a detailed study was undertaken to determine the neuronal circuit underlying this response. The responses to changes in the intensity of a small light source were attributed to the proper ties of the photoreceptor cells (Rowell & O'Shea 1976*b*) but a lateral inhibitory network was proposed as the mechanism responsible for the dependence of these responses on the intensity of the surrounding area (Rowell & O'Shea 1976*a*). Input to the dendritic fan was found to be from transient ON/OFF cells and these cells or their synapses were predicted to be the site of LGMD habituation (Rowell & O'Shea 1976*a*). Finally, experiments were conducted with stimuli comprising small objects and large-field moving stripes, and these showed that the response to a small object was reduced by movement of a stripe pattern behind the object and large stimuli produced inhibitory responses (Rowell et al. 1977).

Taken as a whole, these findings led to the proposal of the model shown in Figure 4.3 for the input circuit of the LGMD. In this circuit lateral inhibition between the inputs to the dendritic fan, believed to involve amacrine cells in the medulla region of the optic lobe (Rowell et al. 1977), prevents habituation during movement of the whole visual field (O'Shea & Rowell 1975). Feed-forward inhibition from separate populations of ON and OFF sensitive cells projecting onto the smaller dendritic sub-fields suppresses responses to large stimuli and changes of the whole visual image generated by movements of the animal (Zaretsky & Rowell 1979). Neurons thought to be responsible for the OFF feed forward connection were identified anatomically in the medulla and the connection was broken by lesioning the dorsal uncrossed bundle of axons which projects from the medulla to the lobula (Rowell et al. 1977). Neurons with properties matching the other proposed cell types have since been found in the medulla (Osorio 1987, O'Carroll et al. 1992, James & Osorio 1996), although recent findings using electron microscopy have unveiled a novel synaptic architecture within the dendrites of the LGMD which may mediate lateral inhibition directly.

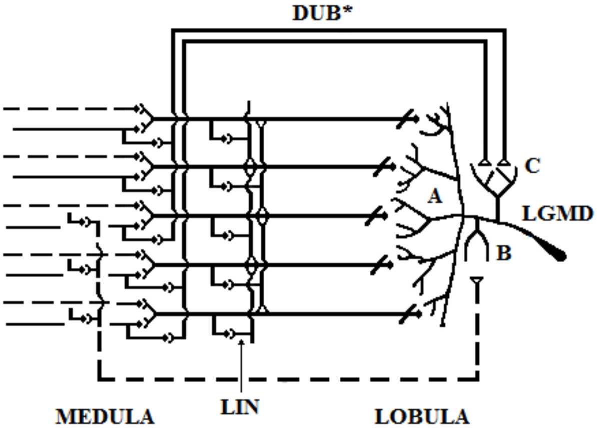


Figure 4.3: Input Circuit to the LGMD from Rowell et al. (1977))

Recordings of the responses of the LGMD to objects moving towards or away from the animal showed an increasing spike rate for an approaching object but only a brief burst of spikes for a receding object, and it was suggested that the neuron is in fact tuned to detect approaching objects, a more relevant stimulus for triggering escape behaviour (Schlotterer 1977). Although this idea was not accepted initially (Pinter et al. 1982) it has now been verified (Rind & Simmons 1999). In addition, the spike rate has been found to be well correlated with the angular acceleration of the edges of the objects. An increase in the object edge length and an increase in edge velocity are necessary for the cell to distinguish an approaching object from a receding object (Rind & Simmons 1999). The neuron is precisely calibrated to looming objects on a collision course (Judge & Rind 1997): objects moving on a path which deviates from the direct collision trajectory by only a few degrees produce a much weaker response.

The responses to approaching objects occur due to an exponential build-up of excitation in the dendritic fan whereas the responses to receding objects show brief excitation followed by prolonged inhibition. likely to be due to feed-forward inhibition (Rind & Bramwell 1996). A neural network model produces responses which match the spike rate of the LGMD and demonstrates the contributions of excitation and inhibition.

Hatsopoulos et al. (1995) suggest that the spike rate peaks before the object reaches the eye which contradicts earlier findings that show the response continuing to rise until after the collision would have occurred (Rind & Bramwell 1996). Although the difference was attributed to the stimulus system used (Rind & Simmons 1999), a repeat of the experiments with different equipment has produced similar results to the initial study (Krapp et al. 1998). The results of these studies have been fitted with an equation that relates the response to the stimulus and predicts that during responses to objects looming on a collision course, the LGMD spike rate peaks when the stimulus subtends a fixed angular size on the eye.

4.4 Modelling of Neural Networks

Computational models of insect neural network are often used to explore the properties of neural systems which are rather difficult to discover experimentally. There are many techniques to build a model of a neural system, ranging from very detailed models of single neurons to large scale networks with simple neurons, and the neuromorphic engineering approach, which are briefly reviewed in this section,

A crucial decision when developing a model of a neural system is the level of complexity to include (Rall 1995), and as the number of neurons in a model increases, their complexity typically decreases. Here, neuromorphic engineering is employed to model the LGMD neural network by applying the design principles of neurobiological systems in electro-mechanical engineering. Neurobiological architecture and models inspire the design of intelligent systems by implementing the elementary analogue blocks and novel neural circuits to clone a particular behaviour. Carver Mead was the pioneer of neuromorphic engineering as he first embedded a neural circuit within a silicon chip (Mead 1989) whose first application was realised in the implementation of a spiking neuron in a silicon retina using an analogue circuit (Mahowald & Mead 1991). Visual, auditory, and other bio-sensors have since been designed and implemented successfully (Andreou et al. 1991, Mahowald & Mead 1991, Liu 1996, Lazzaro & Wawrzynek 1997, Harrison & Koch 1998, Kumar et al. 1998).

However, vision-based models are the most popular sensory circuits implemented, and as the complexity of these systems increased, the computation load was uniformly distributed among several processing nodes to form modular architectures (Higgins & Koch 1999, Higgins & Shams 2002, Ozalevli & Higgins 2005). Such bioinspired asynchronous communication protocols are implemented by several researchers to establish a multi-node contact in a multiprocessor system (Boahen 1998, Landolt et al. 2001). However, Sarpeshkar et al. (1996) addresses basic challenges such as dense, parallel architecture, and low power consumption while designing and prototyping motion processing system. Higgins et al. conducted a survey on three individual motion detector based models to illustrate each one's performance against an identical test case and condition (Higgins et al. 2005). Nevertheless, successful design of bioinspired computational models for collision detection, time to collision, and target tracking using VLSI processors are described in Etienne-Cummings et al. (1996), Indiveri et al. (2002), Etienne-Cummings et al. (2000), Higgins & Pant (2004), Harrison (2005). Analogue circuit architecture is advantageous as it can replicate near-exact neuronal processing by running the algorithm in continuous time, additionally, power consumption on an analogue processor operating at sub-threshold regime of the MOSFET is minimal thus making it the most suitable alternative to be implemented in an autonomous sensor system.

4.5 Computational LGMD Model

The original LGMD model described in Blanchard (1998), Blanchard et al. (2000), Yue & Rind (2006) emulates the input organisation and data processing of a *locust* visual interneuron, which inspires the core of obstacle detection algorithm proposed in this thesis. The LGMD architecture described in this study is an enhanced version of the models discussed in [13, 15 and 16]. It constitutes four groups of cells namely, photoreceptor cells (P layer), excitatory cells (E layer), inhibitory cells (I layer), and summing cells (S layer). Besides that, it constitutes two single cells namely, the feed-forward inhibition cell (FFI cell) and the LGMD cell. The model architecture is illustrated in Figure 4.4

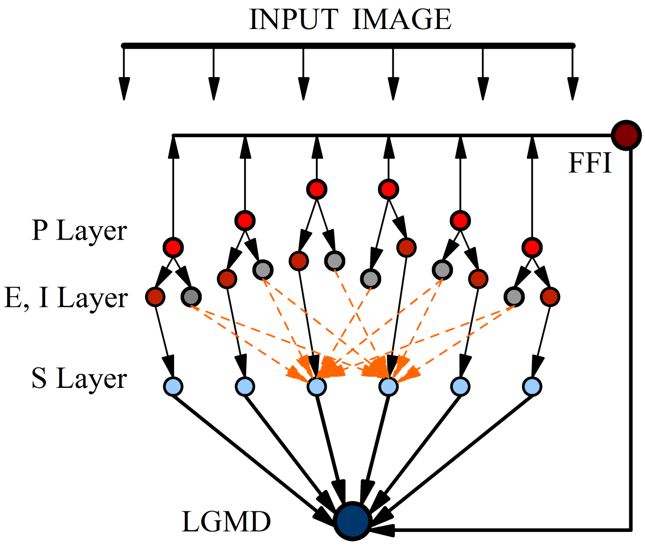


Figure 4.4: Original LGMD Model (from Rind & Bramwell (1996))

The input signal is a grey scale 2 dimensional image at a very low resolution and limited field of view. The image is represented as an array of values zero to 255, forming the input data to photoreceptor nodes (P-layer), which computes the absolute difference between the previous and current input image-luminance, mathematically represented as,

$$P_f(x, y) = |L_f(x, y) - L_{f-1}(x, y)| \tag{4.1}$$

where P_f is the result of layer-P processing at frame f, L_f and L_{f-1} are the image luminance of current and previous frames denoted as f and $f - 1$, respectively. The signal from layer-P provides an input to two layers namely, the inhibition (I) and excitation (E) layer. The excitations coming from layer-P transfer to the retinotopic counterparts directly, whereas the inhibition performs a convolution on the P signal to achieve a blur effect described by the equation,

$$I_f(x, y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} P_{f-1}(x + i, y + j) w_I(i, j), (i\&j \neq 0) \tag{4.2}$$

where w_I is the 3x3 local inhibition weight, further the excitation layer passes the luminance difference directly as,

$$E_f(x, y) = P_f(x, y) \tag{4.3}$$

where I_f is the inhibition signal at current time-step (frame f), and P_{f-1} is the delayed signal from layer-P denoted by $f - 1$. Further, the excitation and inhibition signals are retinotopically summed in the S-layer represented as,

$$S_f(x, y) = [E_f(x, y) - W_I.I_f(x, y)] \tag{4.4}$$

where S_f is the summed output signal at current frame f, with a global inhibition weight W_I that is set as 0.35. However, Yue & Rind (2006) introduced a noise removal layer that eliminates background noise by clustering the excitations in layer-S used to compute the membrane potential for LGMD cell. The process of clustering provides greater individual inputs than the isolated excitations in layer-S. These clusters of excitation are further multiplied by a transient coefficient Ce_f, whose value directly depends on the neighbouring pixels represented as,

$$Ce_f(x, y) = 1/9 \sum_{i=-1}^{1} \sum_{j=-1}^{1} S_{f-1}(x + i, y + j).w_e(i, j) \tag{4.5}$$

where $w_e(i,j)$ is the transient weight (convolution mask) representing the influence of neighbouring pixels in form of a 3×3 all-ones matrix. Now the LGMD cell for current time-step, G_f is summed along rows and columns to produce,

$$\mathrm{G}_f = \sum_{x=1}^{n}\sum_{y=1}^{m} |S_f(x,y)|.Ce_f(x,y).\omega^{-1} \tag{4.6}$$

where n is the number of rows, m is the number of columns, and ω is a scale that is computed at every time-step (frame),

$$\omega = \max(Ce_f).C_w^{-1} + \Delta c \tag{4.7}$$

where Δc is empirically set as 0.01 and C_w as 4.0 to prevent zero values of ω as Ce_f and ω differ for every time-step. Further, spiking mechanism is introduced by computing the sigmoid transformation of LGMD cell's membrane potential as,

$$g_f = (1 + e^{-G_f/n_{cell}})^{-1} \tag{4.8}$$

where n_{cell} represents the total count on summing cells, and the normalised membrane potential $g_f \in [0.5\ 1.0]$ since $\mathrm{G}_f > 0$. Finally the model response and avoidance decision is represented by a spiking mechanism depending on an adaptable threshold T_s introduced in Yue & Rind (2006), which begins with an initial value of T_{in} and updates for every m-frames during the process formulated as,

$$T_s = \begin{cases} T_s + \Delta t, & \text{if } g_{av} > \Pi \text{ and } (T_s + \Delta t) \in [T_l, T_u] \\ T_s - \Delta t, & \text{if } g_{av} < \Pi \text{ and } (T_s - \Delta t) \in [T_l, T_u] \\ T_s, & \text{Otherwise} \end{cases} \tag{4.9}$$

where T_l and T_u represent the lower and upper adaptation limits respectively, Δt represents the step increments, and Π denotes a threshold to determine the average of membrane potential (g_{av}) from the frame $f - n$ to $f - k$ given as,

$$g_{av} = (n - k + 1)^{-1} \sum_{i=k}^{n} g_{f-i} \tag{4.10}$$

Exceeding of the membrane potential G_f beyond a threshold T_s generates a spike whose appearance over 4 consecutive time-steps is interpreted as a potential collision threat. Once a collision is detected, the vehicle initiates avoidance manoeuvres which in turn causes large excitation in the input signal. These excitations are mitigated by the feed forward inhibition layer (Yue & Rind 2006) to avoid haphazard control generation by masking undesirably huge perturbations. The FFI is delayed by one time-step represented as,

$$FFI_f = \sum_{x=1}^{n} \sum_{y=1}^{m} |P_{f-1}(x,y)|.n_{cell}^{-1} \tag{4.11}$$

Once the feed forward inhibition for the current frame (FFI$_f$) exceeds an empirically determined threshold level, the LGMD response is mitigated immediately.

4.6 Quadfurcated Luminance-Difference Processing

The proactive collision avoidance model proposed here is inspired by the luminance difference processing exhibited by an LGMD neuron. The algorithm architecture called the quadfurcated luminance-difference processing (QLDP) illustrated in Figure 4.5, is carefully structured and enhanced to attain maximum robustness required onboard a robot. Every layer performs a significant computation of data within the model where each node represents an individual process.

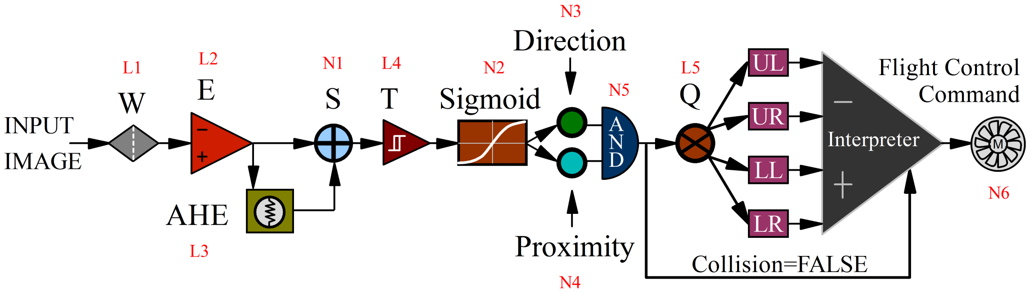


Figure 4.5: Quadfurcated Luminance-Difference Processing (QLDP) Model

The proposed architecture constitutes five layers of processing (L1,L2,...,L5) namely, linear time-invariant filter layer (W), excitation layer (E), adaptive histogram equalization layer (AHE), threshold layer (T), and quadfurcation layer (Q), preceded by six single processing nodes (N1,N2,...,N6) namely, the direction detector, depth estimator, summation, sigmoid transformer, AND logic gate, and flight control command generator node.

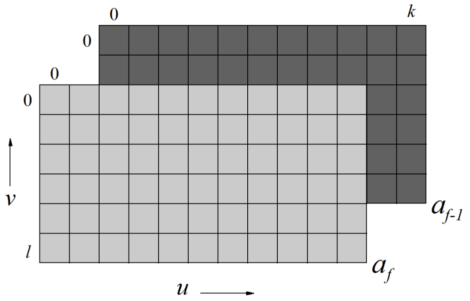


Figure 4.6: Schematic Illustration of a Sample Input Data

4.6.1 Input Image

As shown in Figure 4.5, the input image a_f is a sequence of f number of 2D arrays with a dimension $k \times l$ pixels, where each pixel value $(u, v) \in [0 - 255]$ distributed panoramically along u(horizontal axis) and v(vertical axis) representing grey scale image frames of the input collision movie illustrated in Figure 4.6.

4.6.2 Layer-W

Since UAVs exhibit vibrations over various frequencies, a linear time-invariant filter such as a Wiener filter (Lim 1990) is introduced as the first layer to eliminate noise (undesirable blur) adaptively within the input image exerting a minute computation load by estimating the local mean and variance around each pixel as,

$$\mu_f = \frac{1}{NM} \sum_{u=1}^{N} \sum_{v=1}^{M} a_f(u, v) \tag{4.12}$$

and

$$\sigma_f^2 = \frac{1}{NM} \sum_{u=1}^{N} \sum_{v=1}^{M} [a_f(u, v) - \mu_f]^2, \tag{4.13}$$

where N and M are the local neighbourhood of each pixel along horizontal and vertical axis respectively in the input image, which further creates a pixel-wise filter using,

$$b_f(u, v) = \mu_f + \frac{\sigma_f^2 - w^2}{\sigma_f^2} [a_f(u, v) - \mu_f] \tag{4.14}$$

where w^2 is the noise variance that is considered as the average of all the locally estimated variances.

4.6.3 Layer-E

The filtered signal is then fed to the excitation layer that estimates the luminance difference of two consecutive input image frames by computing the absolute difference of the value of each pixel within the 2D array of an image with respect to its previous time-step, mathematically,

$$E_f(u, v) = |b_f(u, v) - b_{f-1}(u, v)| \tag{4.15}$$

where E_f is the output of the excitation layer at frame-f, b_f and b_{f-1} are the filtered luminance at current and previous frames f and $f-1$, respectively.

4.6.4 Layer-AHE

The next processing node is the adaptive histogram equalizer (AHE) (Zuiderveld 1994) that enhances the excited frame's contrast, facilitating a fine boundary separation between the background and the edges of an obstacle by subdivision and interpolation scheme. As shown in Figure 4.7, grey-level attribute (denoted by a white dot) is determined by the grey-value distribution in its neighbouring contextual regions of a sample point in an image frame with centres denoted as L, M, N, and O where local grey-level mappings ($g_L(c)$, $g_M(c)$, $g_N(c)$, and $g_O(c)$) are based on the histogram of the contained pixels. Considering c as the original pixel intensity for the sample point, we compute its new value by bilinear interpolation of the grey-level mappings that were calculated for each of the neighbouring contextual zones as,

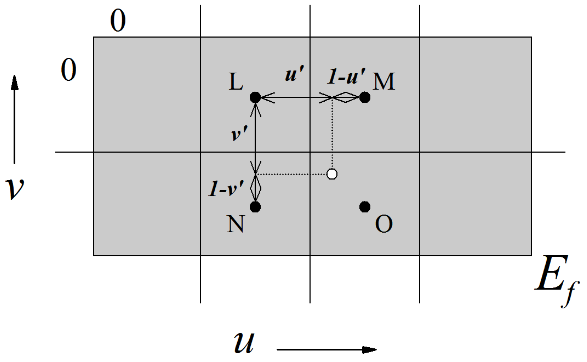


Figure 4.7: Contextual Regions of a Sample Point in an Image Frame.

$$\begin{aligned} c'(u', v') = \{&(1 - v')((1 - u')g_L(c) + u'g_M(c)) \\ &+ v'((1 - u')g_N(c) + u'g_O(c))\} \end{aligned} \tag{4.16}$$

Here u' and v' are the normalized distances with respect to the point L. The optimal contrast is thus calculated by dividing the entire image frame into such rectangular contextual elements shown in Figure (b) 4.7 (a sample zone), and then integrated to attain a whole contrast-equalized image, C_f.

4.6.5 Layer-S

The obtained contrast-corrected image C_f is added to the excited image E_f to get,

$$S_f = \rho_s \times |E_f + C_f| \tag{4.17}$$

We introduce the summed output with a sensitivity factor, ρ_s that is an empirically determined coefficient to help mitigate the undesirable excitations due to perturbations.

4.6.6 Layer-T

Furthermore, the resulting summed data S_f are treated with a simple binary thresholding image segmentation,

$$T_f(u,v) = \begin{cases} 1 \text{ (White)}\ , & \text{if } S_f(u,v) \geq T_r \\ 0 \text{ (Black)}\ , & \text{if } S_f(u,v) < T_r \end{cases} \tag{4.18}$$

Here the output T_f converts the input matrix S_f into binary dataset and replaces all pixels with luminance greater than the defined threshold (T_r= empirically determined) with the value 1 (white) and replaces all other pixels with the value 0 (black).

4.6.7 Response Generation Node

Ultimately the excited (white) pixels that have passed the threshold (T_r) are summed along both dimensions of the array to generate a system response as,

$$R_f(u,v) = \sum_{u=1}^{k} \sum_{v=1}^{l} |T_f(u,v)| \tag{4.19}$$

which is then fed to the spike generator to interpret the model response (collision alarms) as spikes. This is accomplished by transforming the response signal into a sigmoid function as,

$$A_f = (1 + e^{-R_f/s_{cell}})^{-1} \tag{4.20}$$

where s_{cell} is the total number of the response nodes, and since R_f is greater than zero, the normalized spiking response, $A_f \in [0.5 - 1.0]$.

4.6.8 Direction and Proximity Estimation Nodes

The generated spiking responses are then interpreted to decipher the nature of collision alarm, that is whether or not an obstacle is a potential collision threat. This is achieved by estimating the direction of an obstacle's motion relative to the vehicle and assigning it with a binary value $\in$ [0,1] using the condition,

$$Direction = \begin{cases} \text{Approaching} = 1, & \text{if avg}A_f > \text{avg}A_{f-1} \\ \text{Receding} = 0, & \text{if avg}A_f < \text{avg}A_{f-1} \\ \text{Stagnant} = 0, & \text{otherwise} \end{cases} \tag{4.21}$$

and its proximity estimator condition given as,

$$Depth = \begin{cases} \text{Imminent} = 1, & \text{if avg}A_f > (\delta \times \text{avg}A_{f-1}) \\ \text{Distant} = 0, & \text{otherwise} \end{cases} \tag{4.22}$$

where avgA_f is an average value of A_f over 4 time-steps, and δ is an empirically estimated coefficient that determines the nature of generated spikes implying if a threat is distant or imminent which in turn facilitates a prudent avoidance-decision making.

4.6.9 Quadfurcation Node

Direction and proximity estimation nodes lay the foundation of the quadfurcation process where an AND logic gate decides whether or not to initiate the process of quadfurcation (avoidance) by boolean multiplication expressed as,

$$\text{Quadfurcate} = \text{ Direction AND Proximity} \tag{4.23}$$

Following the AND gate truth table, the quadfurcation process is not initiated if either of the parameters ‘direction’ or ‘proximity’ is OFF, whereas if both parameters are excited exhibiting state 1, the algorithm quadfurcates the output of thresholding layer to form four individual image frames each representing one quadrant of the observer’s field of view, illustrated in Figure 4.8. It must be noted that the quadfurcation process introduced in this paper can be substituted by a bifurcation process for ground robots where the vehicle’s field of view is dissected in two equal halves (left and right) since there are only 3 degrees of freedom. Further processing is similar to the previous section but performed on four quarter images in parallel. Quadfurcation of T_f array is represented as,

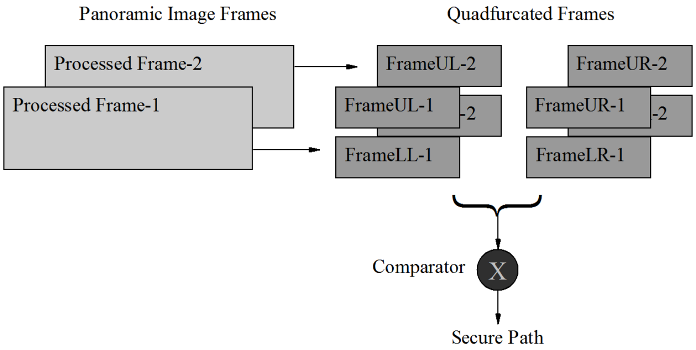


Figure 4.8: Quadfurcation process, where the input image frames are dissected to form four individual image frames each representing one quadrant of the vehicle’s field of view namely, upper-left (UL), upper-right (UR), lower-left (LL), and lower-right (LR).

$$T_{f_1}(x_1, y_1) = T_f(\frac{-u}{2}, \frac{v}{2}) \tag{4.24}$$

$$T_{f_2}(x_2, y_2) = T_f(\frac{u}{2}, \frac{v}{2}) \tag{4.25}$$

Similarly T_{f_3} and T_{f_4} are computed to be fed simultaneously to the spike generator as,

$$R_{f_1}(x_1, y_1) = \sum_{x_1=1}^{k_1} \sum_{y_1=1}^{l_1} |T_{f_1}(x_1, y_1)| \tag{4.26}$$

and

$$A_{f_1} = (1 + e^{-R_{f_1}/m_{cell}})^{-1} \tag{4.27}$$

where m_{cell} is the total number of the response nodes and since R_{f_1} is greater than zero, the normalized spikes, $A_{f_1} \in [0.5 - 1.0]$.

4.6.10 Motor Command Generation

Further their average over four time-steps (frames) is computed by,

$$\text{avg}A_{f_1} = \frac{\sum_{j=0}^{4} A_{(f-j)_1}}{4} \tag{4.28}$$

Similarly, the average response of the remaining three quadrants are computed in parallel which are further compared against each other to estimate the most secure path as,

$$\text{MotorCmd} = \text{avg}A_{f_1} - \text{avg}A_{f_2} - \text{avg}A_{f_3} - \text{avg}A_{f_4} \tag{4.29}$$

where ‘MotorCmd’ is expressed in terms of flight control commands involving various combination of basic manoeuvres such as left, right (pitch, yaw, and roll,) trimmed to navigate the robot through the estimated secure path. Detailed experimental results and analysis of the proposed algorithm for different scenarios, velocities, and test environments are described in the following chapter.

4.7 Summary

This chapter demonstrates that it is possible to achieve both simplicity and precision simultaneously using biological solutions. Inspired by these design trade-off strategies, the proposed QLDP in this thesis exploits such computational simplicity to address the intricate task of obstacle detection and avoidance for micro/nano robots. Major collision avoidance challenges addressed in this thesis are,

- Vibrational noise resistance (blurring)
- Irregular contrast (lighting) correction
- Obstacle size, colour, and shape independence
- Obstacle proximity and direction estimation
- Extreme computational efficiency

Although the contemporary software and hardware technologies are far from comparison with regard to their biological counterparts, researchers are continuously striving to gain momentum in this field, and this thesis similarly attempts for the same.

5

QLDP Visualisation & Modulation

"When we try to build autonomous robots, they are almost literally puppets acting to illustrate our current myths about cognition."

– Inman Harvey

In order to conduct a prudent experimental analysis in the next chapter, a comprehensive set of input data is congregated here to test the model for various possible scenarios preparing the algorithm for a successful real-world real-time application. Input dataset collection is performed systematically considering every vital parameter such as field of view (FOV), data acquisition frequency (Hz), input data resolution, dimension, and format. Similarly, the collision scenarios are orchestrated precisely by defining sample trajectories, various constraints, obstacles, and backgrounds to emulate near-exact real-world conditions.

Further, the model in its simplest form is fed with a sample input data to visualise the performance of each processing node individually and calibrate them consistently. A MATLAB user interface (UI) is designed for this purpose facilitating much straightforward analysis and calibration process that further helped establishing a precise trade-off strategy. The strategy involved elimination of insignificant processing nodes and layers increasing the available computation power that could accommodate the incorporation of advantageous complementary modules such as contrast corrector and proximity estimator layers in the finalised algorithm.

5.1 Input Dataset

As mentioned earlier, for a systematic experimental analysis there is always a need for a comprehensive input dataset prepared according to a established standard procedure reported in the contemporary literature, this requires a clear definition of input data parameters and configuration listed in Table 5.1,

Table 5.1: Off-line Test Input Data Specifications

Field of View (FOV)	90°
Data Acquisition Frequency	30Hz
Linear Forward Velocity Range	0.2-2.0 ms^{-1}
Input Data Dimension	2D
Image Resolution	320x240 Pixels
Recorded Colour Space	Grey Scale
Data Format	.mpg

Once the input parameters are established, the next step involves collision scenario emulation for various assessment parameters such as constant looming velocities, obstacle/background size, colour, and contour.

5.1.1 Collision Avoidance Scenarios

The collision scenarios emulated in this section are further implemented in model analysis. These involve assessment of specific modules introduced within the proposed system, hence, a precise validation of the claimed functionalities is performed by testing the model against a particular scenario to challenge potentialities that include,

1. **Static Obstacle Detection:** This scenario involves movement of the robot at a constant velocity of $\sim$1 ms^{-1} on a straight trajectory that involves two static obstacles of 0.5 m and 1 m width placed consecutively at a 5 m separation, which must be detected and avoided successfully by the robot. Since these tests are off-line, a short epoch is considered for the algorithm to respond to the detected objects and generate collision-free manoeuvres.

2. **Obstacle Direction Detection:** This involves a relatively similar scenario but the obstacles on collision course are static where the first obstacle recedes the observer (robot) but the second obstacle is on collision course approaching the vehicle on a lateral path. The ability of the system to distinguish a potential threat (approaching) from a non-threatening obstacle (receding) is tested using this input dataset.

3. **Proximity Estimation:** This scenario is recorded solely to test the algorithm's ability to determine obstacle proximity, that is, to differentiate between a large distant obstacle approaching and a small imminent one. This involves a looming velocity of 1.2 ms^{-1} towards two static obstacles of 0.5 and 1 m width placed consecutively 5 and 12 m away from the robot's initial position, respectively.

5.1.2 Simulation and Recording

The above described scenarios are orchestrated with varying combinations of hardware configurations that are setup on the robot and data-capture device. Ultimately, the collision videos are recorded and montaged in accordance to the described requirements considering standard procedures reported in current literature.

5.2 Collision Avoidance Model Analysis

Analysis of the proposed model to assess its performance for varying test cases/scenarios is extremely crucial to establish a precise trade-off strategy achieving the asserted system capability or contribution to the respective field. This section demonstrates a detailed analysis of QLDP against a sample off-line collision scenario to visualise the performance of each processing node individually. This analysis involves,

- Description and demonstration of a sample off-line collision scenario (described in previous section) imposing individual challenges to the asserted capabilities of the proposed QLDP.

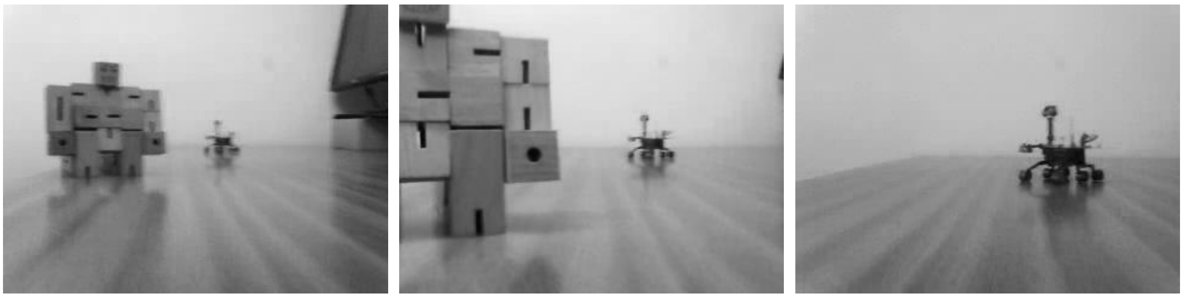
- Detailed demonstration of the QLDP response and results obtained for each individual complementary module and processing layer tested with the same sample trial.

- Presentation of model response interpretation and motor command generation to calibrate the QLDP accordingly and thus establish a precise trade-off strategy.

Here, a sample input data is selected to visualise the model performance where each processing layer and node within the proposed model is tested against its effective contribution to the overall response and its share of computation load, which is further studied and taken into design-trade consideration when compared to reference performance parameters published in the associated research. The inference made with respect to the analysed performance of the system serves as the evidence to implement and validate the model for real-world real-time applications in the following chapter.

5.2.1 Sample Input Dataset

The input data presented here is a sample representation of the collision scenarios orchestrated to test individual prospects of the proposed model. This sample data involves an observer (robot) moving at a constant forward velocity of 1.2 ms^{-1} towards two static obstacles of 0.5 and 1 m width on the left and and right side of the vehicle's field of view (90°), respectively. These obstacles are placed clearly on the collision course and must eventually be avoided by the robot with a short epoch between each turn considered for off-line analysis purposes. Figure 5.1 illustrate sample image frames snapped from the input collision movie.



(a) Frame-15 (b) Frame-45 (c) Frame-115

Figure 5.1: Sample Input Dataset (Collision Video)

5.2.2 Wiener Filter Layer

As mentioned earlier, the input image a_f is a sequence of f number of 2D arrays with a dimension kxl pixels, where each pixel value $(u, v) \in [0 - 255]$, distributed panoramically along u(horizontal axis) and v(vertical axis) representing grey scale image frames of the captured collision video. As robotic platforms (particularly aerial) exhibit vibrations due to high speed rotors on various frequencies, a constant time-invariant filter such as a Wiener filter (Lim 1990) is introduced as the first layer to filter noise (undesirable blur) adaptively within the input image imposing a minute computation load (∼4 milliseconds to process 1 frame) by estimating the local mean and variance around each pixel as,

$$\mu_f = \frac{1}{NM} \sum_{u=1}^{N} \sum_{v=1}^{M} a_f(u, v) \tag{5.1}$$

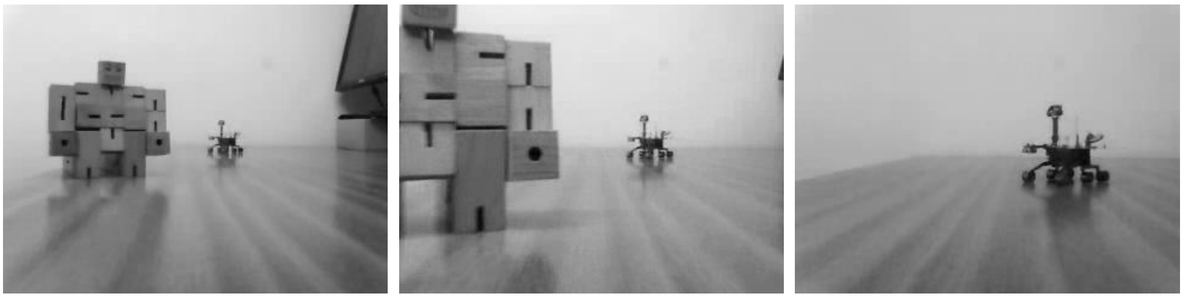
and

$$\sigma_f^2 = \frac{1}{NM} \sum_{u=1}^{N} \sum_{v=1}^{M} a_f^2(u, v) - \mu_f^2, \tag{5.2}$$

where N and M are the local neighbourhood of each pixel along horizontal and vertical axis respectively in the input image, which further creates a pixel-wise filter using,

$$b_f(u,v) = \mu_f + \frac{\sigma_f^2 - w^2}{\sigma_f^2}(a_f - \mu_f) \tag{5.3}$$

where w^2 is the noise variance that is considered as the average of all the locally estimated variances. However, since the representative trial data is collected in laboratory conditions, the extreme vehicular vibrations offered by high-speed rotors does not exist and only when observed closely, the figures illustrated in previous section possess minute noise and blur due to camera motion which are clearly eliminated and mitigated by the Wiener filtering at this stage shown in Figure 5.2.



(a) Frame-15 (b) Frame-45 (c) Frame-115

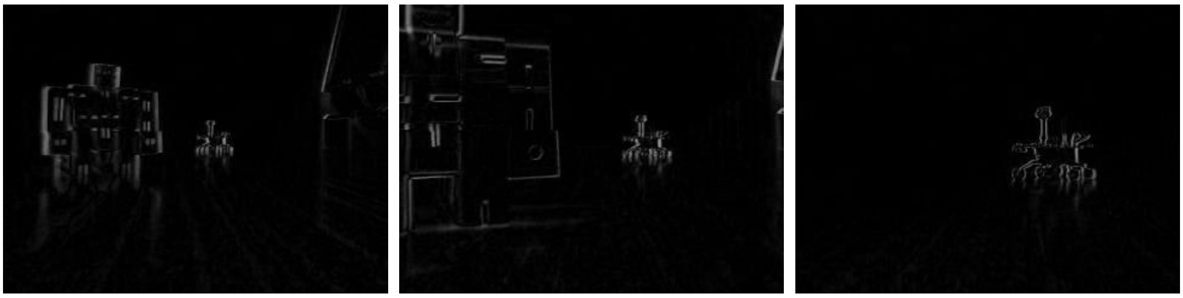
Figure 5.2: Linear Time-Invariant (Wiener) Filtered Frames.

5.2.3 Excitation Layer

One of the significant processing layers inspired by the original LGMD model is the excitation layer (luminance-difference processing). Here the absolute difference of two consecutive mitigated input image frames are determined to identify the changes occurred in the previous time-step, that is for an observer with 60 Hz acquisition frequency, any change in its field of view within 0.016 seconds is detected and passed on to next processing layer as an excited pixel, thus this method is particularly meant for machines whose position with respect to the surrounding changes continuously. Mathematically,

$$E_f(u,v) = |b_f(u,v) - b_{f-1}(u,v)| \tag{5.4}$$

where E_f is the output of the excitation layer at frame-f, b_f and b_{f-1} are the filtered luminance at current and previous frames f and $f-1$, respectively. This layer involves minimal computations as every pixel value of a image frame is subtracted from its corresponding pixel value at the previous time-step.



(a) Frame-15

(b) Frame-45

(c) Frame-115

Figure 5.3: Luminance-Difference Processed Image Frames.

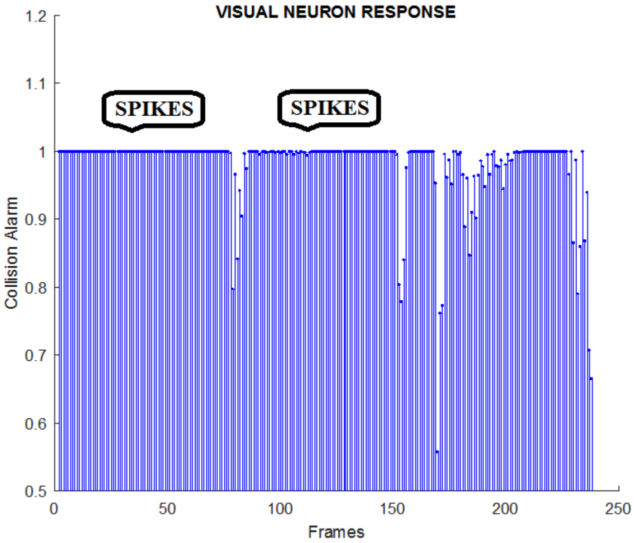
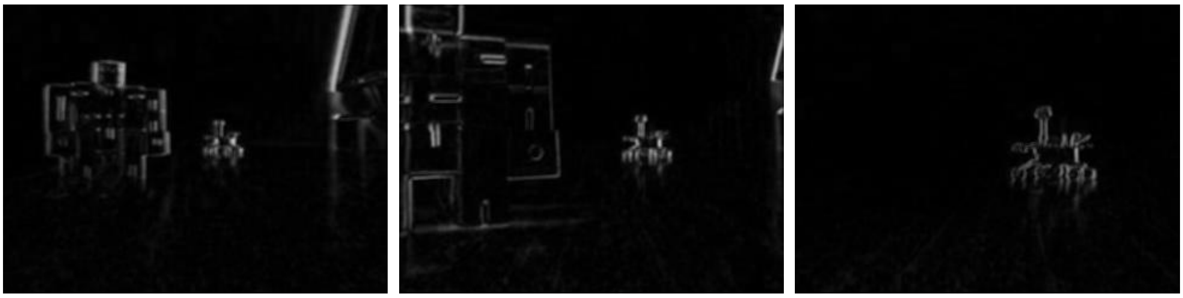


Figure 5.4: Model Response without E-Layer

Figure 5.3 illustrates the visualisation of luminance-difference processing. The excitation layer is also known as the QLDP's luminance-difference processing layer that performs a fundamental image segmentation to detect static obstacle edges. The response of the system without the excitation layer (which is one of the fundamental processing elements of the QLDP) is extremely vague as illustrated in Figure 5.4. The elevated spiking response shown (spike generation explained in section 4.6.8), is unnecessary due to the absence of the fundamental processing layer (luminance-differencing), passing every pixel as an excited (white) cell.

5.2.4 Inhibition Layer

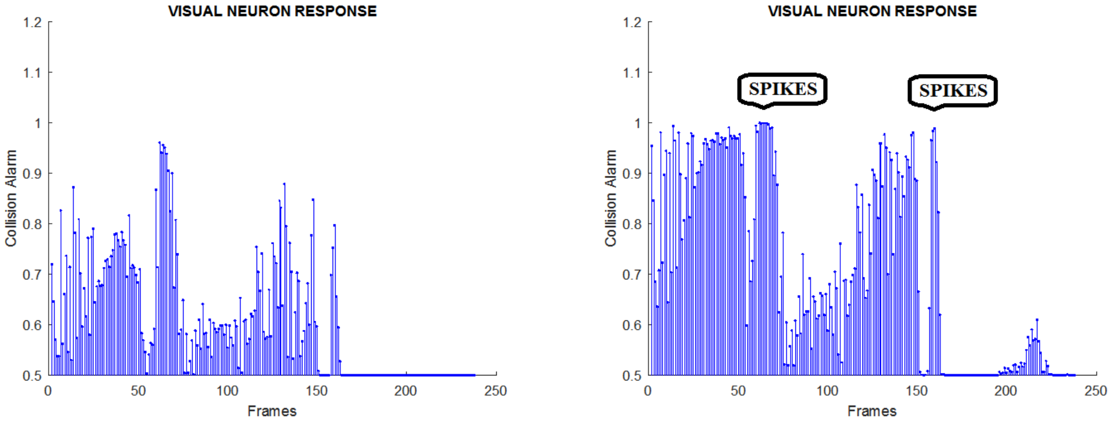
The inhibition is yet another processing layer within the computational model of a *locust* LGMD. This layer basically introduces a convolution with an empirically set kernel to inhibit excited (white) pixels from the output of excitation layer. It states that the convolution spreads only to the nearest neighbouring pixels thus an excited pixel does not inhibit itself, which is depicted from the biological model of *locust* visual system. From the Figure 5.5, it is



(a) Frame-15 (b) Frame-45 (c) Frame-115

Figure 5.5: Inhibited Image Frames.

evident that the layers contribution is not substantial. The justification for elimination of this processing layer from the model is bolstered by the Figure 5.6 which illustrates system response with and without the inhibition layer proving its contribution to be unnecessary as the excitations that are inhibited are already dismissed due their lower-than-unity spiking value.



(a) With I-Layer (b) Without I-Layer

Figure 5.6: Inhibition-Layer Processing Response.

5.2.5 Contrast Correction

As a result of inhibition-layer elimination, there is an increase in the available computation power accommodating a rather more effective and pragmatic processing layer to perform contrast correction facilitating model operation in environments with an irregular lighting. This layer performs Adaptive Histogram Equalisation (AHE) (Zuiderveld 1994) by creating a fine boundary separation between the background and the edges of an obstacle by subdivision and interpolation scheme.

As shown in Figure 5.7, grey-level attribute (denoted by a white dot) is determined by the grey-value distribution in its neighbouring contextual regions of a sample point in an image frame with centres denoted as L, M, N, and O where local grey-level mappings ($g_L(c)$, $g_M(c)$, $g_N(c)$, and $g_O(c)$) are based on the histogram of the contained pixels. Considering c as the original pixel intensity for the sample point, we compute its new value by bilinear interpolation of the grey-level mappings that were calculated for each of the neighbouring contextual zones as,

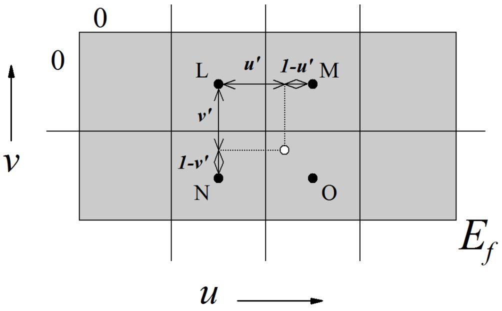
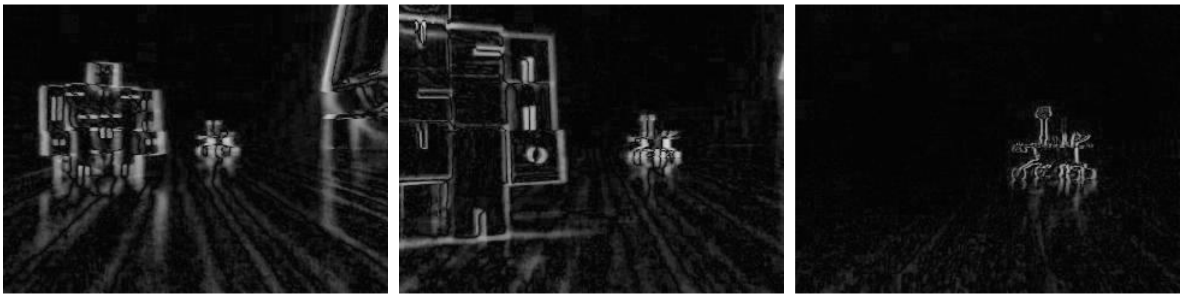


Figure 5.7: Schematic Illustration of Contextual Regions of a Sample Point in an Image Frame.

$$\begin{aligned} c'(u',v') = \{&(1-v')((1-u')g_L(c) + u'g_M(c)) \\ &+ v'((1-u')g_N(c) + u'g_O(c))\} \end{aligned} \tag{5.5}$$

Here u' and v' are the normalised distances to L. The optimal contrast is thus calculated by dividing the entire image frame into such rectangular contextual elements shown in Figure 5.7, and then integrated to attain a whole contrast-equalized image, C_f shown in Figure 5.8.



(a) Frame-15

(b) Frame-45

(c) Frame-115

Figure 5.8: Contrast Corrected Image Frames.

5.2.6 Non-uniform Illumination Correction

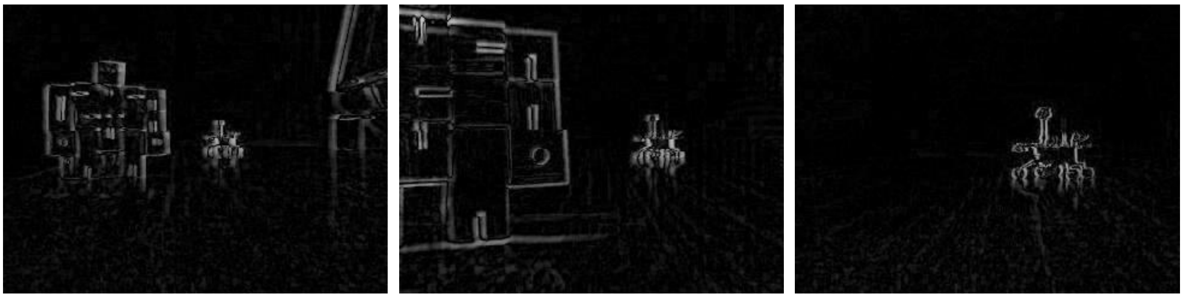
Illumination correction is another complementary module that could enhance the model performance. However, the trade-off strategy necessitated the elimination of this layer due to its massive computation load exerted on the algorithm. In this layer, a morphological-opening is performed to estimate the background illumination of the contrast corrected image by dilating its eroded form (Marques 2011) as,

$$C_f \circ O_f = (C_f \ominus O_f) \oplus O_f \tag{5.6}$$

which is then subtracted from the equalised image frame represented as,

$$I_f = C_f - (C_f \circ O_f) \tag{5.7}$$

To generate a uniformly illuminated output image frame that is immune to shadowing and irregular lighting effects. Although the representative input data does not exhibit severe shadowing, the results of this processing layer can be seen in Figure 5.9,



(a) Frame-15 (b) Frame-45 (c) Frame-115

Figure 5.9: Illumination Corrected Image Frames

5.2.7 Summing Layer

An intermediate processing layer called the summing layer is introduced immediately after generating contrast-corrected pixels whose values are summed with the excited pixels and depreciated to mitigate the effects of unprecedented abrupt motion causing large perturbations, or huge receding objects producing enormous excitations within the excited-frame pixel values. The obtained contrast-corrected image C_f is added to the excited image E_f mathematically represented as,

$$S_f = \rho_s \times |E_f + C_f| \tag{5.8}$$

The summed output is introduced with a sensitivity factor, ρ_s that is an empirically determined coefficient to help mitigate the afore mentioned undesirable excitations due to perturbations.

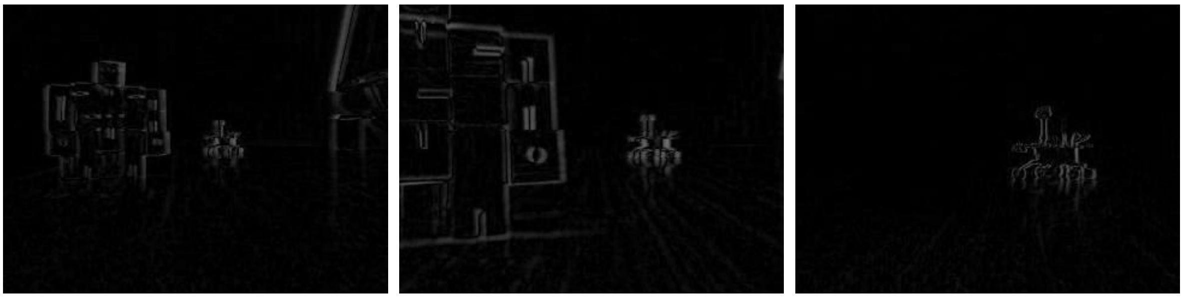

(a) Frame-15 (b) Frame-45 (c) Frame-115

Figure 5.10: Summed Image Frames

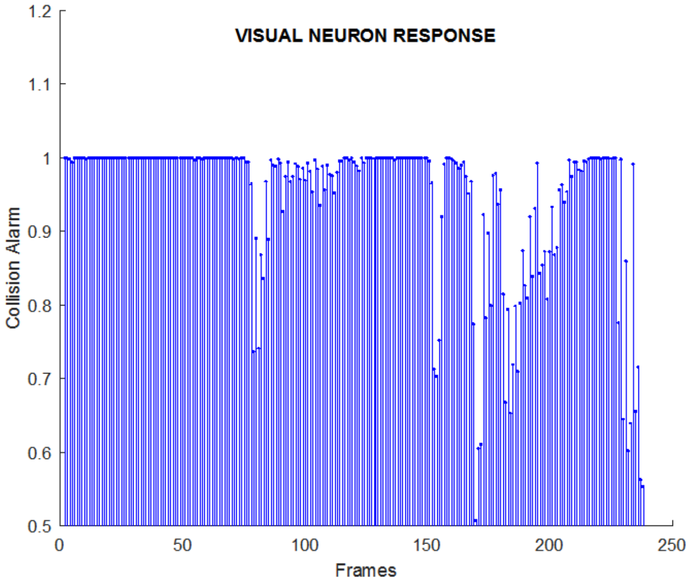


Figure 5.11: Model Response without S-Layer.

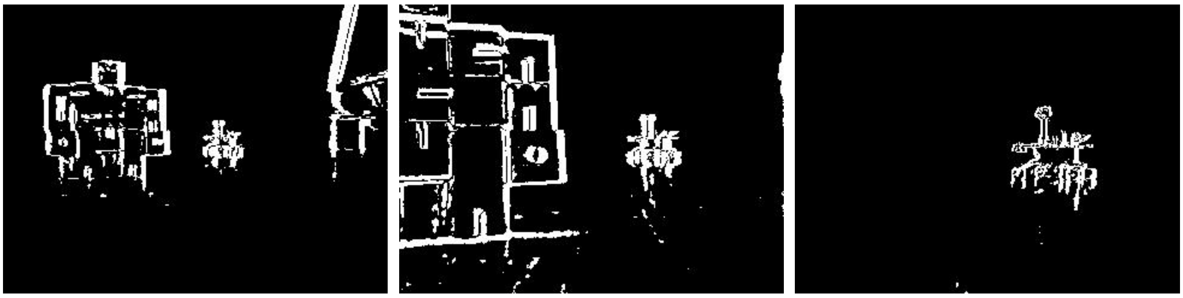
Figure 5.10 shows the summing layer effect on the contrast corrected images. The contribution of summing layer to the overall response is significant despite its simple operation as it provides a mild mitigation for undesirably large perturbations. As shown in Figure 5.11, the absence of summing layer results in an increase in the number of false collision alarms due to undesirable excitations making the overall system response futile.

5.2.8 Thresholding Layer

Ultimately a significant image segmentation layer is introduced where the layer converts an input array of pixel values into binary dataset, and replaces all the values with luminance greater than the empirically (trial and error) determined threshold (T_r) with the value 1 (white), and replaces all other pixels with the value 0 (black) as,

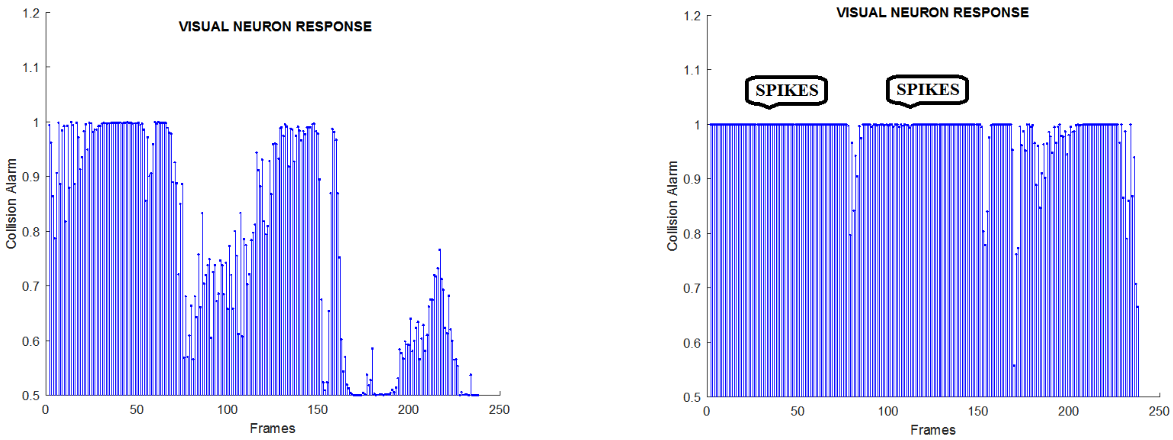
$$T_f(u,v) = \begin{cases} 1 \text{ (White) }, & \text{if } S_f(u,v) \geq T_r \\ 0 \text{ (Black) }, & \text{if } S_f(u,v) < T_r \end{cases} \tag{5.9}$$

This helps to establish distinct and sharp obstacle edges within the image frames, that can further be implemented in proximity and direction detection nodes. Overall system response is generated using the output pixel values of this layer shown in Figure 5.12. Elimination or an ineffective threshold value T_r fails to block undesirable excitations, which would eventually lead to a leakage of mildly excited pixels causing generation of false collision alarms shown in Figure (b) 5.13. However, Figure (a) 5.13 illustrates contribution of an empirically determined threshold maintaining the undesirable excitations below unit-spike level.



(a) Frame-15 (b) Frame-45 (c) Frame-115

Figure 5.12: Threshed Image Frames.

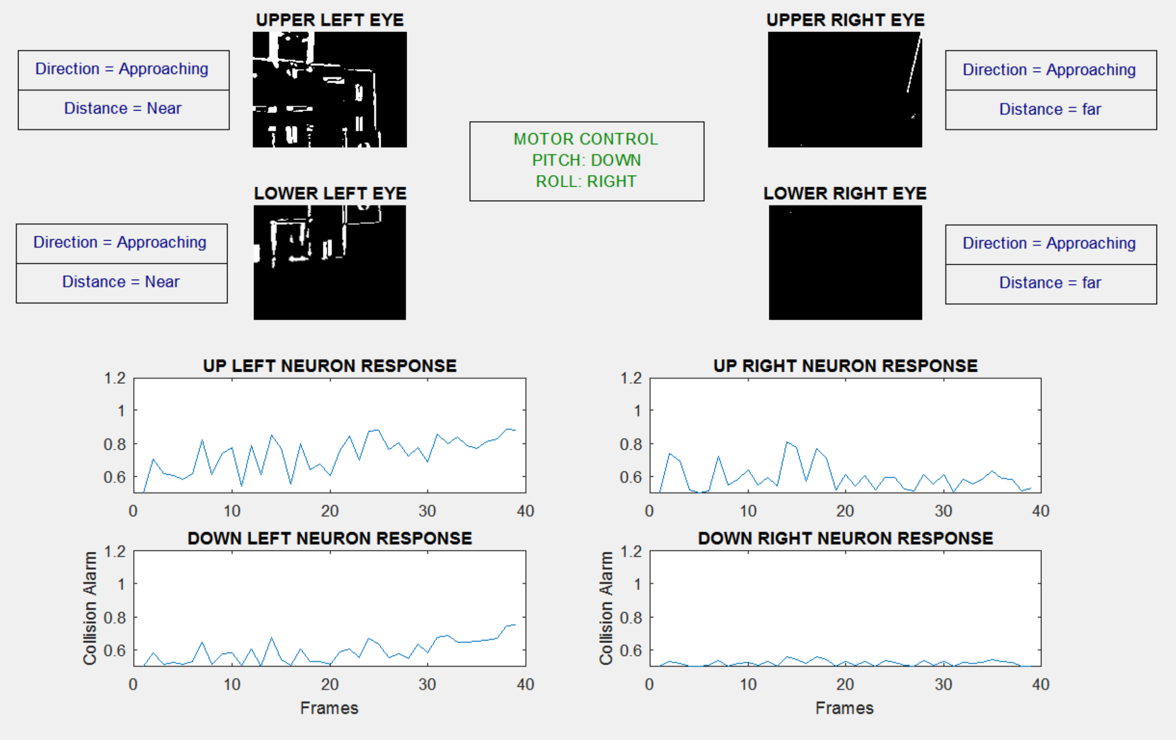


(a) With Thresholding (b) Without Thresholding

Figure 5.13: Model Response with and without Thresholding Layer.

5.3 Response Generation and Interpretation

In this section a MATLAB based user interface (UI) is designed and presented to facilitate a coherent visualisation of the system behaviour, response, and decision making. It consists of five major elements each representing a component of the system namely, data visualisation (robot's field of view dissected according to the robot's degree of freedom), motor control command, obstacle approach direction, obstacle proximity, and the spiking response (collision alarm). Figure 5.14 illustrates a typical view of the interface visualising system response during the program execution. Detailed definition and role of each element is described in following sections.



Direction = Approaching
Distance = Near

Direction = Approaching
Distance = far

MOTOR CONTROL
PITCH: DOWN
ROLL: RIGHT

Direction = Approaching
Distance = Near

Direction = Approaching
Distance = far

Figure 5.14: Model assessment-visualisation interface. Consists of five major elements namely, quadfurcated images, motor control command, obstacle direction, proximity estimation, and the spiking response.

5.3.1 Processed Data Visualisation

Figure 5.15 illustrates the visualisation element within the designed user interface which monitors the output image data from the final processing layer (thresholding) in real-time. Implementation of quadfurcation methodology is rather displayed in this window where the output frames from thresholding layer are dissected into four equal quarters along vertical and horizontal axes respectively emulating the field of view of each upper left, upper right, lower left, and lower right eye in clockwise order. It must be noted that the quadfurcation process introduced in this paper can be substituted by a bifurcation process for ground robots

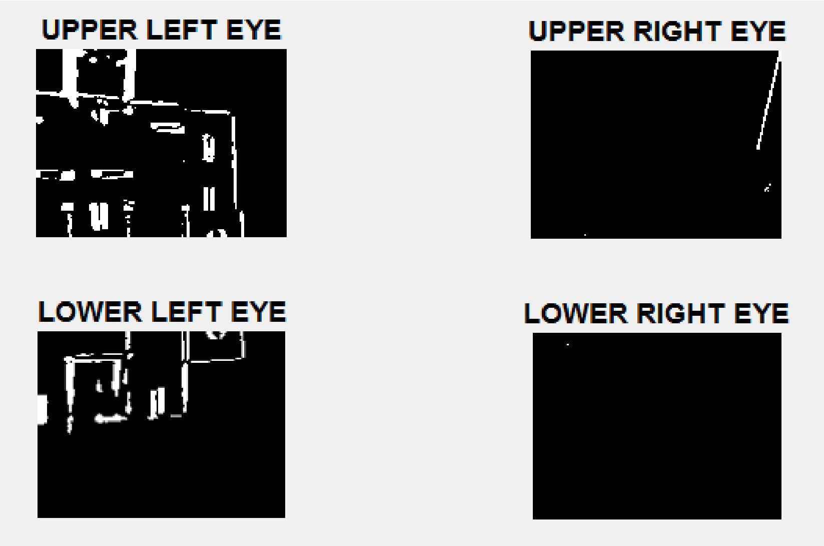


Figure 5.15: Processed Data Visualisation. Each quadrant represents one quarter of the robot's field of view that is quadfurcated.

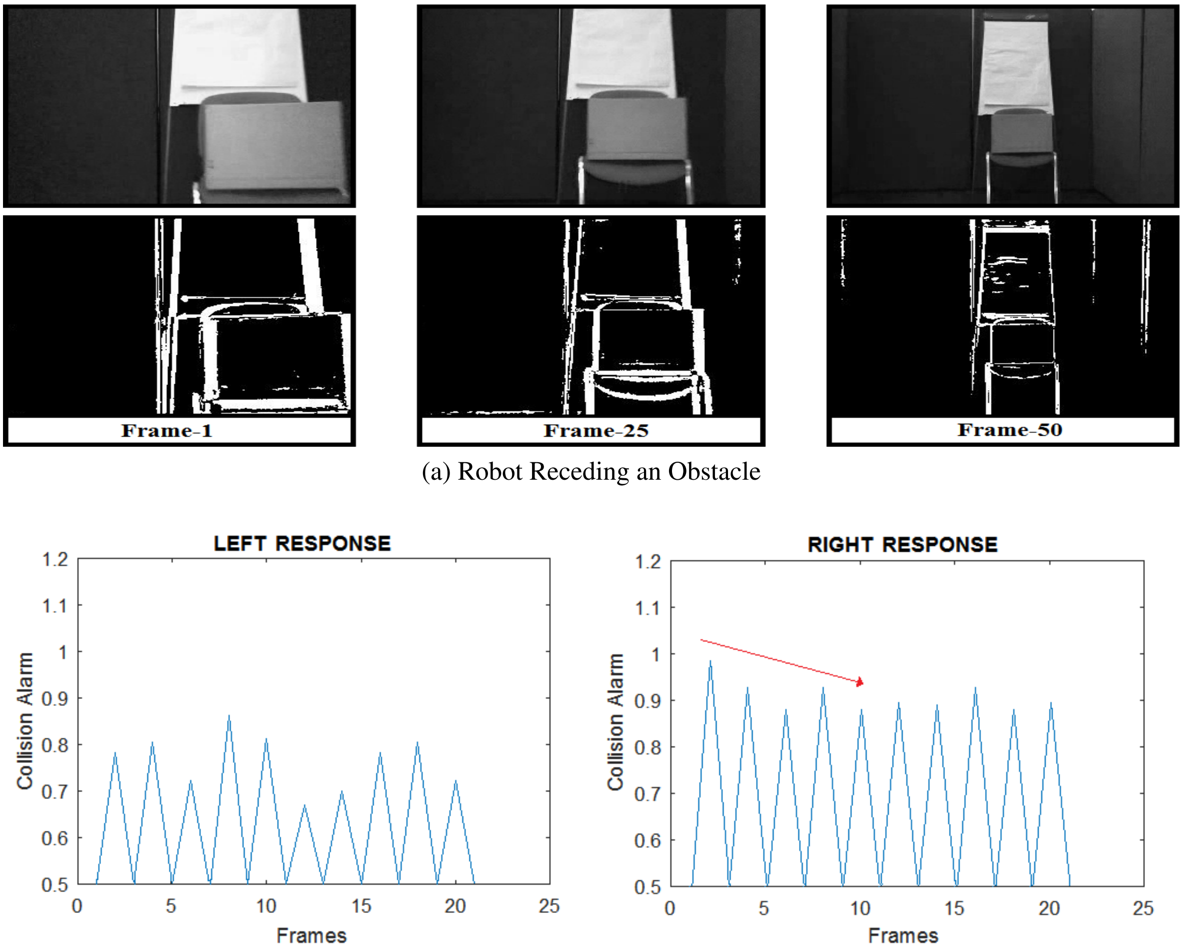
where the vehicle's field of view is dissected in two equal halves (left and right) since there are only 3 degrees of freedom.

5.3.2 Direction Detection

Direction in which the vehicle drifts with respect to an obstacle is estimated by comparing the spiking response of the model over three consecutive time-steps to check if the size of the object is increasing (approaching) or decreasing (receding) implying its course and threat potentiality. This is achieved by estimating the direction of an obstacle's motion relative to the robot and assigning it with a binary value ∈ [0,1] using the condition,

$$Direction = \begin{cases} \text{Approaching} = 1, & \text{if avg}A_f > \text{avg}A_{f-1} \\ \text{Receding} = 0, & \text{if avg}A_f < \text{avg}A_{f-1} \\ \text{Stagnant} = 0, & \text{otherwise} \end{cases} \tag{5.10}$$

where avgA_f is an average value of A_f over 4 time-steps (determined empirically), that helps in estimating the change in obstacle size (growing/reducing) to determine if the obstacle is approaching or receding facilitating a better avoidance-decision making. A collision test case (scenario) was orchestrated solely to evaluate the direction detection capability of the model, where an obstacle of 1 m height recedes the robot's field of view at 1.2 ms^{-1} causing a decaying response shown in Figure (b) 5.16 that is interpreted as a receding obstacle.

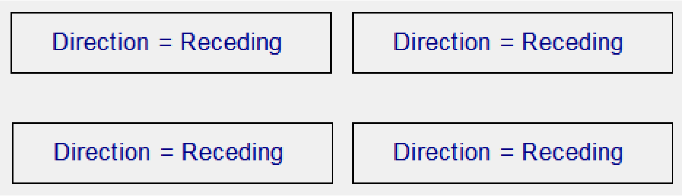


(a) Robot Receding an Obstacle

(b) Output Spiking Response

Figure 5.16: Model Test to Evaluate Direction Detection Capability.

Figure (a) 5.16 illustrates 1st, 25th, 50th image frames extracted from the direction-detection input test video. The detected direction is a triad denoted by either Approaching, Receding, or Stagnant annotated in blue font enclosed in rectangular boxes within the user-interface, shown in Figure 5.17.


Direction = Receding

Direction = Receding

Direction = Receding

Direction = Receding

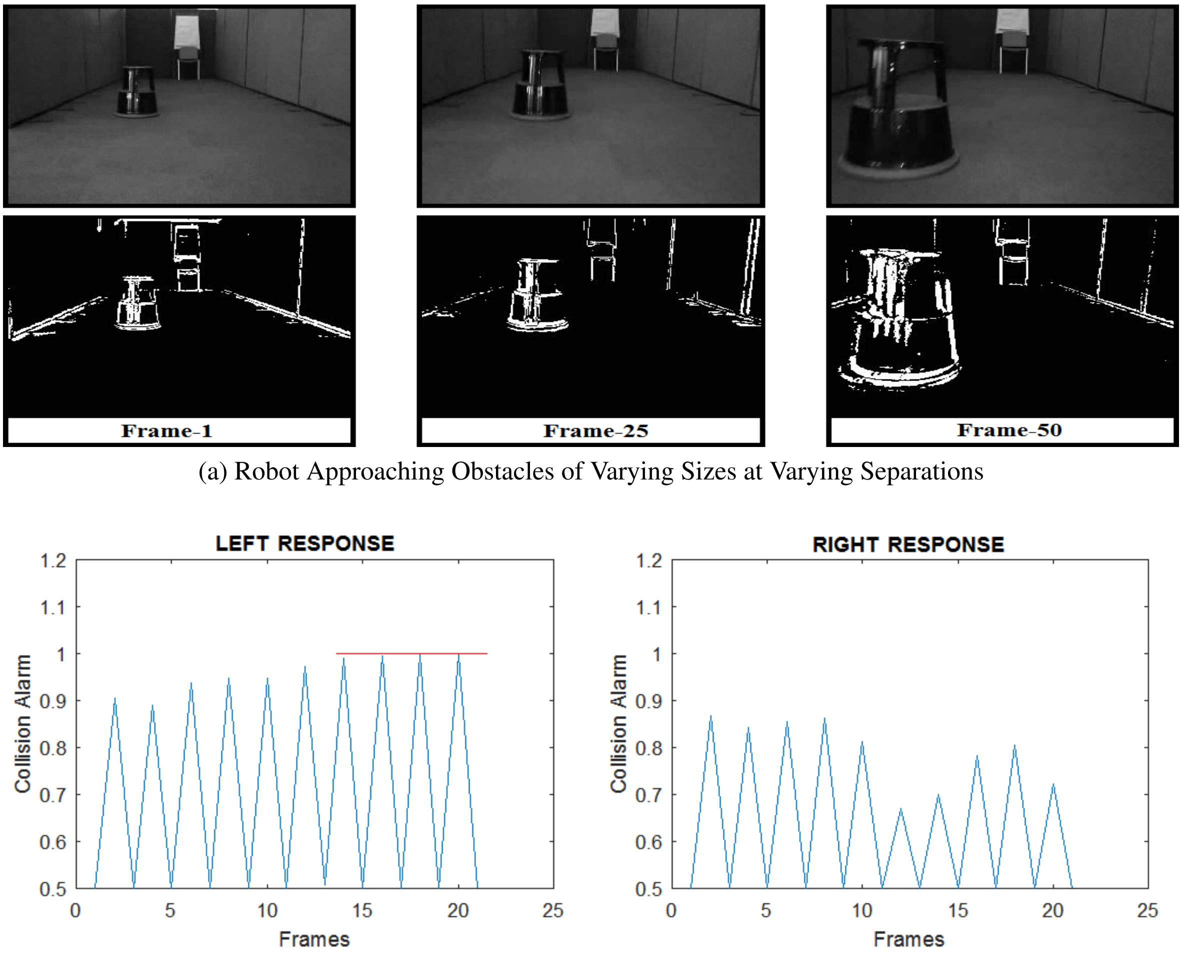
Figure 5.17: Obstacle Motion Direction Indicator

5.3.3 Depth Detection

Depth at which obstacles approach the robot is estimated by comparing the nature of growth of obstacle size that is on collision course and approaching the robot against an empirically determined coefficient δ. This is given by,

$$Depth = \begin{cases} \text{Near} = 1, & \text{if avg}A_f > (\delta \times \text{avg}A_{f-1}) \\ \text{Far} = 0, & \text{otherwise} \end{cases} \tag{5.11}$$

where δ is an empirically estimated coefficient that determines the nature of generated spikes implying if a threat is Far (distant) or Near (imminent) further facilitating an improved avoidance-decision making. To evaluate the proximity estimation capability of the model, a scenario is orchestrated where two obstacles approach the robot on collision course at 5 m separation, with the closer obstacle-A being 1/3$^{\text{rd}}$ of the farther obstacle-B causing the generation of a motor control command to lead the vehicle towards the obstacle-B (though larger) the rate of growth of its size remains smaller than obstacle-A that is imminent.



(a) Robot Approaching Obstacles of Varying Sizes at Varying Separations

(b) Output Spiking Response

Figure 5.18: Model Test to Evaluate Proximity Estimation Capability.

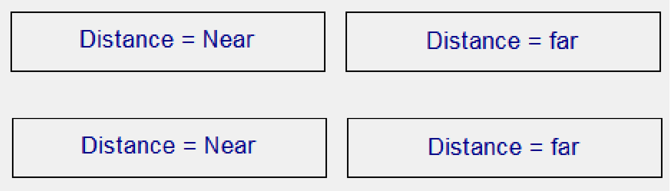


Figure 5.19: Obstacle Proximity Indicator

Figure 5.18 demonstrates looming of the obstacles at 1st, 25th and 50th time-step along with its generated spiking response depicting the approach of a potential collision threat starting from frame-13, hence providing more than 5 seconds long epoch to avoid the obstacle. The rationale is based on the two possible outcomes namely, obstacle is distant (logarithmic spike growth), obstacle is imminent (growth by a factor 'δ'). The output notation for this element is annotated in blue font enclosed in rectangular boxes within the UI, shown in Figure 5.19.

5.3.4 Spiking Response

Sigmoid transformation of the excited pixels that passed the threshold layer determines the spiking response of the model. This mechanism is employed here to generate spikes as collision alarms by summing the threshed cells along both horizontal and vertical axis,

$$R_f(u,v) = \sum_{u=1}^{k} \sum_{v=1}^{l} |T_f(u,v)| \tag{5.12}$$

Which is then fed to the spike generator to interpret the model response (collision alarms) as spikes. This is accomplished by transforming the response signal into a sigmoid function as,

$$A_f = (1 + e^{-R_f/s_{cell}})^{-1} \tag{5.13}$$

where s_{cell} is the total number of the response nodes, and since R_f is greater than zero, the normalized spiking response, $A_f \in [0.5 - 1.0]$, where 1.0 (spike) corresponds to an ON signal and every other value is OFF. Figure 5.20 illustrates the graphical visualisation

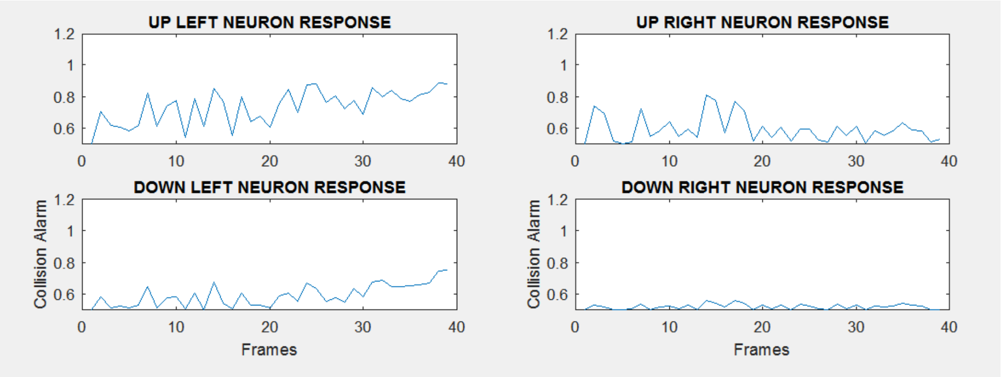


Figure 5.20: Spiking Response Plot

of the spiking mechanism that is built within the UI to represent the behaviour of each eye (quadrant) in response to stimuli. It is clear from the figure that upper-left and upper-right quarters of robot's field of view is experiencing greater threats thus causing the system initiate the quadfurcation process to generate collision-free motor control command.

5.3.5 Motor Control Command

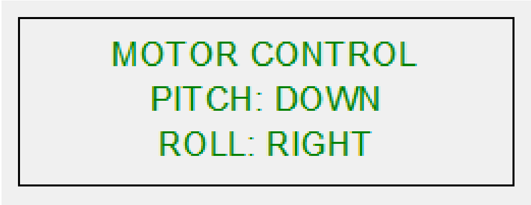
The interpretation of spiking response from the algorithm is performed by comparing the nature and intensity of spikes generated by each quadrant of the robot's field of view, and based on the comparison results over at least five time-steps the motor control (elevator and aileron) commands are generated with respect to robot's control surfaces. This process is represented as,

$$\text{avg}A_{f_1} = \frac{\sum_{j=0}^{4} A_{(f-j)_1}}{4} \tag{5.14}$$

Similarly, the average response of the remaining three quadrants are computed in parallel which are further compared against each other to estimate the most secure path as,

$$\text{MotorCmd} = \text{avg}A_{f_1} - \text{avg}A_{f_2} - \text{avg}A_{f_3} - \text{avg}A_{f_4} \tag{5.15}$$

The decision making (motor control) block exhibits either of the four possible outcomes corresponding to the four quarters of the robot's field of view towards which the robot is guided using its respective control surfaces (for aerial robot, aileron and elevator) that is visualised in real-time by annotating the commands in green font enclosed in a rectangular box within the UI, shown in Figure 5.21.



MOTOR CONTROL
PITCH: DOWN
ROLL: RIGHT

Figure 5.21: Motor Control Command

5.4 Summary

In this chapter, a comprehensive set of input data was congregated under a standard procedure reported in the associated literature, prepared for implementation in the off-line testing and performance assessment of the algorithm, in the following chapter. A detailed analysis of the proposed collision avoidance model is conducted and briefly demonstrated in this chapter. Based on the analysis results and system response, modulation of the model is carried out further to achieve maximum system robustness and reliability.

A successful representation and visualisation of the proposed algorithm's layer-by-layer processing, specifically the advantageous layers such as direction and proximity estimation facilitated a prudent modulation of the model, and assisted in establishing an efficient design trade-off strategy to acquire the most optimum performance versus computation load ready to be validated in the following chapter.

6

QLDP Performance Validation

> *"Vision is the process of discovering from images what is present in the world and where it is."*
>
> – David Marr

This chapter deals with the off-line and the real-world real-time implementation and testing of the proposed algorithm to assess and validate its performance and asserted capabilities. This is achieved by a systematic approach where the off-line collision scenarios orchestrated in the previous chapter are supplied to the model in form of a recorded input data, and the obtained system response is recorded for performance analysis. Further, for the real-world real-time testing, the robotic platform demonstrated in Chapter 3, is prepared and tested with a number of standard and arbitrary collision scenarios to challenge the algorithms specific claimed capabilities.

Ultimately, the performance of the algorithm for both the off-line and the real-world real-time tests are recorded and graphically compared to the contemporary associated literature to deliver a contrasting conclusion of the merits and demerits of the proposed algorithm in specific categories of (1) the number of detection errors, (2) exerted computation load, and (3) robustness and agility to respond to a potential collision threat.

6.1 Off-line Experimentation

A comprehensive off-line analysis of the proposed collision avoidance model is conducted and briefly demonstrated along with the inferences drawn for a wide range of collision scenarios. In order to conduct a prudent off-line analysis, a complete set of input data are gathered (Chapter 5) to test the model for various possible scenarios preparing the algorithm for a successful real-time real-world application. Input dataset collection is performed systematically considering every vital parameter such as field of view (FOV), data acquisition frequency (Hz), image resolution, dimension, and format listed in Table 6.1. Similarly, the collision scenarios are orchestrated precisely by defining sample trajectories, various constraints, obstacles, and backgrounds to emulate near-exact real-world conditions.

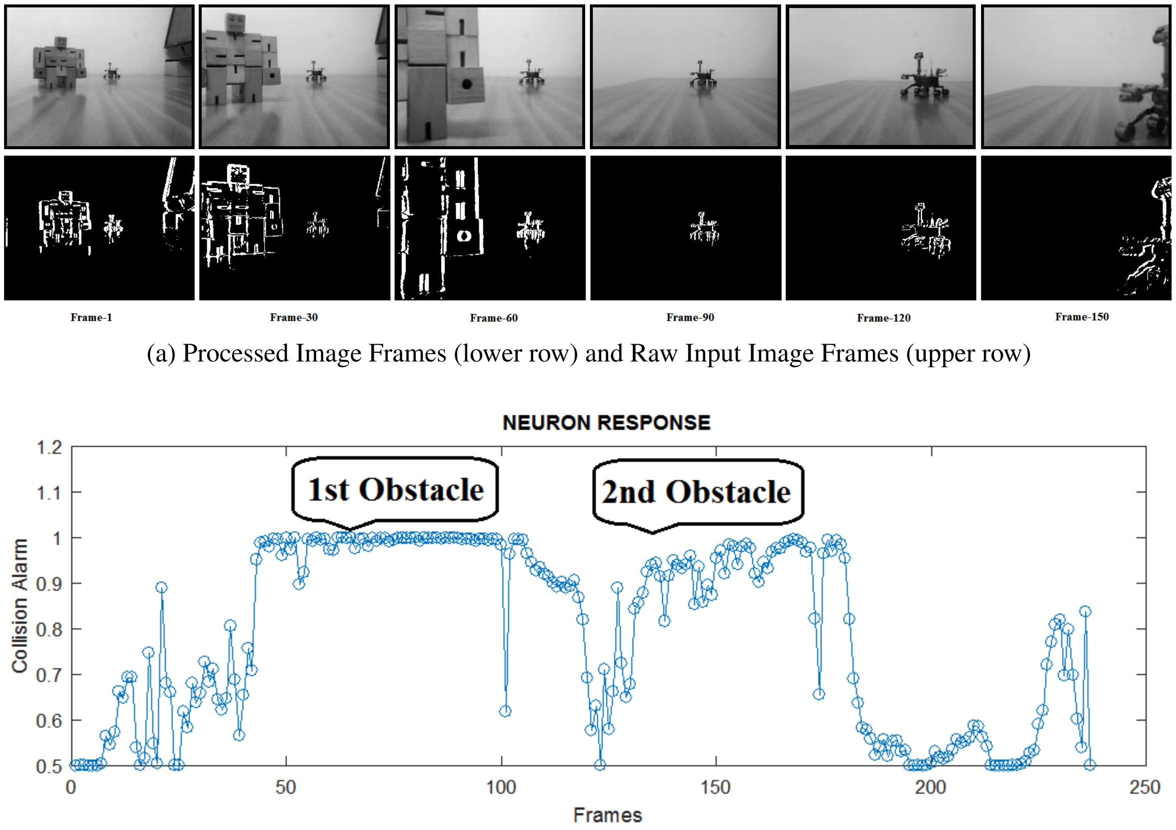
Table 6.1: Input Dataset Configuration and Setup

Input Data Capture	2-Megapixel HP CMOS Sensor
Field of View (FOV)	90°
Data Acquisition Frequency	30Hz
Robot Constant Forward Velocity	0.2-2.0 ms^{-1}
Input Data Resolution	2D RGB 320x240 Pixels
Central Processing Unit	Intel(R) Core(TM) i5-6500
CPU Clock frequency	3.20 GHz
Random Access Memory	8.00 GB
Operating System	MS Windows 7 (64-bit)

6.1.1 Collision Scenarios and Test Setup

The input data presented here is a sample representation of the collision scenarios orchestrated to test individual prospects of the proposed model. This sample involves an observer (robot) moving at a constant forward velocity of 1.2 ms^{-1} towards two static obstacles of 0.5 m and 1 m width on the left and right side of the robot's field of view (90°), consecutively. These obstacles are placed clearly on the collision course and must eventually be avoided. Figure (a) 6.1 illustrates the processed image frames (lower row) and raw input image frames (upper row) extracted from the input collision video. Figure (b) 6.1 represents the spiking response for the input collision video, where the spikes exceed 0.95 border from 40th and 135th frame for the first and second obstacle, respectively, implying that the obstacles are detected more than 3 seconds prior to contact.

In order to validate the proposed model, a number of collision scenarios are orchestrated with varying input parameters such as constant looming velocities, obstacle dimen-



(a) Processed Image Frames (lower row) and Raw Input Image Frames (upper row)

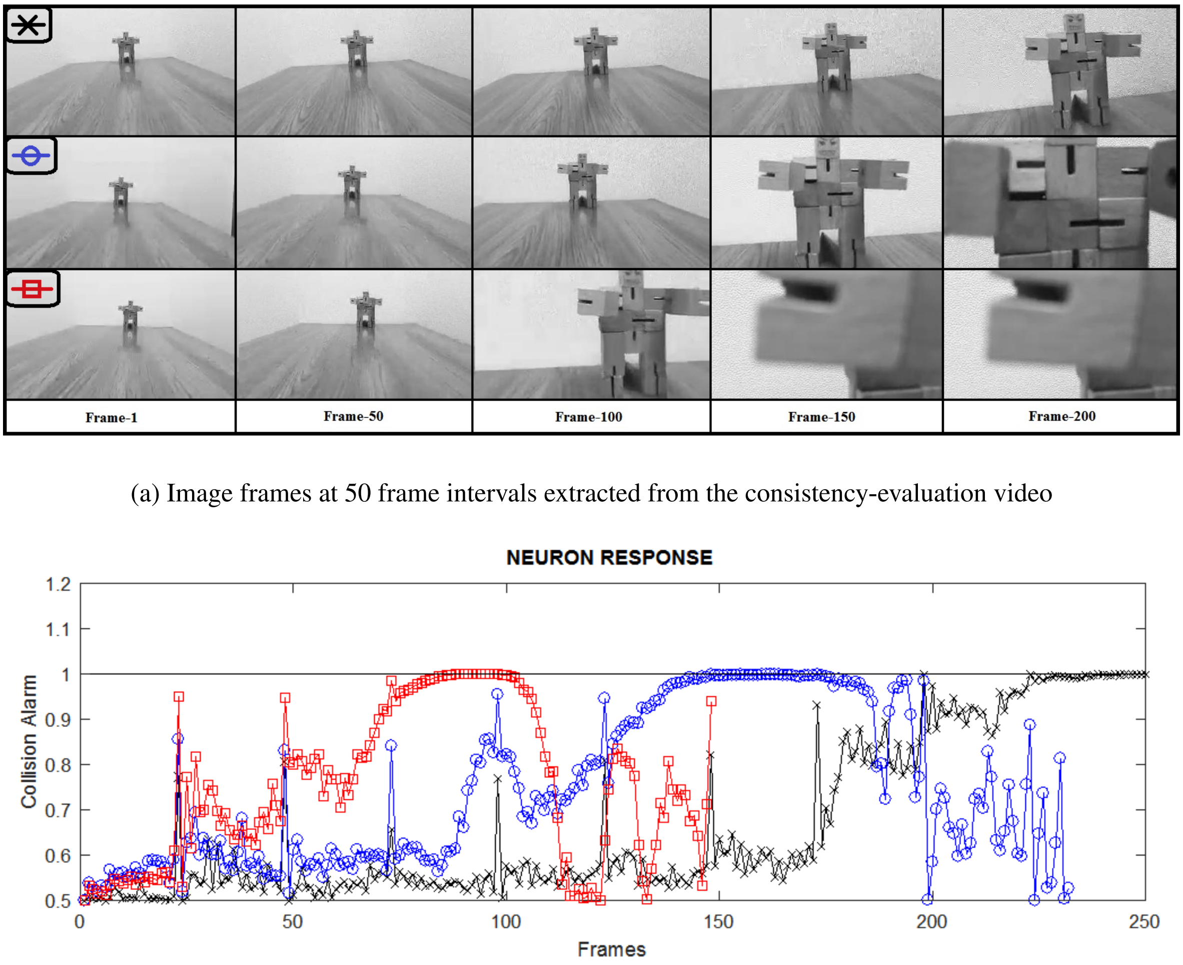
(b) Spiking Response for the Input Collision Video

Figure 6.1: Sample Off-line Experimentation.

sion, colour, and complex backgrounds. The collision scenarios represented in this section are implemented in model analysis involving assessment of specific modules introduced as the proposed model's novelties including, (1) Algorithm consistency and (2) Operation in complex backgrounds.

6.1.2 Model Consistency

A simple scenario is designed to evaluate the QLDP's response agility for various forward velocities. Here the robot approaches a brightly coloured static obstacle with an initial separation of 1.2 m at 0.24, 0.48, and 1.2 ms^{-1} corresponding to black cross, blue circle and red square, shown from top to bottom in Figure (a) 6.2, respectively. The spiking response for all three cases are presented in Figure (b) 6.2, which illustrates the black cross, blue circles, and red squares spike level rise to unity from the 200th, 130th, and 70th frame. Using the equation 6.1 from section 6.3.3, we compute distances prior to a potential collision as 0.66, 0.51, and 0.3 m, for the velocities in ascending order, respectively. Graphics in Figure 6.12.



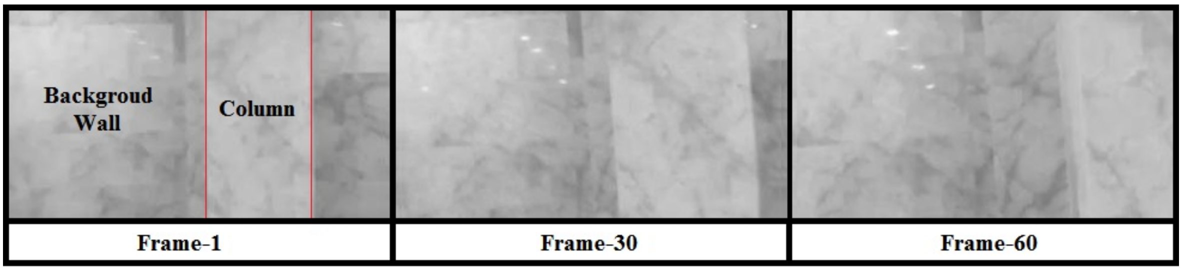
(a) Image frames at 50 frame intervals extracted from the consistency-evaluation video

(b) Output spiking response of the model over 250 frames

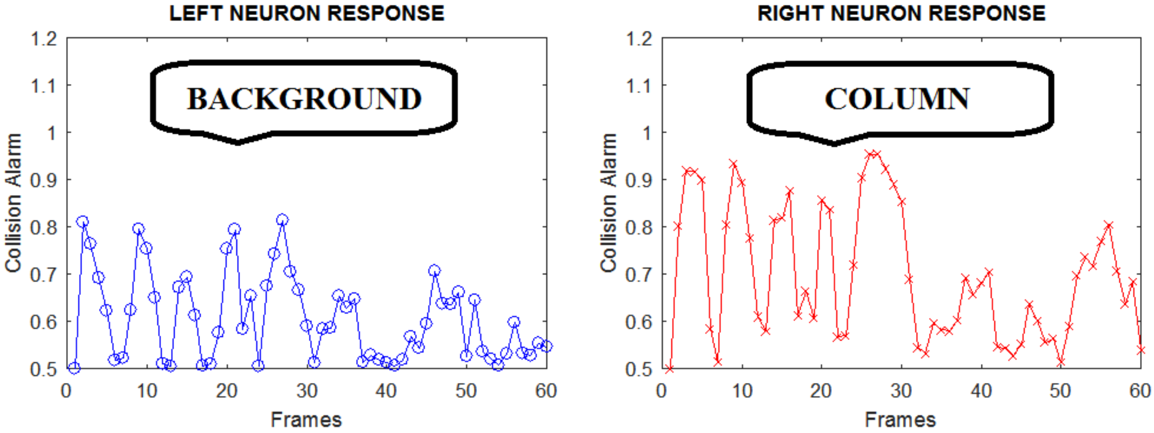
Figure 6.2: Consistency Evaluation Setup.

6.1.3 Complex Indoor Scenario

To evaluate the proposed model's capability to operate indoors, a complex-background scenario was orchestrated involving a brightly coloured column (pillar) surfaced with the exact same granite used on the background wall. The vehicle (robot) approaches the granite column at a constant velocity of 1.4 ms^{-1} travelling a distance of 5.6 m in 4 seconds. The model responds to the looming obstacle by increasing the collision alarm (spikes) shown in Figure 6.3, as the column impends, gradually the right half of the vehicle's field of view is activated, elevating the spike levels in the right neuron response coloured in red crosses. Further, the spike interpreter generates left-steering motor control commands to manoeuvre the vehicle away from the pillar. It is evident that the model successfully detects like-coloured obstacles and differentiates edges independent of their size, colour and contour.



(a) Image frames extracted from the column detection video (90Hz)



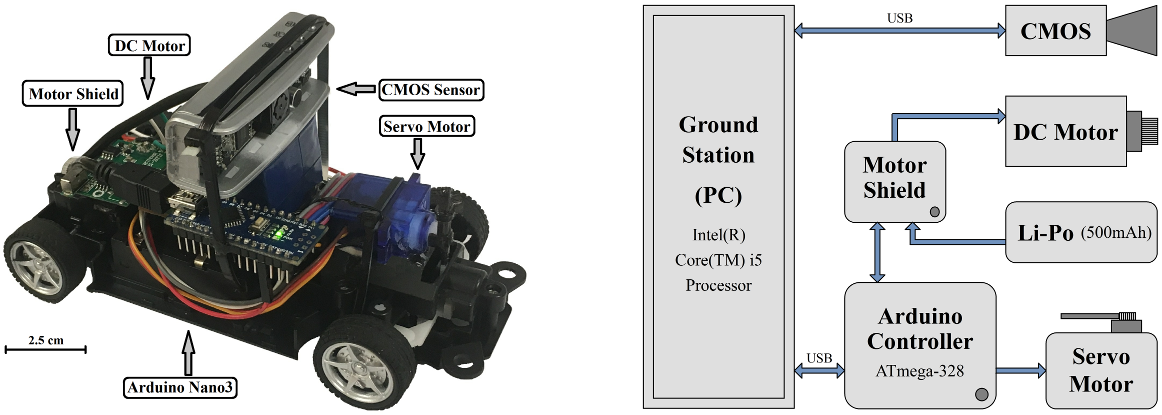
(b) Output spiking response of the model for the left and right half of the robot's field of view

Figure 6.3: Model Response to Evaluate Edge Detection Capability.

6.2 Real-time Real-World Experimentation

In this section, experimental setup and demonstration of model's real-world real-time implementation on a ground robot is presented. The algorithm is tested against a number of different collision scenarios involving multiple obstacles of varying colour, background, dimension, and contour.

The proposed model is implemented on a 3-DOF (degree of freedom) ground robot designed and fabricated at the Unmanned Autonomous Systems Laboratory (UASL), Cranfield University, to meet the specific requirements of the afore mentioned experimentations. It exhibits the necessary agility to accomplish successful collision avoidance using a DC motor, 9g servo motor, Arduino nano development board, motor shield, and a 2-Megapixel CMOS sensor. The Atmel ATmega328 8-bit AVR micro-controller with a maximum of 20 MHz operating frequency built into an Arduino development board is implemented to interface the robot with the proposed algorithm developed in MATLAB. Image frames captured by the CMOS sensor are transmitted through a USB cable to the ground station where the images are processed and motor control commands generated. These commands are transmitted through the micro-controller to the servo and motor shield to control the robot's locomotion.



(a) Assembled Robot

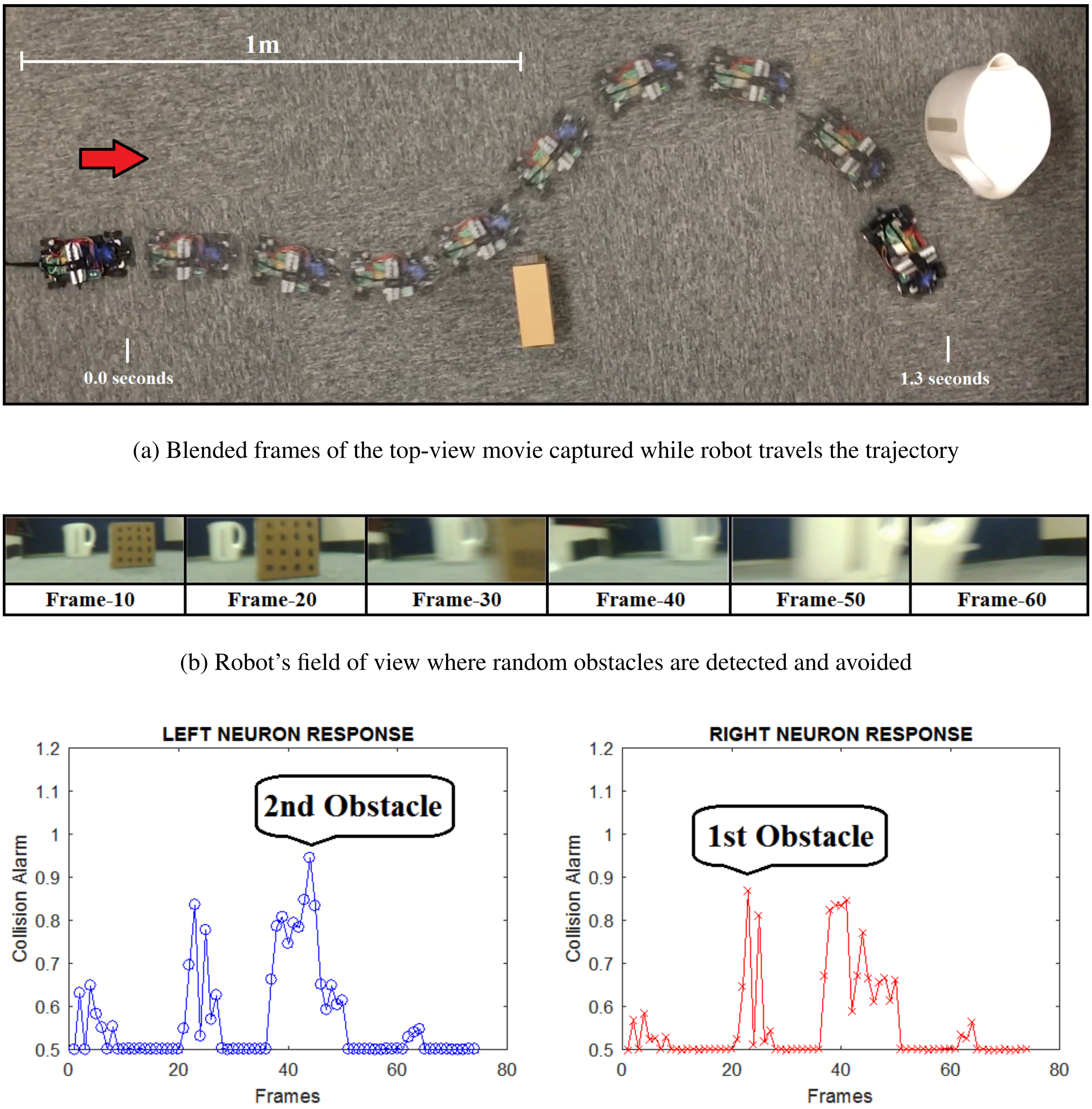
(b) Circuit Connection

Figure 6.4: Robotic Platform.

The conventional robotic differential steering was substituted with an Ackerman steering to replicate a near-exact real-world four wheel vehicle dynamics posing greater manoeuvring challenges as a result of underactuation. The schematic illustration of the designed robot, test platform and circuit connections are shown in Figure 6.4.

6.2.1 A Sample Real-World Scenario

As mentioned in Section 4.6.9, the vehicle's field of view is bifurcated here (3-DOF Robot). A top-view camera and a screen recorder is setup to record the experimental arrangement. Initially the algorithm is executed in MATLAB and the vehicle launched along a pre-defined trajectory. The robot is assigned to travel through two waypoints 2 m apart steered autonomously with the help of the QLDP's collision-free motor commands interfaced through an Arduino Nano microcontroller. The path of the vehicle is impeded with two obstacles, 0.15 m and 0.2 m high, laid 0.7 m apart. The robot moves with a constant forward velocity of 1.5 ms^{-1} traversing the assigned 2 m distance in 1.33 seconds. However, this mission is accomplished without a collision at a success rate of $\sim$90% shown in Figure (a) 6.5. The robot's field of view is bifurcated and processed to create a comparison between either directions (left-right) facilitating collision-free motor control command generation.



(a) Blended frames of the top-view movie captured while robot travels the trajectory

(b) Robot's field of view where random obstacles are detected and avoided

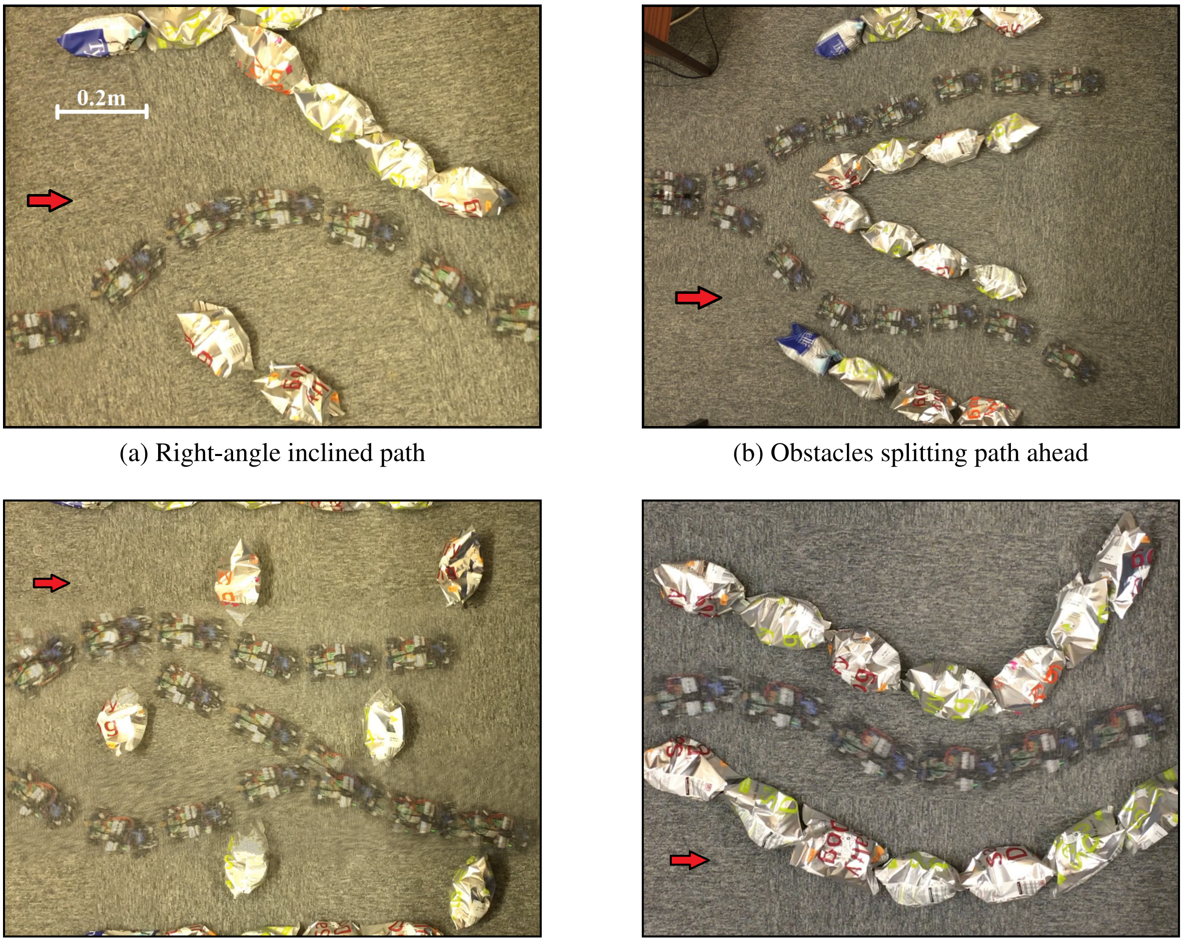
(c) Spiking response of the algorithm with respect to potential collision threats

Figure 6.5: Real World Experimentation.

The spiking response of the processed images for either half of robot's field of view is illustrated in Figure (c) 6.5. The entire process of consecutive obstacle avoidance is completed in 20 frames (0.6 seconds), initiated at frame-20 and completed by frame-40. Substituting the equations 6.1 and 6.2 with, initial obstacle-robot separation ($I =$1m), frame number at which the avoidance is initiated ($x =$20$^{\text{th}}$)(Figure (c) 6.5), velocity of the robot ($v =$1.5ms^{-1}), processing frequency ($f =$90Hz), we obtain the proposed model's capability to avoid arbitrary real-world obstacles at 670 millimetres prior to collision.

6.2.2 Arbitrary Standard Collision Scenarios

In order to realise the ultimate objective of our proposed algorithm, an arbitrary set of conventional collision scenarios were orchestrated. Figure 6.6 illustrates four individual scenarios tested for various trials indicating successful collision-free manoeuvring of the vehicle through obstacles laid in eclectic formations. For these tests, the robot is assigned to traverse a straight trajectory connecting two waypoints from extreme left to extreme right hand side. The velocity of the vehicle remains constant at $\sim$1 ms^{-1} travelling an approximately 1.5 m distance crammed with 0.1 m high obstacles ($\sim$40 g crisps packets) whose flexible profile proved that the algorithm remains independent of obstacle contour, dimension, and colour. The illustrated test cases involve conventional scenarios and obstacle setup such as, right-angled path necessitating sharp manoeuvres, a 45° junction splitting trajectory, chequered-pattern-laid obstacles crammed to convolute the test, and ultimately the most challenging scenario considering the robot's underactuation (Ackerman steering) to make a steep turn demanding a correct sequence of consecutive motor control commands.



(a) Right-angle inclined path

(b) Obstacles splitting path ahead

(c) Obstacles laid in chequered pattern

(d) A challenging steep turn

Figure 6.6: Standard Collision Scenarios.

6.3 Performance Assessment

A number of tests were executed to obtain a detailed and comprehensive conclusion on the model's performance and capabilities. In order to draw valid inferences, a set of established reference parameters such as,

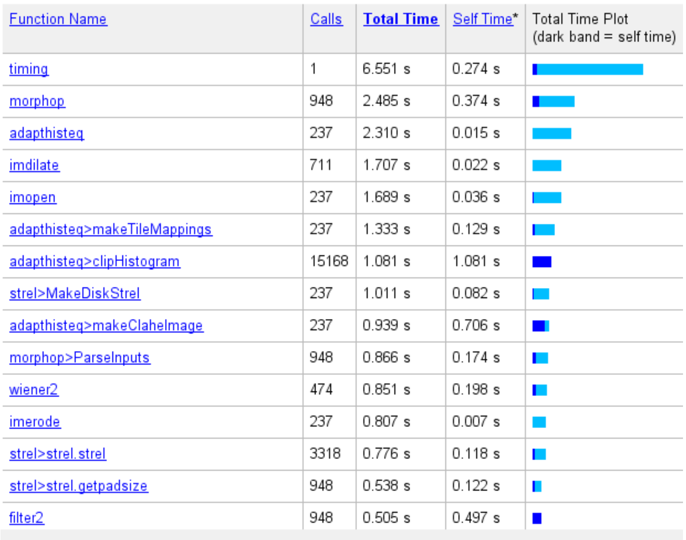
- Computation Load
- Detection Error
- Detection Time and Distance Prior to Collision

are systematically studied in this section to assess and validate performance of the proposed model in off-line as well as real-world real-time scenarios.

6.3.1 Computation Load

One of the most important performance assessment parameters in this thesis includes the algorithm's processing power and memory requirements, since the designed system's primary target applications are micro UAVs, it is crucial to assess the system's required processing power and memory. For the computation load assessment, all the visualisation elements such as response plot, playback object, and processed frames are eliminated to minimise unnecessary load.

Timing the model execution in MATLAB with the specifications described in 6.1, deduced the Figure 6.7 as the time distribution chart for each function. The timing chart clearly indicates the total required time to process an eight seconds video, with the 'contrast and illumination correction' layers consuming the greatest portions of processing time which leads to elimination of illumination-correction layer to further increase efficiency of the model.



Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
timing	1	6.551 s	0.274 s	
morphop	948	2.485 s	0.374 s	
adapthisteq	237	2.310 s	0.015 s	
imdilate	711	1.707 s	0.022 s	
imopen	237	1.689 s	0.036 s	
adapthisteq>makeTileMappings	237	1.333 s	0.129 s	
adapthisteq>clipHistogram	15168	1.081 s	1.081 s	
strel>MakeDiskStrel	237	1.011 s	0.082 s	
adapthisteq>makeClaheImage	237	0.939 s	0.706 s	
morphop>ParseInputs	948	0.866 s	0.174 s	
wiener2	474	0.851 s	0.198 s	
imerode	237	0.807 s	0.007 s	
strel>strel.strel	3318	0.776 s	0.118 s	
strel>strel.getpadsize	948	0.538 s	0.122 s	
filter2	948	0.505 s	0.497 s	

Figure 6.7: MATLAB-Program Timing Chart

An additional load test was performed using MATLAB's built-in stopwatch to time every individual processing layer of the algorithm deducing a time distribution table and a pie chart illustrated in Figure 6.8.

Time to Process 1 Frame (milliseconds)	
Noise Filter	4
Contrast Correction	3
Luminance Disparity	0.4
Binary Thresholding	0.3
Others	0.8

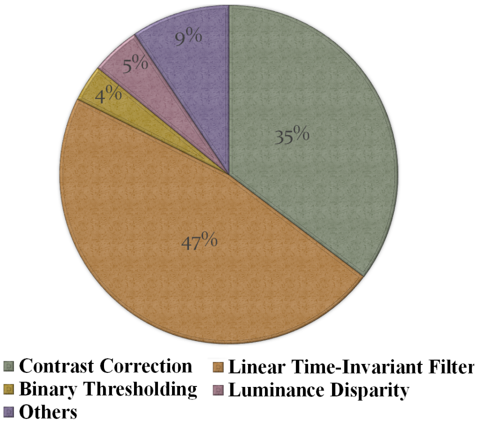
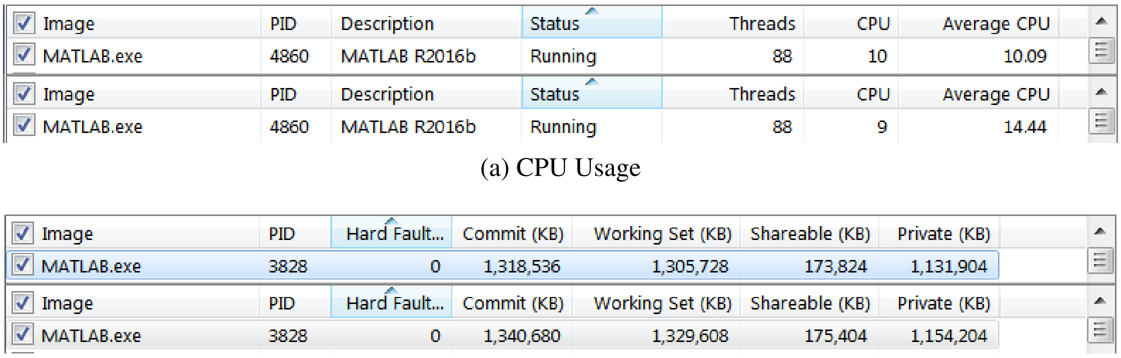


Figure 6.8: The table lists epoch for different layer processing of 1 image frame, and the pie chart illustrates the graphical distribution of computation load

It is evident that on average the total required time to process a single image frame of 320 × 240 pixels is ∼8.5 milliseconds, which implies that a maximum of 120 Hz processing speed can be achieved that is highly superior compared to the contemporary associated literature. However, greater frequencies are feasible by eliminating processing layers with a slightly lesser significance such as contrast correction, which introduces a rather potential future research topic to conduct a prudent performance trade-off by analysing these layers' significance with respect to their computation cost.



☑ Image	PID	Description	Status	Threads	CPU	Average CPU
☑ MATLAB.exe	4860	MATLAB R2016b	Running	88	10	10.09

☑ Image	PID	Description	Status	Threads	CPU	Average CPU
☑ MATLAB.exe	4860	MATLAB R2016b	Running	88	9	14.44

(a) CPU Usage

☑ Image	PID	Hard Fault...	Commit (KB)	Working Set (KB)	Shareable (KB)	Private (KB)
☑ MATLAB.exe	3828	0	1,318,536	1,305,728	173,824	1,131,904

☑ Image	PID	Hard Fault...	Commit (KB)	Working Set (KB)	Shareable (KB)	Private (KB)
☑ MATLAB.exe	3828	0	1,340,680	1,329,608	175,404	1,154,204

(b) RAM Usage

Figure 6.9: MS Windows Resource Monitor

Nevertheless, the test setup configuration presented here, is substantially beyond the model's actual requirements as the MS Windows Resource Monitor shown in Figure 6.9 demonstrates usage of resources being infinitesimally small, which can be briefly listed as,

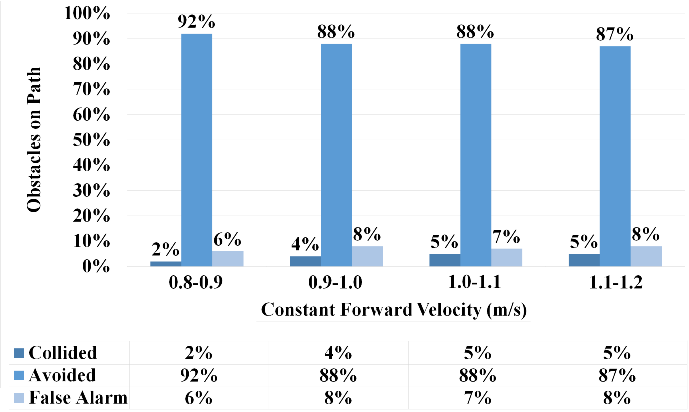
1. Percentage CPU Usage = 4.3% of Core i5 3.20-GHz
2. Percentage Used Physical Memory = 0.25% (21 MB)
3. Storage Disk Usage = 4.5 KB

These requirements hint at the fact that the proposed model is computationally simple, hence feasible to be successfully implemented and executed on a basic single core ARM microprocessor.

6.3.2 Detection Errors

The proposed model was analysed for its consistency and performance robustness by testing it against a number of different scenarios involving static obstacles of sizes within a range of 0.5 to 1 m wide sequentially obstructing the robot's path which is moving at an approximate forward velocity of 0.8 to 1.2 ms^{-1}.

To draw a clear inference in this section, a graph of the percentage of obstacles; missed (collided), avoided, and false detections were plotted against the robot's (observer) constant velocity in ms^{-1} respectively, shown in Figure 6.10. A detailed assessment result for different off-line scenarios is illustrated in Figure 6.12. The model remains highly effective at lower (0.10-0.20 ms^{-1}) velocities, although efficiency drops causing slight rise (2%) in generation of false collision alarms at higher forward velocities (1-1.2 ms^{-1}), yet it remains highly feasible for indoor applications since robots do not operate at elevated velocities within confined environments. Also considering the model's computational simplicity, the delivered efficiency is reasonably high compared to the literature reported in the associated research (Yue & Rind 2013, Hu et al. 2016, Silva et al. 2014).



	0.8-0.9	0.9-1.0	1.0-1.1	1.1-1.2
■ Collided	2%	4%	5%	5%
■ Avoided	92%	88%	88%	87%
■ False Alarm	6%	8%	7%	8%

Figure 6.10: System Detection Errors

6.3.3 Distance and Time to Collision

One of the most common reference parameters for examining the performance of a collision avoidance algorithm is the distance and time before which a potential collision threat is successfully detected and avoided. This is particularly important for aerial applications as flying robots exhibit underactuation that requires quicker detections providing sufficient time to initiate successful avoidance manoeuvres. The related contemporary research works provide performance assessments of LGMD model on ground vehicles which we compare with the capabilities and performance of the model proposed in this thesis.

These assessments involve the study of results from Hu et al. (2016), Yue & Rind (2013) and Silva et al. (2014), which are slightly contrasting, as they implement different approaches to summarise their results by demonstrating that an increase in vehicle velocity results in greater distances before which an avoidance manoeuvre can be initiated. However, the constant looming velocities involved in these experiments does not quite suffice aerial applications where the vehicle velocity exceeds a minimum of 1 ms^{-1} most of the time, and due to their underactuation, the distance prior to collision must be much greater than ground vehicles to accommodate safe manoeuvres. Nevertheless, their rate of correct detection remains higher (95%) due to the lower forward velocities used in their experimentation.

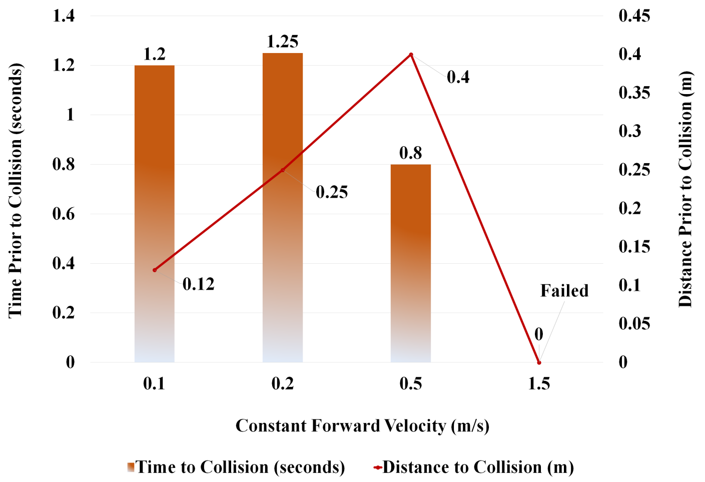


Figure 6.11: Detection Distance and Time Prior to Collision (model from Hu et al. (2016))

On the other hand, Silva et al. (2014) present their model performance based on four simulated collision scenarios and one real-world recorded video involving approach of a rectangular box at constant velocities in the range of 1 ms^{-1} whose distance prior to collision is estimated to remain between 0.25 to 0.35 m with minimum consistency caused by the simulation methodologies and environment. However, for the offline real-world video analysis involving a dark ball on a white background, the looming velocity implemented is much smaller ($\sim$0.24 ms^{-1}) where the detection distance prior to collision still remains approximately the same as simulated results (0.2 meters).

Further to draw a contrast with respect to the related methodologies proposed by different authors, the collected input dataset was tested with a collision avoidance model with inhibition (main processing layer in LGMD) as described in Hu et al. (2016), Yue & Rind (2013) to deduce a set of results illustrated in Figure 6.11. In line with their published results, the time to collision in these models remain fairly greater than 1 second when compared to our >3 seconds response shown in Figure 6.5 and 6.12. Hence the distance prior to collision is almost 1/3rd of the corresponding results deduced in our proposed model for the same scenario and settings explained in detail below. Similarly, the behaviour of the system varies inconsistently as well, where the rise in forward velocity causes increase in distance prior to collision slightly up to an approximately 0.5 ms^{-1}, however, at higher velocities (>1 ms^{-1}) the model fails to cope as a result of increasing false collision alarms (50%).

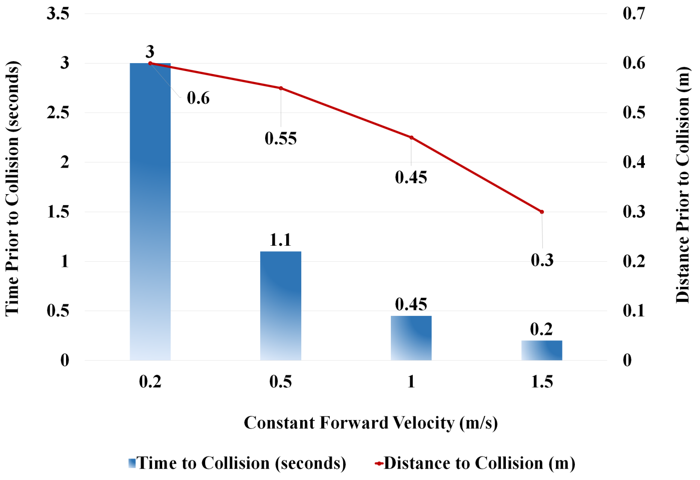


Figure 6.12: Detection Distance and Time.

Now to correlate the assessment results of the proposed model with the above references, we approximate the growth of model's spiking response to a collision threat by an exponential curve whose slope directly depends on, (1) processing power, (2) memory, (3) data acquisition frequency (Hz), (4) robot's constant forward velocity, and (5) obstacle's distance from the robot. To establish testing standards, the above specifications are approximated to the nearest values with respect to the associated research, and tests are performed at near-exact laboratory conditions. The distance (D) and time (T) at which the model successfully detects a potential threat prior to collision and fires spiking response is computed using the robot's constant forward velocity, initial obstacle distance from vehicle, and the frame at which the first avoidance command is generated, expressed as,

$$D = I - (x.v.f^{-1}) \tag{6.1}$$

$$T = D.v^{-1} \tag{6.2}$$

where, I is the initial obstacle distance from robot at 0th frame, v is the robot's constant forward velocity, f is the data acquisition frequency, and $\times$ is the frame number at which avoidance is initiated. Using the above relations, time and distances were computed and plotted on a graph of robot's constant velocity illustrated in Figure 6.12,

Avoidance manoeuvres are initiated prior to at least five times the detected obstacle's size providing sufficient distance and time for a successful avoidance manoeuvre which is extremely favourable in aerial robotics where underactuation is a significant concern. Although the manoeuvring and control command generation is performed on an entirely different principle when compared to the methodologies implemented by the afore described associated research, nevertheless, required objectives are achieved successfully as Figure 6.12 and 6.5 represent the time before collision for higher forward velocities as (1) 500 mm for off-line tests, and (2) 670 mm for real-time real-world tests, which evidently bolsters the briskness of the proposed model for indoor aerial collision avoidance. Hence it can be claimed that using this system, flight control commands may be generated well prior to the occurrence of a collision providing secure path for the vehicle with an infinitesimally small computation load.

6.4 Summary

This chapter successfully describes a comprehensive and systematic approach to a brief overall assessment of the proposed model's performance. The results demonstrated in this chapter provide sufficient evidence to support the model's following list of claimed capabilities to successfully detect and avoid obstacles in real-time real-world conditions. The algorithm is fairly advanced and robust compared to its peer models addressing a number of indoor robot navigation-specific challenges highlighted as:

- Brisk response ($\sim$700 mm prior to collision at 1.5 ms^{-1})
- Irregular contrast (lighting) correction
- Motion direction detection
- Simplified proximity estimation
- Minimal computation load
- Optimised performance (elimination of insignificant processing nodes)

7

Conclusions and Future Work

> *"One machine can do the work of fifty ordinary men.*
> *No machine can do the work of one extraordinary man."*
> – Elbert Hubbard

7.1 Salient Achievements

This thesis presents a novel methodology and its performance assessment to achieve a computationally efficient and robust collision avoidance system for micro and nano class of robots. This is primarily aimed at assisting the development of autonomous navigation systems requiring minimum computation power that is a challenging constraint in compact robots. A biologically-inspired vision based collision avoidance system was studied in Chapter 4 which inspired an original algorithm coined as the Quadfurcated Luminance-Difference Processing (QLDP), modulated and enhanced (Chapter 5) during the course of this research, whose performance assessment results inferred, (1) QLDP was capable of achieving an obstacle detection time of more than 500 milliseconds prior to collision in off-line testing scenarios (Chapter 6), and (2) it was capable of achieving an obstacle detection distance of more than 600 mm prior to a collision in Real-world real-time testing scenarios (Chapter 6), processing at 120 Hz which is much superior compared to its peer models reported in the contemporary literature. Major detection challenges that were successfully addressed in this thesis are,

- Brisk response

- Edge detection in complex background
- Operation in GPS-denied environments
- Irregular contrast (lighting) correction
- Motion direction detection
- Proximity estimation and
- Optimised performance and minimal computation load

Over the past few decades, biological approaches to address engineering problems have evolved continuously gaining momentum rapidly. This is because of the simple yet robust structure of the biological beings we observe in nature. For instance, optic flow algorithms used in autonomous navigation of aerial robots are far more efficient when compared to the conventional computer vision algorithms that require extensive computational power to perform the same task. Similarly, in this thesis, the simple luminance-differencing behaviour of a *locust* visual neuron (LGMD) lays the foundation of the proposed algorithm to address challenging tasks such as edge detection in complex backgrounds. Such efficient solutions offered by the biological models are particularly favourable in micro and nano robotics whose payload delivery is strictly limited due to underdeveloped hardware technologies.

In line with the simplicity and efficiency of biologically-inspired algorithms, vision-based solutions on the hardware end, offer far better prospects in further simplifying the overall unmanned autonomous system (UAS) as they are much lighter in weight, operate on a broader bandwidth, require minimal power, and are compact in size when compared to their peer hardware technologies such as LIDARS. Therefore it is evident that vision-based technologies would emerge as primary preference for robotic researchers similar to the evolution of biological beings implementing vision as their primary navigation sensory organ.

Although the methodology proposed in this thesis addresses only a see-and-avoid aspect of a navigation system, it contributes crucially towards elevation of autonomy level in a UAS applied on micro/nano robots operating in a complex GPS-denied environment. Augmenting the autonomy level of autonomous system is a vast and trending research topic particularly in aerial robotics since their applications are expanding rapidly including, inspection and maintenance of infrastructures, agricultural seeding, imaging, pest-management, transportation and logistics, search and rescue organisations, disaster management, surveillance, military and many other fields. UAS are particularly suitable for the afore mentioned tasks due to the fact that they eliminate the requirement of direct human interaction and supervision thus greater the autonomy levels, greater would be the mission success rate at reduced cost.

7.2 Limitations

Although the proposed algorithm performs admirably better than its peer models, there exists major limitations to this methodology on different grounds that could perhaps unveil a range of potential future research topics to achieve further efficiency and robustness. Some of these important limitations worth perceiving are explained systematically below.

7.2.1 Environmental Conditions

Autonomous navigation in a complex cluttered environment is an arduous task that is currently possible to address only by computationally expensive algorithms capable of differentiating, tracking and identifying the nature, origin, and type of a detected obstacle which is not facilitated in simpler algorithms such as the QLDP presented in this thesis. However, the model proposed here detects edges and establishes a correlation among segments of its field of view to differentiate a distant from an imminent object or a receding from an approaching one simply by computing the nature of expansion/contraction of the detected object edge. Thus further development of the edge detection processing layer within this model may improve the performance drastically.

There exists many other operation-environment limitations to our proposed algorithm, such as extremely dark, rainy, misty, poorly-visible, and basically any possible environmental condition that impedes the operation of a visual sensor (camera) limits the successful implementation and operation of the algorithm proposed in this thesis.

7.2.2 Forward Velocity

One of the fundamental constraints particularly in aerial robotic navigation is operating successfully at high-speed flights and extreme accelerations due to powerful propulsion systems capable of producing a vertical lift against the gravitational force. To perform successful collision avoidance at higher approach velocities, the algorithm must be capable of processing at higher frequencies likewise. This is achieved by optimising and increasing efficiency of the system exhaustively, however, due to hardware limitations and computational power constraints onboard micro robots, high speed processing at high success rate is currently almost impossible. Similarly, the algorithm proposed in this thesis does not serve its purpose successfully at extreme looming velocities such as post 2 ms^{-1} as the number of false collision alarms increase exponentially, demonstrated in Figure 6.12.

7.2.3 Planning and Mapping

As mentioned earlier, the methodology proposed in this thesis is based on see-and-avoid principle involving only a sensing and reacting elements to avoid obstacles across its field of view. These reactive systems may not satisfy the computation intensive processes such as environmental mapping and planning optimised trajectories to reach its destination. Mapping and planning are two of the fundamental elements of robotic navigation particularly in complex cluttered environments where the need for perceiving and comprehending details of the surrounding is crucial to traverse and plan the most optimised trajectories. However, as mentioned earlier, the contemporary methodologies executing these processes are computationally expensive demanding powerful processors onboard larger robots, hence, not facilitated in the QLDP presented in this thesis.

7.2.4 Dynamic Obstacle Tracking

This is yet another advantageous feature possessed by sophisticated processing algorithms capable of tracking moving (dynamic) objects to map and plan an optimised safe trajectory far ahead. Contemporary tracking algorithms such as Lucas-Kanade-Tomasi, Kalman Filtering are computation expensive and require dedicated processing power that is not available onboard micro/nano aerial robots. However, designing and incorporating a simple tracking algorithm to increase the proposed model's robustness is a potential area of development as future work, this may be achieved by sensor-fusion and computation coupling, dedicating an individual processing module to track and anticipate future positions of the detected dynamic obstacles on collision course.

7.3 Future Work

The ultimate goal of an unmanned autonomous system is to accomplish its assigned tasks at highest possible success rate absolutely independent of human supervision. Contemporary automation technologies are capable of performing basic tasks autonomously, nevertheless complex tasks and missions particularly in aerial robotics, such as take-off, landing, and noncooperative collision avoidance are yet to be executed by UAVs completely autonomously. Therefore, investigating the methodologies to perform such missions are blazing topics of ongoing research benefiting from a large capital and resource investment.

For the future developments of the model particularly proposed in this thesis, addressing all the afore mentioned limitations without compromising computational simplicity is highly recommended. Specifically, design and integration of an advanced image segmentation methodology (that is currently based on luminance-difference and thresholding) may facilitate successful operation in high-speed flights through complex cluttered environments. Introduction of complementary modules to process dynamic obstacle tracking is another possible advancement to achieve greater robustness. Incorporation of a simple optimised trajectory generator fused with the tracking data may facilitate a proactive collision avoidance, thus greater system autonomy.

Man-made autonomous systems or any technological advancement in general, are far inferior to their biological counterparts in a number of different aspects; efficiency, agility, compactness, ergonomics, resilience, and robustness. Such biological superiority is self explanatory compelling the researchers to focus further on such novel solutions and stimulate the future research to accomplish the same.

7.4 Concluding Remarks

This thesis successfully demonstrates a secondary vision-based collision avoidance system for micro aerial vehicles, whose performance is validated against an eclectic set of off-line (recorded) as well as real-world real-time collision scenarios. Contribution of every individual processing layer is illustrated to bolster the claimed capabilities of the system such as, brisk response, irregular contrast (lighting) correction, direction and proximity estimation, and optimised performance and minimal computation load. Ultimately, the performance assessment results validate the proposed model's capability to detect obstacles at more than 670 mm (real-world) prior to collision, pacing at 1.2 ms^{-1} with a successful avoidance rate of greater than 90% processing at 120 Hz independent of obstacle colour, dimension, and contour, sufficient to conclude the system success to effectively avoid obstacles in real-time and real-world robotic applications.

Bibliography

Abbeel, P., Coates, A., Quigley, M. & Y. Ng, A. (2006), An application of reinforcement learning to aerobatic helicopter flight, *in* 'Neural Information Processing Systems', pp. 1–8.

Alturbeh, H. & Whidborne, J. F. (2014), Real-time obstacle collision avoidance for fixed wing aircraft using B-splines, *in* 'Proc. of UKACC Int. Conf. on Control', Loughborough, pp. 115–120.

Andreou, A. G., Boahen, K. A., Pouliquen, P. O., Pavasovic, A., Jenkins, R. E. & Strohbehn, K. (1991), 'Current mode subthreshold MOS circuits for analogue VLSI neural systems', *IEEE Trans. Neural Networks* pp. 205–213.

Armstrong, J. D. & van Hemert, J. I. (2009), 'Towards a virtual fly brain', *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **367**(1896), 2387–2397.
URL: *http://rsta.royalsocietypublishing.org/cgi/doi/10.1098/rsta.2008.0308*

Atkeson, C. G. (1993), Using local trajectory optimizers to speed up global optimization in dynamic programming, *in* 'Proc. Neural Information Processing Systems', Denver, pp. 663–671.

Atkeson, C. G. & Morimoto, J. (2003), 'Nonparametric representation of policies and value functions: a trajectory-based approach', *J. Advances in Neural Information Processing Systems* **15**, 1611–1618.

Austin, R. (2010), *Unmanned Aircraft Systems*, Wiley.

Bachrach, A. G. (2013), Trajectory bundle estimation for perception-driven planning, Phd thesis, Massachusetts Institute of Technology.

Barber, D. B., Griffiths, S. R., McLain, T. W. & Beard, R. W. (2005), 'Autonomous landing of miniature aerial vehicles', *Proc. AIAA Infotech@Aerospace* **4**(5), 770–784.

Barry, A. J. (2012), Flying between obstacles with an autonomous knife-edge maneuver, Ms thesis, Massachusetts Institute of Technology.

Barry, A. J. (2014), Flying between obstacles with an autonomous knife-edge maneuver, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', p. 2559.

Barry, A. J. (2016), High-speed autonomous obstacle avoidance with pushbroom stereo, Phd thesis, Massachusetts Institute of Technology.

Bates, D. & Postlethwaite, I. (2002), *Robust multivariable control of aerospace systems*, Delft University Press.

Bemporad, A. & Morari, M. (1999), Robust model predictive control: a survey, *in* 'Lecture Notes in Control and Information Sciences', pp. 207–226.

Beyeler, A., Zufferey, J. & Floreano, D. (2009*a*), optiPilot: control of take-off and landing using optic flow, *in* 'Proc. European Micro Air Vehicle conference and competition'.

Beyeler, A., Zufferey, J. & Floreano, D. (2009*b*), 'Vision-based control of near-obstacle flight', *J. Autonomous Robots* **27**(3), 201–219.

Blanchard, J. M. (1998), Collision avoidance: a biologically inspired neural network for the detection of approaching objects, Phd thesis, Newcastle University.

Blanchard, M., Rind, F. C. & Verschure, P. F. M. J. (2000), 'Collision avoidance using a model of the locust LGMD neuron', *J. Robotics and Autonomous Systems* **30**(1), 17–38.

Boahen, K. A. (1998), Communicating neuronal ensembles between neuromorphic chips, *in* 'Neuromorphic Systems Engineering', pp. 229–259.

Bock, H. G. & Plitt, K. J. (1984), A multiple shooting algorithm for direct solution of optimal control problems, *in* 'Proc. IFAC World Congress Budapest', pp. 243–247.

Burtt, E. T. & Catton, W. T. (1952), 'Nerve impulses originating from the compound eye of the locust', *J. Nature* **170**, 285.

Byrne, J., Cosgrove, M. & Mehra, R. (2006), Stereo based obstacle detection for an unmanned air vehicle, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 2830–2835.

Chowdhary, G., Johnson, E. N., Magree, D. & Wu, A. (2013), 'GPS-denied indoor and outdoor monocular vision aided navigation and control of unmanned aircraft', *J. Field Robotics* **30**(3), 415–438.

Cory, R. (2010), Supermaneuverable Perching, Phd thesis, Massachusetts Institute of Technology.

Cory, R. & Tedrake, R. (2008), Experiments in fixed-wing UAV perching, *in* 'Proc. AIAA Guidance, Navigation, and Control Conf.', pp. 1–12.

Cowling, I. D., Whidborne, J. F. & Cooke, A. K. (2001), 'Optimal trajectory planning and LQR control for a quadrotor UAV', *J. Automatica* **37**, 1057–1064.

Cowling, I. D., Yakimenko, O. a., Whidborne, J. F. & Cooke, A. K. (2007), A prototype of an autonomous controller for a quadrotor UAV, *in* 'Proc. of European Control Conference', pp. 1–8.

Cuadri, J., Linan, G., Stafford, R., Keil, M. S. & Roca, E. (2005), A bioinspired collision detection algorithm for VLSI implementation, *in* 'Proc. SPIE Conf. Bioengineered and Bioinspired System', Vol. 5839, pp. 238–248.

Davison, A. J. (2007), 'MonoSLAM: real-time single camera SLAM', *IEEE Trans. Pattern Analysis and Machine Intelligence* **29**(6), 1052–1067.

Dey, D., Shankar, K. S., Zeng, S., Mehta, R., Agcayazi, M. T., Eriksen, C., Daftry, S., Hebert, M. & Bagnell, J. A. (2015), 'Vision and learning for deliberative monocular cluttered flight', *J. Field and Service Robotics* pp. 392–400.

Diehl, M., Bock, H. G., Diedam, H. & Wieber, P.-B. (2006), Fast direct multiple shooting algorithms for optimal robot control, *in* 'Fast Motions in Biomechanics and Robotics', Springer, pp. 65–93.

Eggers, A. & Gewecke, M. (1993), The dorsal rim area of the compound eye and polarization vision in the desert locust (Schistecerca Gregaria), *in* 'Sensory Systems of Arthropods', pp. 101–109.

Eichner, H. (2011), Internal structure of the fly elementary motion detector, Phd thesis, Ludwig–Maximilians–Universität München.

Engel, J., Sturm, J. & Cremers, D. (2012), Camera-based navigation of a low-cost quadrocopter, *in* 'Proc. IEEE Int. Conf. Intelligent Robots and Systems', pp. 2815–2821.

Etienne-Cummings, R., Spiegel, J. & Mueller, P. (1996), ‘A visual smooth pursuit tracking chip’, *J. Advances in Neural Information Processing Systems* **8**, 706–712.

Etienne-Cummings, R., Spiegel, J., Mueller, P. & Zhang, M. Z. (2000), ‘A foveated silicon retina for two-dimensional tracking’, *IEEE Trans. Circuit and Systems* **47**, 504–517.

Floreano, D., Zufferey, J. & Klaptocz, A. (2011), Aerial locomotion in cluttered environments, *in* ‘Proc. Int. Symp. Robotics Research’, pp. 1–19.

Forster, C., Pizzoli, M. & Scaramuzza, D. (2014), SVO: fast semi-direct monocular visual odometry, *in* ‘Proc. IEEE Int. Conf. Robotics and Automation’, pp. 15–22.

Frazzoli, E., Dahleh, M. A. & Feron, E. (2000), Autonomous vehicle motion planning, *in* ‘Proc. IEEE Conf. Decision and Control’, pp. 821–826.

Garcia, C. E., Prett, D. M. & Morari, M. (1989), ‘Model predictive control: theory and practice a survey’, *J. Automatica* **25**(3), 335–348.

Gavrilets, V. (2002), Control logic for automated aerobatic flight of a miniature helicopter, *in* ‘Proc. AIAA Guidance, Navigation and Control Conf.’, pp. 1–8.

Gewecke, M. & Hou, T. (1993), Visual brain neurons in locusta migratoria, *in* ‘Sensory Systems of Arthropods’, pp. 119–144.

Goldberg, S. B. & Matthies, L. (2011), Stereo and IMU assisted visual odometry on an OMAP3530 for small robots, *in* ‘Proc. IEEE Computer Vision and Pattern Recognition Workshops’, pp. 169–176.

Harris, C. G. & Pike, J. M. (1988), ‘3D positional integration from image sequences’, *J. Image and Vision Computing* **6**(2), 87–90.

Harrison, R. R. (2005), ‘A biologically inspired analog IC for visual collision detection’, *IEEE Trans. Circuits and Systems* **52**, 2308–2318.

Harrison, R. R. & Koch, C. (1998), ‘An analog VLSI model of the fly elementary motion detector’, *J. Advances in Neural Information Processing Systems* **10**, 880–886.

Hatsopoulos, N., Gabbiani, F. & Laurent, G. (1995), ‘The locust DCMD, a movement-detecting neurone tightly tuned to collision trajectories’, *J. Science* **270**, 1000–1003.

Higgins, C. M. & Koch, C. (1999), Multi-chip neuromorphic motion processing, *in* 'Proc. Conf. Advanced Research in VLSI', Atlanta.

Higgins, C. M. & Pant, V. (2004), 'A biomimetic VLSI sensor for visual tracking of small moving targets', *IEEE Trans. Circuits and Systems* **51**(12), 2384–2394.

Higgins, C. M., Pant, V. & Deutschmann, R. (2005), 'Analog VLSI implementation of spatio-temporal frequency tuned visual motion algorithms', *IEEE Trans. Circuits and Systems* **52**(3), 489–502.

Higgins, C. M. & Shams, S. A. (2002), 'A biologically-inspired modular VLSI system for visual measurement of self-motion', *IEEE Trans. Sensors* **2**(6), 508–528.

Honegger, D. (2012), Real-time velocity estimation based on optical flow and disparity matching, *in* 'Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems', pp. 5177–5182.

Honegger, D., Oleynikova, H. & Pollefeys, M. (2014), Real-time and low latency embedded computer vision hardware based on a combination of FPGA and mobile CPU, *in* 'Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems', Chicago, pp. 4930–4935.

Horridge, G. A. (1975), *The compound eye and vision of insects*, Clarendon Press.

Horridge, G. A. (1978), 'The separation of visual axes in apposition compound eyes', *Philosophical Trans. Royal Society of London* **285**(1003), 1–59.

Horridge, G. A. (1980), 'Invertebrate Vision', *J. Nature* **283**, 510–510.

Horridge, G. A., Duniec, J. & Mareelja, L. (1981), 'A 24-hour cycle in single locust and mantis photoreceptors', *J. Experimental Biology* **91**, 307–322.

Hrabar, S., Sukhatme, G. S., Corke, P., Usher, K. & Roberts, J. (2005), Combined optic-flow and stereo-based navigation of urban canyons for a UAV, *in* 'Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems', pp. 3309–3316.

Hu, C., Arvin, F., Xiong, C. & Yue, S. (2016), 'A bio-inspired embedded vision system for autonomous micro-robots: the LGMD Case', *IEEE Trans. Autonomous Mental Development* pp. 1–14.

Hutchings, T., Jeffryes, S. & Farmer, S. J. (2007), Architecting UAV sense & avoid systems, *in* 'Proc. IET Conf. Autonomous Systems', number 1, pp. 1–8.

Indiveri, G., Oswald, P. & Kramer, J. (2002), 'An adaptive visual tracking sensor with a hysteretic winner-take-all network', *IEEE Trans. Neural Networks* pp. 324–327.

James, A. C. & Osorio, D. (1996), 'Characterisation of columnar neurons and visual signal processing in the medulla of the locust optic lobe by system identification techniques', *J. Comparative Physiology* **178**, 183–199.

Johnson, A., Montgomery, J. & Matthies, L. (2005), Vision guided landing of an autonomous helicopter in hazardous terrain, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 4470–4475.

Judge, S. J. & Rind, F. C. (1997), 'The locust DCMD, a movement-detecting neurone tightly tuned to collision trajectories', *J. Experimental Biology* **200**, 2209–2216.

Kanade, T., Amidi, O. & Ke, Q. (2004), Real-time and 3D vision for autonomous small and micro air vehicles, *in* 'Proc. IEEE Int. Conf. Decision and Control', pp. 1655–1662.

Karaman, S. & Frazzoli, E. (2010), Incremental sampling-based algorithms for optimal motion planning, *in* 'Robotics: Science and Systems'.

Kavraki, L. E. (1996), 'Probabilistic roadmaps for path planning in high-dimensional configuration spaces', *IEEE Trans. Robotics and Automation* **12**(4), 566–580.

Kim, J. & Sukkarieh, S. (2003), Airborne simultaneous localisation and map building, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', Taipei, pp. 406–411.

Klein, G. & Murray, D. (2007), Parallel tracking and mapping for small AR workspaces, *in* 'Proc. IEEE/ACM Int. Symp. Mixed and Augmented Reality', pp. 225–234.

Krapp, H. G., Gabbiani, F., Koch, C. & Laurent, G. (1998), Neuronal multiplication in the locust visual system, *in* 'Proc. Göttingen Neurobiology Conf.', p. 410.

Kumar, N., Himmelbauer, W., Cauwenberghs, G. & Andreou, A. G. (1998), 'An analog VLSI chip with asynchronous interface for auditory feature extraction', *IEEE Trans. Circuit and Systems* pp. 600–606.

Lambrinos, D., Maris, M., Kobayashi, H., Labhart, T., Pfeifer, R. & Wehner, R. (1997), 'An autonomous agent navigating with a polarized light compass', *J. Adaptive Behaviour* **6**(1), 131–161.

Landolt, O., Mitros, A. & Koch, C. (2001), Visual sensor with resolution enhancement by mechanical vibrations, *in* 'Proc. Conf. Advance Research in VLSI', pp. 249–264.

Langelaan, J. W. (2006), State estimation for autonomous flight in cluttered environments, Phd thesis, Stanford University.

Langelaan, J. W. (2007), 'State estimation for autonomous flight in cluttered environments', *J. Guidance, Control, and Dynamics* **30**(5), 1414–1426.

Langelaan, J. W. & Rock, S. (2005), Towards autonomous UAV flight in forests, *in* 'Proc. AIAA Guidance, Navigation and Control Conf.', pp. 1–13.

LaValle, S. M. (1998), Rapidly-exploring random trees: a new tool for path planning, Technical report, Iowa State University, Dept. of Computer Science.

LaValle, S. M. (2006), *Planning Algorithms*, Cambridge University Press.

Lazzaro, J. & Wawrzynek, J. (1997), 'Speech recognition experiments with silicon auditory models', *J. Analog Integrated Circuits Process* **13**, 37–51.

Lee, G. H., Fraundorfer, F. & Pollefeys, M. (2011), MAV visual SLAM with plane constraint, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 3139–3144.

Lewis, F. L. & Syrmos, V. L. (1995), *Optimal Control.*

Li, M., Kim, B. H. & Mourikis, A. I. (2013), Real-time motion tracking on a cellphone using inertial sensing and a rolling-shutter camera, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', Karlsruhe, pp. 4712–4719.

Li, M. & Mourikis, A. I. (2013), 3-D motion estimation and online temporal calibration for camera-IMU systems, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', Karlsruhe, pp. 5709–5716.

Lim, J. S. (1990), 2-D adaptive noise-removal filtering, *in* 'Two-Dimensional Signal and Image Processing', Prentice Hall, Englewood Cliffs, NJ, p. 548.

Liu, S. C. (1996), Silicon model of motion adaptation in the fly visual system, *in* 'Proc. Caltech/UCSD Symp. Neural Computation', Pasadena.

Ljung, L. (2007), 'System identification toolbox for use with MATLAB'.

Mahowald, M. A. & Mead, C. (1991), 'The silicon retina', *J. Scientific American* **264**(5), 76–82.

Majumdar, A. & Tedrake, R. (2012), Robust online motion planning with regions of finite time invariance, *in* 'Proc. Workshop Algorithmic Foundations of Robotics', Cambridge, pp. 1–16.

Majumdar, A. & Tedrake, R. (2016), *Funnel libraries for real-time robust feedback motion planning*, MIT Press.

Marques, O. (2011), Binary morphological image processing, *in* 'Practical Image and Video Processing Using MATLAB'.

Mcfadyen, A. (2015), Visual control for automated aircraft collision avoidance systems, Phd, Queensland University of Technology.

Mcfadyen, A. & Mejias, L. (2015), 'A survey of autonomous vision-based see and avoid for unmanned aircraft systems', *J. Progress in Aerospace Sciences* **80**, 1–17.
URL: *http://dx.doi.org/10.1016/j.paerosci.2015.10.002*

Mead, C. (1989), *Analog VLSI and neural systems*, Addison-Wesley Longman, Boston.

Meier, L., Tanskanen, P., Heng, L., Hee, G., Friedrich, L. & Marc, F. (2012), 'PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision', *J. Autonomous Robots* **33**(2), 21–39.

Meinertzhagen, I. A. (1976), 'The organization of perpendicular fibre pathways in the insect optic lobe', *Philosophical Trans. Royal Society of London* **274**, 555–594.

Mellinger, D. & Kumar, V. (2011), Minimum snap trajectory generation and control for quadrotors, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 2520–2525.

Meng, H., Yue, S., Hunter, A., Appiah, K., Hobden, M., Priestley, N., Hobden, P. & Pettit, C. (2009), A modified neural network model for lobula giant movement detector with additional depth movement feature, *in* 'Proc. Int. Joint Conf. Neural Networks', pp. 2078–2083.

Michels, J., Saxena, A. & Ng, Y. A. (2004), High speed obstacle avoidance using monocular vision and reinforcement learning, *in* 'Proc. Int. Conf. Machine Learning', pp. 593–600.

Netter, T. & Franceschini, N. (2002), 'A robotic aircraft that follows terrain using a neuromorphic eye', *J. Intelligent Robots and Systems* **1**, 129–134.

Nicol, D. & Meinertzhagen, I. A. (1982), 'An analysis of the number and composition of the synaptic populations formed by photoreceptors of the fly', *J. Comparative Neurology* **207**, 29–44.

Nilsson, D. (1989), Optics and evolution of the compound eye, *in* 'Facets of Vision', pp. 30–73.

Nistér, D., Naroditsky, O. & Bergen, J. (2004), Visual odometry, *in* 'Proc. IEEE Conf. Computer Vision and Pattern Recognition', pp. 652–659.

Nowel, M. S. & Shelton, P. M. (1981), 'A golgi-electron microscopical study of the structure and development of the lamina ganglionaris of the locust optic lobe', *J. Cell and Tissue Research* **216**, 377–401.

O'Carroll, D. C., Osorio, D., James, A. C. & Bush, T. (1992), 'Local feedback mediated via amacrine cells in the insect optic lobe', *J. Comparative Physiology* **171**, 447–455.

Oleynikova, H., Honegger, D. & Pollefeys, M. (2015), Reactive avoidance using embedded stereo vision for MAV flight, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 50–56.

O'Shea, M. & Rowell, C. H. F. (1975), 'A spike-transmitting electrical synapse between visual interneurones in the locust movement detector system', *J. Comparative Physiology* **97**, 143–158.

O'Shea, M., Rowell, C. H. F. & Williams, J. L. D. (1974), 'The anatomy of a locust visual interneurone: the descending contralateral movement detector', *J. Experimental Biology* **60**, 1–12.

O'Shea, M. & Williams, J. L. D. (1974), 'The anatomy and output connection of a locust visual interneurone; the lobular giant movement detector (LGMD) neurone', *J. Comparative Physiology* **91**(3), 257–266.

Osorio, D. (1987), 'The temporal properties of non-linear, transient cells in the locust medulla', *J. Comparative Physiology* **161**, 431–440.

Osorio, D. (1992), Retinotopic vision in the locust, *in* 'Nonlinear vision: determination of neural receptive fields, function, and networks', pp. 377–390.

O'Sullivan, L., Corke, P. & Mahony, R. (2013), Image-based visual navigation for mobile robots, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 5277–5283.

Ozalevli, E. & Higgins, C. M. (2005), 'Reconfigurable biologically-inspired visual motion systems using modular neuromorphic VLSI chips', *IEEE Trans. Circuits and Systems* **52**(1), 79–92.

Pinter, R. B., Olberg, R. M. & Abrams, T. W. (1982), 'Is the locust DCMD, a looming detector?', *J. Experimental Biology* **101**, 327–331.

Rall, W. (1995), Perspective on neuron model complexity, *in* 'The Handbook of Brain Theory and Neural Networks', pp. 728–732.

Reichardt, W. & Egelhaaf, M. (1988), 'Properties of individual movement detectors as derived from behavioural experiments on the visual system of the fly', *J. Biological Cybernetics* **58**(5), 287–294.

Rind, F. C. (1984), 'A chemical synapse between two motion detecting neurones in the locust brain', *J. Experimental Biology* **110**, 143–167.

Rind, F. C. (1987), 'Non-directional, movement sensitive neurons of the locust optic lobe', *J. Comparative Physiology* **161**, 477–494.

Rind, F. C. & Bramwell, D. I. (1996), 'Neural network based on the input organization of an identified neuron signaling impending collision.', *J. Neurophysiology* **75**(3), 967–985.

Rind, F. C. & Simmons, P. J. (1999), 'Seeing what is coming: building collision-sensitive neurones', *J. Trends in Neurosciences* **22**(5), 215–220.

Ritz, R. (2012), Cooperative quadrocopter ball throwing and catching, *in* 'Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems', pp. 4972–4978.

Roberts, J. M., Corke, P. I. & Buskey, G. (2003), Low-cost flight control system for a small autonomous helicopter, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 546–551.

Roberts, R. (2012), Saliency detection and model-based tracking: a two part vision system for small robot navigation in forested environment, *in* 'Proc. SPIE', Vol. 8387, pp. 1–12.

Rowell, C. H. F. (1971), 'The opthopteran descending movement detector (DMD) neurones: a characterisation and review', *J. Zeitschrift für vergleichende Physioiogie* **73**, 167–194.

Rowell, C. H. F. & O'Shea, M. (1976*a*), 'The neuronal basis of a sensory analyser, the acridid movement detector system. I. effects of simple incremental and decremental stimuli in light and dark adapted animals', *J. Experimental Biology* **65**, 273–288.

Rowell, C. H. F. & O'Shea, M. (1976*b*), 'The neuronal basis of a sensory analyser, the acridid movement detector system. III. control of response amplitude by tonic lateral inhibition', *J. Experimental Biology* **65**, 617–625.

Rowell, C. H. F., O'Shea, M. & Williams, J. L. D. (1977), 'The neuronal basis of a sensory analyser, the acridid movement detector system', *J. Experimental Biology* **68**, 157–185.

Sa, I. (2013), Monocular vision based autonomous navigation for a cost-effective MAV in GPS-denied environments, *in* 'Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics', pp. 1355–1360.

Saripalli, S., Montgomery, J. F. & Sukhatme, G. S. (2002), Vision-based autonomous landing of an unmanned aerial vehicle, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 2799–2804.

Sarpeshkar, R., Kramer, J., Indiveri, G. & Koch, C. (1996), Analog VLSI architectures for motion processing: from fundamental limits to system applications, *in* 'Proc. IEEE Computational & Neural System Program', pp. 969–987.

Scaramuzza, D. & Fraundorfer, F. (2011), 'Visual odometry', *IEEE Robotics & Automation Magazine* (4), 80–92.

Scherer, S. (2007), Flying fast and low among obstacles, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 2023–2029.

Schlotterer, G. R. (1977), 'Response of the locust descending contralateral movement detector neuron to rapidly approaching and withdrawing visual stimuli', *Canadian Journal of Zoology* **55**, 1372–1376.

Scholes, J. (1965), Discontinuity of the excitation process in locust visual cells, *in* 'Proc. Cold Spring Harbor Symp. on Quantitative Biology', pp. 517–527.

Shamma, J. S. & Cloutier, J. R. (1993), 'Gain-scheduled missile autopilot design using linear parameter varying transformation', *J. Guidance, Control, and Dynamics* **16**(2), 256–263.

Shaw, S. R. (1978), 'The extracellular space and blood-eye barrier in an insect retina: an ultrastructural study', *J. Cell and Tissue Research* **188**, 35–61.

Shaw, S. R. (1984), 'Early visual processing in insects', *J. Experimental Biology* **112**, 225–251.

Shen, S., Mulgaonkar, Y., Michael, N. & Kumar, V. (2013), Vision-based state estimation for autonomous rotorcraft MAVs in complex environments, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 1758–1764.

Silva, A. C., Silva, J. & Santos, C. P. (2014), A modified LGMD based neural network for automatic collision detection, *in* 'Informatics in Control, Automation and Robotics', Vol. 283, pp. 217–233.

Sim, R., Elinas, P., Griffin, M. & Little, J. J. (2005), Vision-based SLAM using the rao-blackwellised particle filter, *in* 'Proc. IJCAI Workshop Reasoning with Uncertainty in Robotics', Vol. 14, pp. 9–16.

Simmons, P. (1980), 'Connexions between a movement-detecting visual interneurone and flight motoneurones of a locust', *J. Experimental Biology* **86**, 87–97.

Singh, L. & Fuller, J. (2001), Trajectory generation for a UAV in urban terrain, using nonlinear MPC, *in* 'Proc. IEEE American Control Conf.', pp. 2301–2308.

Single, S. & Borst, A. (1998), 'Dendritic integration and its role in computing image velocity', *J. Science* **281**(5384), 1848–1850.

Sira-Ramirez, H. & Agrawal, S. K. (2004), *Differentially Flat Systems*, illustrate edn, CRC Press.

Sobolic, F. M. (2009), Agile flight control techniques for a fixed-wing aircraft, Ms thesis, Massachusetts Institute of Technology.

Sourakov, A. (2011), 'Faster than a flash: the fastest visual startle reflex response is found in a long-legged fly, condylostylus sp. (Dolichopodidae)', *J. Florida Entomologist* **94**(2), 367–369.
URL: *http://dx.doi.org/10.1653/024.094.0240*

Srinivasan, M. V. (2004), 'An overview of insect-inspired guidance for application in ground and airborne platforms', *J. Aerospace Engineering* **218**(6), 375–388.

Stolle, M. & Atkeson, C. G. (2006), Policies based on trajectory libraries, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 3344–3349.

Strausfeld, N. J. (1976), *Atlas of an Insect Brain*, Springer-Verlag.

Strausfeld, N. J. & Campos-Ortega, J. A. (1977), 'Vision in insects: pathways possibly underlying neural adaptation and lateral inhibition', *J. Science* **195**, 894–897.

Strausfeld, N. J. & Gronenberg, W. (1990), 'Descending neurons supplying the neck and flight motor of diptera: organization and neuroanatomical relationships with visual pathways', *J. Comparative Neurology* **302**(4), 954–972.

Stryk, O. V. (1993), Numerical solution of optimal control problems by direct collocation, *in* 'Optimal Control', Vol. 111, pp. 129–143.

Svestka, P. & Overmars, M. H. (1997), 'Motion planning for car-like robots using a probabilistic learning approach', *J. Robotics Research* **16**(2), 119–143.

Sztarker, J. & Rind, F. C. (2014), 'A look into the cockpit of the developing locust: looming detectors and predator avoidance', *J. Developmental Neurobiology* **74**(11), 1078–1095.

Tedrake, R. (2009), LQR-trees: feedback motion planning on sparse randomized trees, *in* 'Proc. Robotics: Science and Systems', Seattle, pp. 1–8.

Tobenkin, M. M. (2014), Robustness analysis for identification and control of nonlinear systems, Phd thesis, Massachusetts Institute of Technology.

Tunstall, J. & Horridge, G. A. (1967), 'Electrophysiological investigation of the optics of the locust retina', *J. Zeitschrift für vergleichende Physioiogie* **55**, 167–182.

Wagter, C. D. (2014), Autonomous flight of a 20-gram flapping wing MAV with a 4-gram onboard stereo vision system, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', Hong Kong, pp. 4982–4987.

Wang, Y. & Boyd, S. P. (2008), Fast model predictive control using online optimization, *in* 'Proc. Int. Federation of Automatic Control World Congress', Vol. 17, pp. 6974–6979.

Webb, B. (1996), 'A cricket robot', *Scientific American Magazine* pp. 62–67.

Weiss, S. (2012), Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments, *in* 'Proc. IEEE Int. Conf. Robotics and Automation', pp. 957–964.

Williams, D. S. (1983), ‘Changes of photoreceptor performance associated with the daily turnover of photoreceptor membrane in locusts’, *J. Comparative Physiology* **150**, 509–519.

Wilson, M. (1975), ‘Angular sensitivity of light and dark adapted locust retinula cells’, *J. Comparative Physiology* **97**, 323–328.

Wilson, M., Garrard, P. & McGinness, S. (1978), ‘The unit structure of the locust compound eye’, *J. Cell and Tissue Research* **195**, 205–226.

Wolf, R., Gebhardt, B., Gademann, R. & Heisenberg, M. (1980), ‘Polarization sensitivity of course control inDrosophila melanogaster’, *J. Comp. Physiol.* **139**(177), 1–13.

Yang, R. & Pollefeys, M. (2003), Multi-resolution real-time stereo on commodity graphics hardware, *in* ‘Proc. IEEE Conf. Computer Vision and Pattern Recognition’, pp. 1–7.

Yue, S. & Rind, F. C. (2006), ‘Collision detection in complex dynamic scenes using an LGMD-based visual neural network with feature enhancement.’, *IEEE Trans. Neural Networks* **17**(3), 705–716.

Yue, S. & Rind, F. C. (2013), ‘Redundant neural vision systems-competing for collision recognition roles’, *IEEE Trans. Autonomous Mental Development* **5**(2), 173–186.

Zaretsky, M. & Rowell, C. H. F. (1979), ‘Saccadic suppression by corollary discharge in the locust’, *J. Nature* **280**, 583–585.

Zufferey, J., Beyeler, A. & Floreano, D. (2008), Optic flow to steer UAVs, *in* ‘Proc. Int. Symp. Unmanned Aerial Vehicles’, Orlando, FL, pp. 1–16.

Zuiderveld, K. (1994), Contrast limited adaptive histograph equalization, *in* ‘Graphic Gems IV’, pp. 474–485.