CRANFIELD UNIVERSITY

JASON GAUCI

OBSTACLE DETECTION IN AERODROME AREAS THROUGH THE USE OF COMPUTER VISION

SCHOOL OF ENGINEERING

PhD THESIS

CRANFIELD UNIVERSITY

DEPARTMENT OF AEROSPACE ENGINEERING
SCHOOL OF ENGINEERING

PhD THESIS

Academic Year 2009-2010

JASON GAUCI

Obstacle Detection In Aerodrome Areas Through The Use Of Computer Vision

Supervisor: Dr. David Zammit-Mangion

April 2010

This thesis is submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy

Abstract

This thesis addresses the problem of ground collisions between an aircraft and obstacles (including other aircraft) on the ramp and taxiway regions of an airfield. A safety study is conducted by looking at current operating procedures and analysing accident statistics and reports. An onboard non-collaborative system for large transport aircraft is proposed and its main requirements and performance characteristics are discussed. The main requirement is to detect and track generic obstacles around an aircraft during taxi manoeuvres.

The suitability of computer vision to the application of interest of this work is investigated through comparison with other candidate sensor technologies and computer vision, using visible cameras, is selected as the preferred technology. A study of different optical solutions is carried out and stereo vision is considered to be the most suitable choice. Two locations on the aircraft are considered for camera installation and the installation of a stereo vision system on each wingtip is chosen.

Algorithms are implemented for the different processing blocks of the stereo vision system. These comprise calibration, rectification, correspondence, reconstruction, detection, clustering, and tracking algorithms. For each process, existing methods and techniques are reviewed and the most appropriate ones are selected, modified and improved in order to meet the specific requirements of this application. The values of several parameters of each algorithm are found experimentally using synthetic data and each algorithm is tested individually before being integrated with the rest of the system.

Overall system performance is evaluated by testing for positional accuracy, generic obstacle detection and tracking capabilities, and sensitivity to calibration errors. Testing is conducted for a range of realistic conflict scenarios, under different illumination, visibility, and image noise conditions. Both synthetic images and real images are used. The results of both sets of images are compared and these suggest that the stereo vision system developed in this research has the potential to reduce wingtip collisions and can therefore improve safety and situational awareness in aerodrome areas.

Acknowledgements

I am grateful to a number of people who, directly or indirectly, knowingly or unknowingly, have helped me throughout my research.

First of all I would like to thank my supervisor, Dr. David Zammit-Mangion, who suggested that I take this research opportunity. Thank you for being a mentor and a friend throughout this journey.

The Department of Aerospace Engineering at Cranfield University has always supported my research and has provided me with the necessary tools and training to become a better researcher. For this reason I would like to thank the Head of Department and my co-supervisor, Professor John Fielding.

A big thanks goes to Dr. Toby Breckon for always being available whenever I sought his guidance on aspects of image processing and computer vision and for organising several interesting seminars related to these research areas.

I would like to thank Dr. Huamin Jia and Dr. James Whidborne who, together with my supervisor and co-supervisor, formed part of my progress review panel and provided me with very useful suggestions and constructive criticisms at various stages of my research.

Carrying out experiments, especially on an active airfield, can create logistical hurdles and necessarily involves the cooperation of several people. I would like to thank the following people without whose help such experiments would not have been possible: Barry Walker, Simon Hegarty and the Department of Air Transport, Roy Chamberlain, Richard Long, Terry Billings and the staff at Cranfield Airport.

My thanks also go to the British Machine Vision Association (BMVA) for providing me with a travel grant in order to be able to attend a conference in the United States and disseminate my research findings.

Last, but not least, I would like to thank my parents John and Pauline, my sister Mariella, and my relatives and friends for their constant moral support and for the good times we spent together. I dedicate this dissertation to all of you.

Contents

List of Figures

List of Tables

Nomenclature

A-SMGCS	Advanced Surface Movement Guidance and Control Systems
ACARE	Advisory Council for Aeronautics Research in Europe
ADS-B	Automatic Dependant Surveillance-Broadcast
ANN	Artificial Neural Network
ARF	Aircraft Reference Frame
ATC	Air Traffic Control
ATIS	Automatic Terminal Information Service
AWGN	Additive White Gaussian Noise
BMVA	British Machine Vision Association
CAP	Civil Aviation Publication
CC	Cross-Correlation
CCD	Charge-Coupled Device
CRF	Camera Reference Frame
DCAM	1394-based Digital Camera Specification
DCM	Direction Cosine Matrix
DMA	Direct Memory Access
DSP	Digital Signal Processor

EKF	Extended Kalman Filter
FOE	Focus of Expansion
FOV	Field of View
FPGA	Field-Programmable Gate Array
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HDR	High Dynamic Range
ICAO	International Civil Aviation Organisation
IR	Infrared
IRF	Image Reference Frame
LADAR	Laser Detection and Ranging
LIDAR	Light Detection and Ranging
MCMC	Markov chain Monte Carlo
MHT	Multi-Hypothesis Tracking
MMW	Millimeter-wave
MMX	MultiMedia eXtensions
MST	Minimum Spanning Tree
NOTAM	Notice To Airmen
OHCI	Open Host Controller Interface
PCA	Principal Component Analysis
PCI	Peripheral Component Interconnect
PMMW	Passive millimeter-wave

RAeS	Royal Aeronautical Society
RCS	Radar Cross Section
RTHP	Runway Taxi-Holding Position
SAD	Sum of Absolute Differences
SIMD	Single Instruction Multiple Data
SIR	Sequential Importance Resampling
SMR	Surface Movement Radar
SNR	Signal-to-Noise Ratio
SOM	Self-Organising Map
SOP	Standard Operating Procedure
SSD	Sum of Squared Differences
SSE	Streaming SIMD Extensions
SVD	Singular Value Decomposition
ToA	Time of Arrival
TTC	Time to Collision
UAV	Unmanned Air Vehicle
UKF	Unscented Kalman Filter
WRF	World Reference Frame

Chapter 1

Introduction

In a report issued in 2001 by the Advisory Council for Aeronautics Research in Europe (ACARE) it was predicted that, by 2020, air traffic will triple with respect to 2000 levels [4]. This implies that airports will continue to get bigger and busier and that the number of ground movements will increase. This will make it harder to ensure safe separation between aircraft and surrounding objects on the ground.

Figure 1.1: Typical apron environment (Singapore Changi Airport)

1.1 Current Procedures for Separation

The most congested area at an airport is the ramp (Figure 1.1). This is a very dynamic environment, with several aircraft taxiing in and out of the stands and parked aircraft being refueled, loaded/unloaded and boarded simultaneously. Aircraft are situated very close to each other, making it demanding to manoeuvre an aircraft in such

confined spaces. Taxiways are also very busy, with multiple aircraft moving between the runway and the ramp and queuing to enter the runway.

Many different types of obstacles can be found on ramps and taxiways. These include:

- Aircraft
- Vehicles (such as cars, fuel trucks, tow trucks, fire trucks, coaches, and baggage trucks)
- Fixed structures (such as light poles, hangars, and terminal buildings)
- Other (such as air bridges, stairs, and construction equipment)

In such a crowded environment, ground collisions are minimised by designing airports, operating procedures, and avionic systems in such a way as to ensure appropriate separation between an aircraft and obstacles. The following are a number of ways in which a safe separation is maintained:

- **Taxiways:** Taxiways are designed for use by all or certain types of aircraft. Several markings are made in specific areas on a taxiway, such as at the centreline, holding positions and taxiway edges. Other surface markings are used to provide information, directions or instructions. The role of centrelines in providing clearance is clearly defined in Civil Aviation Publication (CAP) 637 [5], which states that:
  "Taxiway centrelines are located to provide safe clearance between the largest aircraft that the taxiway is designed to accommodate and fixed objects such as buildings, aircraft stands etc, provided that the pilot of the taxiing aircraft keeps the 'Cockpit' of the aircraft on the centreline and that aircraft on a stand are properly parked."

  According to the same document:
  "Taxi Holding Positions are normally located so as to ensure clearance between an aircraft holding and any aircraft passing in front of the holding aircraft,

provided that the holding aircraft is properly positioned behind the holding position. Clearance to the rear of any holding aircraft cannot be guaranteed."

- **Air Traffic Control:** One of the roles of Air Traffic Control (ATC) is to maintain safe separation between aircraft on the ground. Controllers do this by monitoring ground movements, issuing clearances, and instructing aircraft to follow specific taxiway routes. Controllers are also aided by equipment such as the Surface Movement Radar (SMR). However, this has limitations and is primarily used in low visibility conditions.

- **Ground crew:** These include marshallers and wing walkers whose main function is to guide an aircraft when it is moving in or out of the gate. They are also responsible for keeping a good look-out for obstructions. The number of wing walkers normally increases with aircraft size.

- **Onboard systems:** More and more aircraft are being equipped with systems that help pilots with ground navigation. For example, the Airbus A380 is equipped with a camera system that aids pilots to judge each landing gear's location on the tarmac [6]. The aircraft also has an airport navigation system display which shows a detailed airport map and the aircraft's position. This improves situational awareness, especially in unfamiliar airports. Such systems will become more common in the near future as new aircraft and airports install Advanced Surface Movement Guidance and Control Systems (A-SMGCS).

- **Aircraft lights:** Aircraft have several different types of lights located on the exterior, such as on the wingtips, wing roots, landing gear, fuselage and tail. During ground manoeuvres, certain lights (such as taxi lights on the nose landing gear) help the flight crew to navigate more easily, especially in night-time and low visibility operations. In addition, all of the lights (such as strobe lights and position lights on the wingtips) make the aircraft more visible to surrounding traffic and to ATC, thus reducing the risk of collisions.

- **Speed limits:** Airport operators usually impose limitations on taxi speeds. Moreover, airlines normally place Standard Operating Procedure (SOP) limits and recommendations for their pilots. The speed limit varies with location. It is highest in straight taxiway runs (approx. 20-25kts) and decreases on the ramp (approx. 15kts) and in tight turns (approx. 10kts).[1] These limits help to reduce collision risks and also minimise damage in the event of a collision.

As explained above, controllers have an important role in maintaining separation between aircraft. However, according to Rule 37(2) of the Rules of the Air Regulations 2007 [7], the ultimate responsibility for aircraft safety when taxiing lies with the aircraft commander. In fact, the potential hazards for wingtip collision are sometimes known to the airport operator and their liability is mitigated by issuing Automatic Terminal Information Service (ATIS) or Notice To Airmen (NOTAM) statements such as *'wingtip clearance is not assured'*.

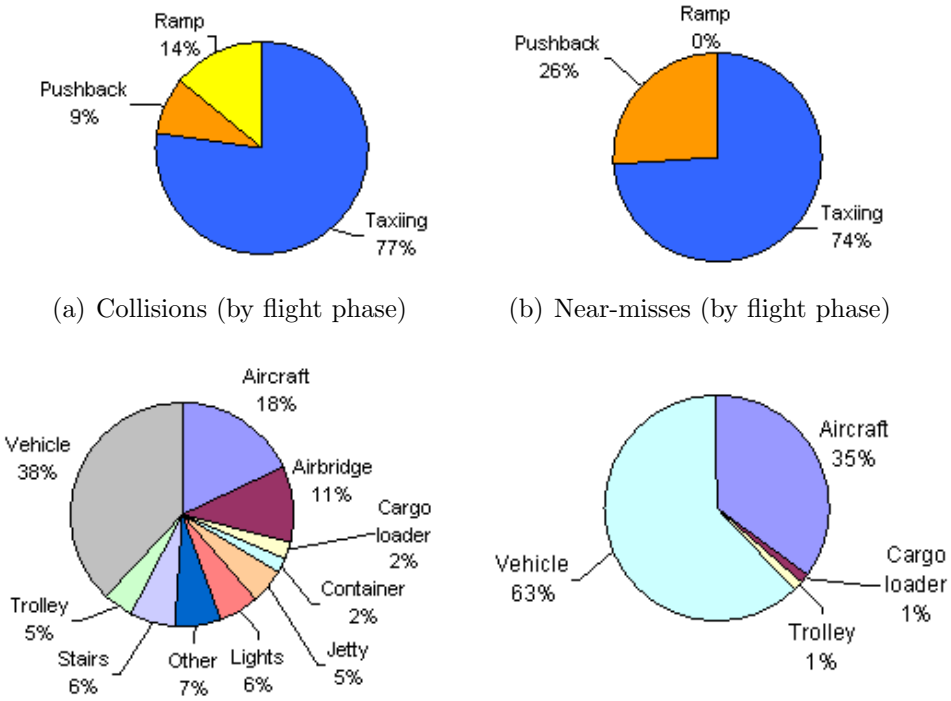
1.2 Investigation of Incidents and Accidents

A safety study conducted by the University of Malta [1] contains a comprehensive list of incidents and accidents that occurred in aerodrome areas worldwide over the period 1991-2005. The results shown in Figures 1.2(a)-1.2(d) are derived from this study and focus on three ground manoeuvres: ramp movements, pushback, and taxiing. These account for a total of 253 incidents and accidents, of which 70% are actual collisions while the rest are near-misses. From Figures 1.2(a)-1.2(b) it can be observed that the majority of collisions occur when an aircraft is taxiing. The greatest collision threats are vehicles and other aircraft as observed in Figures 1.2(c)-1.2(d).

Most incidents and accidents occur due to a combination of contributory factors, including:

- lack of ground crew to provide guidance
- poor communication between ATC, ground crew and flight crew

[1] These values are only guidelines. The actual speed limits vary between aircraft.



(a) Collisions (by flight phase)

(b) Near-misses (by flight phase)

(c) Obstacles involved in collisions

(d) Obstacles involved in near-misses

Figure 1.2: Analysis of collisions and near-misses (based on the safety study conducted in [1])

- ramp congestion and improper positioning of equipment or aircraft
- poor visibility
- violation of clearances
- crew distraction
- misjudgement of separation by the flight crew

These factors are essentially all due to a failure to follow procedures and often are a result of inadequate staff training. Of particular interest to this research are collisions between two aircraft when taxiing. Reports of some of these collisions can be found in [8–12]. The collisions investigated in these reports involve large commercial

passenger aircraft. In each of these accidents, the wing of a taxiing aircraft has come into contact with the wing or tail of a stationary aircraft as shown in Figure 1.3. Therefore, the parts of an aircraft that are commonly damaged include the wingtips, winglets, rudders, fins, stabilisers and elevators (Figure 1.4). What is interesting to note is that most of the collisions between two aircraft occur in fine weather and good visibility, proving that weather conditions may have very little effect as a contributory factor. In most cases, the pilots of the taxiing aircraft are aware of the other aircraft but misjudge the separation between the two aircraft.

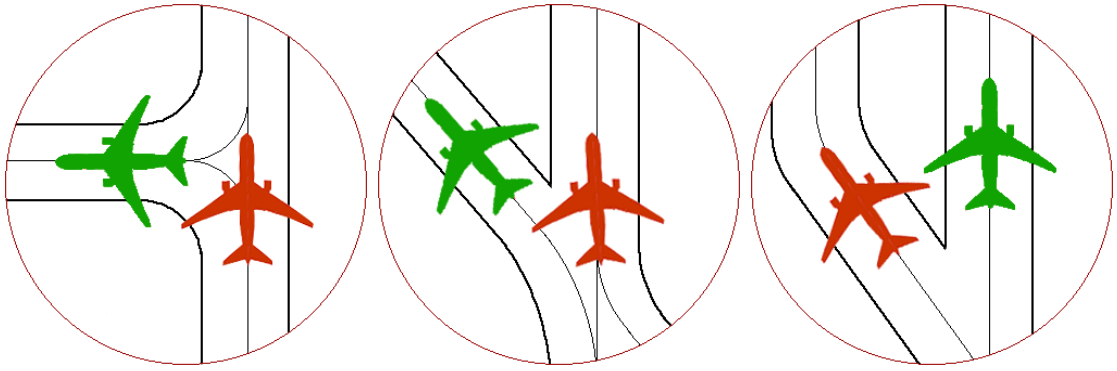


Figure 1.3: The three most common collision scenarios between a taxiing aircraft (red) and another aircraft (green) (identified from the University of Malta survey [1] and collision investigation reports [8–12])

Judging distances from the cockpit of a large aircraft is not a trivial task. For instance, the Airbus A380 has a semi-wingspan of 39.9m and the Boeing B747 has a semi-wingspan of 33.2m. In most cases, pilots either have a restricted view of the wingtips (a wingtip might only be visible with difficulty from one pilot seat), an impaired view (due to precipitation or dirt on the side windows) or no view at all.[2] Distance judgement is complicated by the fact that most commercial transport aircraft have swept wings and these are subject to an effect known as *swept wing growth* or *wing creep* [13]. This means that, during a turn, the wingtip describes an arc greater than the normal wingspan due to the geometry of the aircraft and the

[2]For example, flight crew on a B777 are unable to see their aircraft's wingtips from the flight deck [9].

(a) (b)

Figure 1.4: Damage sustained as a result of a ground collision between two aircraft: (a) damage to horizontal stabiliser of B737 involved in a collision with a B767 at Manchester Airport [10] (b) damage to left wingtip of B747 involved in a collision with a B767 at Melbourne Airport [11]

arrangement of the landing gear. This effect is most noticeable in tight turns but it still degrades wingtip clearance judgement in any turn.

Although incidents and accidents on ramps and taxiways pose a relatively low risk when compared to for example, runway incursions, they are highly undesirable. Apart from the direct costs associated with an accident (due to passenger injuries, aircraft damage and repair) there are several indirect costs such as those due to investigations, flight cancellations, aircraft down-time, leasing of replacement aircraft, and tarnishing of airlines' public image.

1.3 Proposed Solution and System Considerations

From the previous sections it can be observed that, although several precautions are taken to prevent ground collisions on ramps and taxiways, accidents still occur quite frequently. This suggests that current methods and systems only provide a partial solution to the problem. This research proposes a novel system that can be installed on an aircraft to provide further protection against such occurrences, particularly

wingtip collisions. The solution proposed is an onboard non-collaborative[3] system. This has the advantage of being completely independent of airport infrastructure and of other aircraft and obstacles. The platform assumed for this research is a large transport aircraft (with dimensions similar to the Airbus A380) since such an aircraft is expected to benefit most from this system. From this point onwards, the platform will be referred to as the *ownship*.

The main functional requirements of the proposed system are:

1. To detect and track obstacles around an aircraft during ground manoeuvres
2. To alert the flight crew in the event of loss of separation or a potential collision

This research focusses mainly on the first requirement. This requirement presents a number of challenges:

- **Obstacles:** As seen in Section 1.2, aircraft come into conflict with several types of obstacles that are very diverse in shape and size. Therefore, the system needs to be able to detect generic obstacles. However, since the biggest threats are by far vehicles and other aircraft, more care needs to be taken to detect these types of obstacles. In the case of aircraft, the system needs to be able to detect the extremities (especially the wingtips and tail).
- **Monitoring zone:** Most collisions occur because pilots have limited visibility of the wingtips and misjudge separation from surrounding obstacles. For the system to have maximum effectiveness, it needs to focus on the most vulnerable areas of an aircraft. Accordingly, it therefore needs to focus mainly on obstacle detection around the wingtips.

  The extent of the monitoring zone around the wingtips depends on the minimum specified wingtip clearances for a particular aircraft. These clearances vary according to aircraft size and ground manoeuvre. For instance, in the case of the A380, three main clearances specified by the International Civil Aviation

[3] i.e. independent

Organisation (ICAO) are: 17.5m during manoeuvres on a taxiway, 10.5m during manoeuvres on an apron or during turns, and 7.5m when parked on a stand [14]. In this research, a minimum wingtip clearance of 10m is arbitrarily assumed.

The extent of the monitoring zone also depends on aircraft speed, the pilot reaction time to an alert, and the braking distance required. For example, consider the scenario when an aircraft is taxiing at 25kts (approx. 12.86m/s) on a straight taxiway, with a stationary aircraft located ahead at an intersection with another taxiway as shown in Figure 1.3(a). If the ownship continues moving in the same direction, its left wingtip will come in contact with the tail of the other aircraft. The distance required to bring the ownship to a halt (d_{stop}) can be estimated using standard linear equations of motion:

$$\begin{aligned} d_{stop} &= t_{react} v_i + d_{brake} \\ &= t_{react} v_i + \frac{{v_f}^2 - {v_i}^2}{2a_{brake}} \end{aligned} \tag{1.3.1}$$

where:

d_{brake} is the braking distance,

v_f is the final speed (0m/s),

v_i is the initial speed (12.86m/s),

t_{react} is the pilot reaction time,

a_{brake} is the average braking deceleration.

Pilot reaction time to an alert in the cockpit depends on several factors, including: alerting system design, pilot fatigue, crew workload, and pilot training. Typical pilot reaction times lie in the range 1-3s [15–17].

The deceleration of an aircraft (due to the wheel brakes) depends on multiple parameters such as aircraft weight, condition of brakes and tyres, and ground surface friction. For example, during a brake test conducted on the landing gear of an A380-800, a mean deceleration of $-3.62 m/s^2$ was achieved [18]. In an experiment carried out at Cranfield University's B747 flight simulator [19],

the average braking deceleration of the B747 flight model was $-4.56m/s^2$ (Refer to Appendix A for more details and results of this experiment).

Assuming a pilot reaction time of $2s$ and an average braking deceleration of $-3.5m/s^2$ and substituting these values in Equation (1.3.1), the longitudinal distance required to stop d_{stop} is found to be 49.35m. The combination of minimum lateral wingtip clearance (10m) and distance required to stop (49.35m), in essence, define a protection zone around the vulnerable areas of an aircraft (i.e. the wingtips), as shown in Figure 1.5.

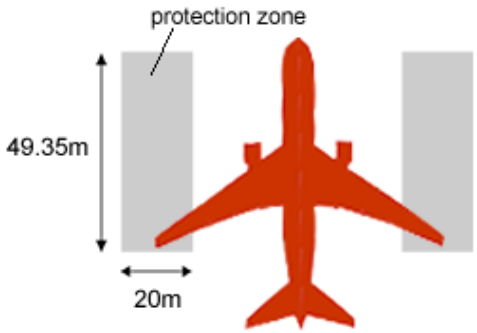


Figure 1.5: Definition of a protection zone around the wingtips

The protection zone is the smallest region around a wingtip that has to be clear of obstacles such that, in the event of a conflict (where an obstacle is detected as being on a collision course with the wingtip), the pilots will have enough time and space to react and bring the ownship to a stop without colliding with the obstacle. This means that the system needs to focus on the detection and tracking of obstacles in the region beyond the boundaries of the protection zone. This is necessary in order to be able to detect potential collisions - and, hence, to issue timely warnings - before an obstacle penetrates the protection zone. Once an obstacle enters the protection zone, it may be too late to avoid a collision.

In this research, the protection zone is assumed to be fixed. However, in a practical implementation of the system, the size of the protection zone should be changed dynamically depending on the ownship's speed. The higher the

ownship's speed, the longer the protection zone should be in order to cater for the greater distance required to stop the ownship in the event of a conflict.

Given the size of the protection zone chosen in this work (49.35m by 20m), a detection range of a 100m was considered necessary in order to provide adequate monitoring of obstacles prior to entry into the protection zone. Considering that the protection zone is a worst case scenario, this monitoring range essentially represents the maximum performance, in terms of range, required of the optical system.

- **Robustness:** The system needs to be sensitive enough to detect conflicts due to obstacles inside the monitoring zone (to keep the missed detection rate low) while at the same time being able to reject outliers (to keep the false alarm rate low). Naturally, a compromise between missed alarms and false alarms has to be reached; however, it is preferable to have a lower false alarm rate at the expense of a slightly worse missed detection rate than vice-versa. This is because, even if not all obstacles are detected, the new system will still be an improvement over the current (baseline) situation.

- **Positional accuracy:** The accuracy associated with the measurement of obstacle position needs to be related to the size of the protection zone. Therefore, for this application, the positional accuracy needs to be on the order of a few metres (or better). Positional accuracy can affect the robustness of the system, particularly at the boundaries of the protection zone. For example, if an obstacle is outside the protection zone but is detected as being inside, a conflict could be incorrectly detected. Similarly, if an obstacle is inside the protection zone but is detected as being outside, a potential conflict might go undetected or is detected after some time delay. Hence, the higher the accuracy, the lower the possibility of having missed or false warnings at the border of the protection zone.

- **Update rate:** Alerting systems naturally require low latency, that is, minimal

time delays in generating an alert. Consequently, any detection algorithm will need to have an update rate that does not introduce significant delays. This places constraints on the processing time of the algorithm. For this application, the system needs to update in real-time, with the time interval between updates being on the order of milliseconds. With such an update rate, the potential distance travelled by the ownship in the time span between conflict occurrence and conflict detection, is negligible. However, if the time interval between updates is large (e.g. on the order of seconds), then the time delay in conflict detection and the potential distance travelled by the ownship during this time delay, are also large. In this case, the size of the protection zone needs to be increased in order to take into account the distance travelled by the ownship during the time delay and to ensure that the pilots still have enough room to avoid a collision.

- **Time of day and weather:** Ideally, the system and its sensors should perform well at any time of the day and in bad weather (low visibility) as well as good weather conditions. However, as mentioned previously, most ground collisions (especially wingtip collisions) occur during day-time and in clear weather and visibility. During the night and in adverse weather conditions, there is generally less traffic on the ground and ATC take extra care to ensure separation between aircraft.

1.4 Aim and Objectives

The aim of this research was to develop an obstacle detection and tracking system to support the averting of ground collisions between the ownship and surrounding objects when taxiing on ramps and taxiways. Such a system would assist pilots of large transport aircraft by improving situational awareness and detecting potential conflicts.

Consequently, the main objectives of this research were:

- To assess and compare the suitability of optical sensors for this application
- To select a suitable optical sensor and optical solution
- To select signal processing and computer vision techniques that are best adapted to the task
- To develop signal processing and computer vision algorithms to detect and track obstacles around an aircraft when taxiing
- To evaluate the performance of the system by means of synthetic and real data, the latter obtained during field trials in an actual aerodrome environment

1.5 Thesis Outline

The rest of the thesis is organised as follows.

The first part of Chapter 2 reviews and compares various sensor technologies and demonstrates that computer vision is indeed suitable for the application of interest of this work. Computer vision, using visible cameras, is selected as the preferred technology. Then, different computer vision solutions are reviewed and discussed in detail. The second part of the chapter gives an overview of the proposed system. First, the suitability of stereo vision to this application is discussed. Then, each of the processing blocks of stereo vision is described. This is followed by an investigation of two possibilities for the location of cameras on the ownship. Finally, the values chosen for the main stereo vision parameters are presented and the procedure used to generate synthetic images with the proposed stereo setup is outlined.

Chapter 3 focusses on calibration of the optical setup. Different reference frames are defined and the camera model is introduced. The intrinsic and extrinsic camera parameters are presented and the calibration algorithms used to estimate these parameters are explained. The calibration results of the simulated camera setup are presented and discussed.

Chapter 4 addresses the closely-linked problems of rectification and correspondence. The epipolar geometry, necessary to understand the purpose of rectification, is explained. Then, the rectification algorithm is discussed and some rectification results are presented. The remainder of the chapter then focusses on correspondence. The issues, constraints, and assumptions associated with correspondence are presented and the main correspondence methods are outlined. The various features of the correspondence algorithm designed for this application are explained and several correspondence results are presented.

The first part of Chapter 5 addresses three-dimensional (3D) reconstruction and describes the process of reconstruction by means of triangulation. The effect of the baseline distance and focal length on the accuracy of triangulation is discussed and the results of experiments used to select appropriate values for these two parameters are presented. The second part of the chapter discusses obstacle detection. The first phase of detection, involving height thresholding, is explained and the impact of wing flexing on performance is addressed. Then, the second phase of detection, which is based on the concept of clustering, is discussed. An overview of the main clustering techniques is presented and the clustering algorithm developed in this work is outlined. Finally, various obstacle detection results are presented.

Chapter 6 discusses obstacle tracking. An overview of visual tracking and state estimators is presented and the benefits of tracking through the use of a Kalman filter are outlined. The design of the filter is discussed in detail and the logic used for obstacle tracking and outlier rejection is explained. Several tracking results are also presented.

Chapter 7 focusses on validation testing which was carried out in order to determine overall system performance and to identify any limitations. The first part of the chapter focusses on simulation testing. The design of experiments used to test specific aspects of the system in different conflict scenarios under various conditions of simulated illumination, visibility, and image noise, is described. Then, the results of these experiments are presented and discussed. The second part of

the chapter addresses the experiments carried out using real cameras. The hardware setup and experimental design are explained and the results obtained are discussed and compared with the simulation results.

Chapter 8 discusses the strong points and limitations of the system, highlights the key contributions to the field of avionics, proposes areas for further research, and presents the main conclusions of this work.

Chapter 2

Literature Review and System Overview

This chapter is divided into two main sections. Section 2.1 provides a literature review of different technologies and techniques that are available to detect and locate obstacles. Section 2.2 presents the solution that is proposed in this work and gives an overview of the designed system.

2.1 Literature Review

2.1.1 Review of Candidate Sensor Technologies

There are several sensor technologies that can be used for the purpose of obstacle detection and localisation. The most commonly used are: active and passive millimiter-wave (MMW) radar, Light Detection and Ranging (LIDAR),[1] infrared (IR) cameras and visible cameras. Passive MMW radar sensors, IR cameras and visible cameras produce 2D images by perspective projection and these images can be processed to extract 3D information about the scene. This concept is known as *computer vision*. In the literature, computer vision is most commonly associated with images captured by visible cameras or IR cameras. Therefore, for the purpose of this work, the term 'computer vision' is used to refer to the processing of images acquired

[1]The acronym LADAR (Laser Detection and Ranging) is often used in military contexts.

by these types of sensors. From the very beginning of this research, the intention was to use either one, or both, of these sensors. Hence, the main objectives of this section are (a) to demonstrate the suitability of IR cameras and visible cameras to this application by comparing them with the other sensors and (b) to decide whether to use visible cameras and/or IR cameras.

Millimeter-wave (MMW) radar: Two techniques of MMW radar operation exist, namely passive and active. Passive MMW radar makes use of the inherent electromagnetic radiation of all objects at temperatures above zero degrees Kelvin. The magnitude of this radiation increases with temperature and object emissivity. This radiation peaks in the IR region[2] but narrow spectral windows in the MMW region have been identified at 35GHz, 94GHz, 140GHz and 220GHz. At these frequencies, atmospheric absorption is very low. Hence, passive MMW radar sensors can detect this radiation.

Active MMW radar sensors emit MMW frequencies to illuminate a target and then measure the reflected signal. From the amplitude, spectral content and Time of Arrival (ToA) of the return signal (the echo), it is possible to determine target distance, speed, azimuth, elevation, size and other characteristics. Active MMW radar is ideal for detecting metallic objects because of their high reflectivity. However, one problem of this sensor is that the Radar Cross Section (RCS) of a target (and therefore the amplitude of the return signal) can fluctuate with changes in the target's attitude. This effect is known as *glint*.

The selected frequency of operation depends on the application. The lower frequences (35GHz) support further signal propagation than the higher frequencies (220GHz) but this is achieved at the expense of spatial (angular) resolution. In order to scan a multi-dimensional region of interest, opto-mechanical or electronic scanning techniques are used. Passive MMW radar can generate images using a 2D

[2]Black-body radiation peaks in the IR region for a body at approx. $300K$, but at shorter wavelengths for hotter objects.

phased array radiometer. The contrast and resolution of the images is dependent on a number of factors such as the beamwidth and the antenna aperture. The larger the aperture, the narrower the beam can be and the better the angular resolution that can be achieved. Also, for a particular size of antenna, the beamwidth can be made narrower by increasing the operating frequency. In order to make MMW radar sensors practical and portable for several applications (such as automotive and Unmanned Air Vehicle (UAV) applications), the size of the antenna (and therefore the aperture) is severely constrained. As a result, MMW cameras generally have low spatial resolution and the images obtained tend to be blurred. Nevertheless, from the images it is possible to identify basic object parameters such as position, size and geometry. Some image enhancement can be achieved by using super-resolution [20] and deblurring [21] techniques.

MMW systems have excellent weather penetration capabilities and are practically unaffected by fog, rain, snow, smoke, dust or clouds. Applications of passive MMW radar include aircraft landings in low visibility conditions [22] and target detection [23] whereas applications of active MMW radar include urban area navigation [24] and automotive obstacle detection [25].

LIDAR: LIDAR involves an active optical sensing technology operating in the ultraviolet (UV), visible or near-infrared (NIR) regions. The most common sensing method consists of emitting a laser beam and measuring the time delay between transmission and reception of the reflected signal. The width of a laser beam can be much narrower than that of a radar beam, resulting in higher spatial resolution. The range resolution of LIDAR is also very high. A multi-dimensional region of interest can be scanned using opto-mechanical or electronic scanning techniques (in which case the device is also known as a *laser scanner*). Object properties that can be detected with LIDAR include distance, speed, direction, size and geometry. The operation of this sensor is not affected by illumination conditions and by slightly bad weather. Applications of LIDAR include terrain mapping and classification [26], obstacle and terrain detection during aircraft landings in low visibility [27], and weather detection

(such as windshear detection [28]).

Infrared (IR) cameras: Infrared cameras are classified into the reflective type, which operate at the near- and short-wavelength IR bands, and the thermal type, which operate at the mid- and long-wavelength IR bands. Thermal cameras are passive sensors that capture IR radiation (heat) emitted by objects and the environment. They can therefore detect objects whose temperature is higher than that of the surroundings. Thermal cameras are ideal for detecting warm or hot objects. On the other hand, reflective IR cameras illuminate the scene under observation to produce a thermal contrast between features of interest and the background. This approach is necessary when the inspected parts are in equilibrium with the surroundings.

IR sensors are suitable for day-time and night-time operations but IR radiation is absorbed by fog, clouds and precipitation. IR cameras generally have good spatial resolution. Object properties that can be extracted from the images include position, size and geometry. It is also possible to identify and classify an object on the basis of its thermal 'signature'. However, it is not possible to measure distance directly from the images. Typical applications of thermal IR cameras are pedestrian detection [29] and vehicle detection and localisation [30]. Reflective IR cameras are used extensively in surveillance applications.

Visible cameras: Visible cameras are passive devices that respond to visible light that is reflected by different objects in the environment. They are generally the most compact and cost-effective of all of the sensors described. They also provide the highest level of spatial resolution. The images obtained from these devices are very rich in content and provide a lot of information about the captured objects, such as position, size, geometry, texture and color. Due to the high resolution, it is also possible to identify and classify objects. However, it is not possible to measure object distance directly from the images. The performance of visible cameras degrades considerably in poor illumination and bad weather conditions. Visible cameras are used extensively for research purposes and applications of

these cameras include road obstacle detection and collision avoidance [2], vehicle tracking [31], face recognition [32], aircraft state estimation [33] and aircraft guidance and navigation [34].

Table 2.1 summarises the key properties of each of the sensors described. The values presented in the table for the maximum detection range of each sensor are only meant to give an indication of the capabilities of each sensor. These values were obtained by looking at different obstacle detection systems - each making use of a particular sensor - designed for automotive applications [30, 35–38].

Table 2.1: Summary of comparison of different sensor technologies

	MMW radar (Passive or active)	**LIDAR**	**IR camera** (Thermal or reflective)	**Visible camera**
Spectral band	MMW	UV, visible or NIR	IR	Visible
Spatial resolution	Poor	Very good	Good	Excellent
Weather penetration capability	Excellent	Good	Good	Poor
Object detection capabilities	Distance, speed, direction, size, geometry	Distance, speed, direction, size, geometry	Position, size, geometry, object class	Position, size, geometry, texture, color, object class
Maximum detection range (m)	150 [35]	150 [36]	100 [30]	90 [37], 100 [38]
Size	Medium to large	Medium to large	Small to medium	Small
Moving parts	Depends on the scanning mechanism	Depends on the scanning mechanism	No	No
Cost	Medium to high	Medium to high	Medium to high	Low to medium

As can be observed from the previous discussion and from Table 2.1, some of the sensors have complementary capabilities. For example, passive MMW radar has excellent weather penetration capabilities but poor spatial resolution. On the other hand, visible cameras are susceptible to changes in illumination and weather conditions but have excellent spatial resolution. Therefore, in order to improve

performance and reliability in different operating conditions, several applications use a combination of sensors for obstacle detection and localisation. For instance, in [39, 40], a MMW radar and a visible camera are mounted on a car whereas, in a similar application [41], a laser scanner and two visible cameras are used. In [42], a total of five sensors are mounted on an UAV: two visible cameras, two IR cameras and a Ka-band radar.

The information gathered by sensors operating in different spectral bands can be combined using sensor fusion (also known as *multi-sensor data fusion*) techniques. Sensor fusion can be implemented at a low level (also known as *measurement-level fusion*), intermediate level (also known as *feature-level fusion*) or high level (also known as *decision-level fusion*). The concept of measurement-level fusion is that the raw outputs (measurements) of various sensors are merged in order to produce a single output (such as an image) which then undergoes further processing. This output contains the relevant information from each spectral band. During feature-level fusion, certain features (such as edges, lines and texture parameters in the case of images) are first extracted from the raw output of each sensor. Then, these features are combined into a single output (as in the case of low level fusion) which is passed on to other processes. When fusion is carried out at a high level, the raw outputs of the sensors are first processed individually. Then, the results obtained from each sensor are combined to reach a global decision. Several sensor fusion algorithms are available in the literature, including Kalman filtering [43], neural networks [44], fuzzy logic [45] and Principal Component Analysis (PCA) [46].

There are a number of issues related to sensor fusion. These include:

- **Synchronisation:** Acquisition of data from multiple sensors requires the synchronisation (temporal alignment) of the sensors' outputs. This is particularly relevant in real-time applications (such as the application of interest of this work) and dissimilar technologies can result in the added implications of different sensor update rates that may affect overall performance.

- **Registration:** When fusing data from different imaging sensors at a low level

(such as images from a thermal IR camera and a visible camera), the images do not only have to be aligned temporally but also spatially (at the pixel level). This process is called *registration* and is complicated by differences (such as in Field of View (FOV) and image resolution) between the imaging sensors.

- **Computation time:** Signal processing computational time is dependent on a number of factors such as the number of sensors, the complexity of the techniques and algorithms, and the processing power available. The computational requirements can be very demanding and can render certain techniques un-implementable in real-time on a particular platform.

From the discussion presented in this section it can be said that IR cameras and visible cameras are suitable for the application of interest of this research. The typical detection range of both types of sensors is sufficient to detect obstacles within and beyond the protection zone defined around the ownship's wingtips in Section 1.3. Also, the images captured by both types of sensors provide a lot of information about objects in the scene. Moreover, these sensors have complementary properties. Visible cameras have excellent spatial resolution but are affected by changes in illumination and weather conditions. On the other hand, IR cameras have lower spatial resolution but can operate at any time of the day and are less affected by bad weather conditions. This suggests that the two sensors could be used together in this application. Nevertheless, for this research it was decided to use only visible cameras, due to the rich content of their images and their superior spatial resolution. Spatial resolution is a very important property for this application because of the need to not only detect obstacles but also to clearly identify their contours (boundaries). This is particularly challenging in the case of narrow and long obstacles, such as aircraft wings. Another reason for the selection of visible cameras is that the technology of these cameras is more mature than that of IR cameras and the techniques available are much more extensive. Fusion of visible and IR cameras may be considered in future work (refer to Section 8.3).

2.1.2 Review of Candidate Optical Solutions

This section discusses different optical solutions that make use of visible cameras in order to detect and localise obstacles. These solutions are discussed mainly in the context of automotive and UAV applications, since these two areas bear the greatest resemblance to certain aspects of the problem that is addressed in this research. Different optical solutions can be applied depending on the number of cameras used. These solutions can be broadly classified into three groups: monocular, polyocular, and hybrid systems.

2.1.2.1 Monocular Systems

In a monocular system, images are captured by a single camera. If there is relative motion between the camera and the scene, an image pixel corresponding to a 3D point in the scene will appear to move from one frame to the next. This apparent motion is called *optical flow* (or *optic flow*). If the optical flow is computed for every pixel in the image, a 2D motion vector field is obtained, where each vector represents the velocity of an image pixel. By detecting changes in the optical flow field and grouping neighboring vectors according to their size and orientation, it is possible to detect objects in the scene.

Assuming that objects in the scene are stationary, the optical flow F of an object can be expressed as follows [47]:

$$F = -\omega + \frac{vsin\theta}{d} \tag{2.1.1}$$

where:

ω is the rotational velocity of the camera,

v is the translational velocity of the camera,

d is the distance between the object and the camera,

θ is the angle between the object and the direction of motion of the camera.

This expression is illustrated in Figure 2.1. Note that the optical flow vector F is perpendicular to the line joining the camera to the obstacle.

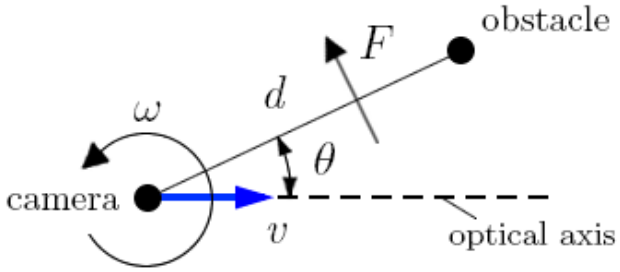


Figure 2.1: Plan view of a camera moving past an obstacle

From Equation (2.1.1) it can be observed that the optical flow components, due to the rotational and translational velocities of the camera, are linearly separable. The rotational flow component ω appears as a constant optical flow vector that is added to each pixel in the motion vector field. This component does not encode any range information and does not contribute to object detection. For this reason, it has to be removed from the optical flow field before any further processing is carried out. It can also be observed that, if the camera is stationary (that is $v = 0$), the translational flow component is also 0 and, hence, no stationary objects can be detected. Similarly, if an object lies directly in front of the camera (that is $\theta = 0$), the translational flow component will be 0, even if the object and camera are approaching each other. This occurs because the image projection of the object is located at the Focus of Expansion (FOE), which normally coincides with the centre of the image, along the optical axis of the camera. In this case, the object will not appear to move in the image plane. The translational flow component reaches a maximum value when $\theta = 90°$.

Apart from detecting obstacles from the optical flow measurements, it is possible to estimate the Time to Collision (TTC). The TTC can be expressed solely in terms of θ as follows [48]:

$$TTC = \frac{cos\theta sin\theta}{\dot{\theta}} \tag{2.1.2}$$

From Figure 2.2 it can be observed that θ can be estimated just by relying on optical parameters, as follows:

$$\theta = tan^{-1}\left(\frac{d_{FOE}}{f}\right) \tag{2.1.3}$$

where f is the lens focal length and d_{FOE} is the image plane distance between the

obstacle and the FOE. If the translational velocity of the camera is known, it is also possible to determine the distance between the camera and obstacle by multiplying the TTC by the closure rate between the camera and the obstacle.

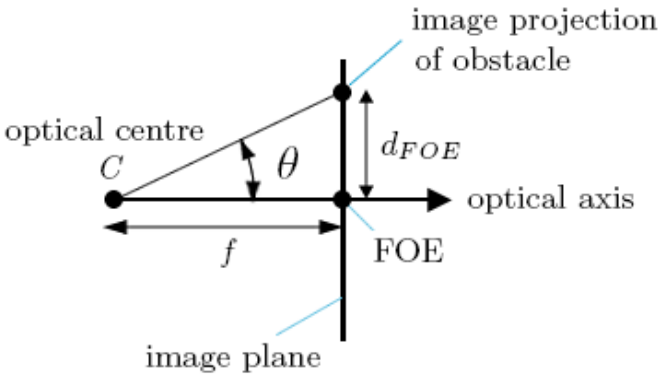


Figure 2.2: Plan view of camera and image plane

There are a number of problems associated with optical flow techniques. One issue is that these techniques are sensitive to camera shocks and vibrations which introduce random deflections in the optical flow field vectors. For this reason, image stabilisation and motion compensation techniques are required to correct this effect. Another drawback of optical flow is that the accuracy of the flow measurements depends on the relative speed between obstacles and the camera. TTC and distance estimates degrade at low speeds because the optical flow between consecutive frames will be very small. One solution to this problem is to increase the *temporal baseline*, that is to compute the optical flow between the current image and an image captured a number of frames N previously. This effectively magnifies the optical flow and improves the detection of distant objects in the scene. However, the larger the value of N, the greater the differences between the images used to compute the optical flow and the harder it is to find corresponding points between the two frames.

In order to identify moving objects in the scene from the measured optical flow pattern, the 3D motion of the camera (also known as *egomotion*) needs to be estimated first in order to remove the component of the optical flow which is caused by the movement of the camera. The remaining optical flow will be due to any moving objects in the scene. The camera motion parameters can change between frames

and, therefore, the egomotion needs to be estimated in each frame. Most egomotion estimation methods require a number of image features or regions (corresponding to stationary elements in the scene) to be tracked in consecutive frames [49, 50]. This is a computationally expensive process and is complicated by the fact that, in several cases (such as in automotive applications), very few (if any) reliable features are present in the scene.

Monocular systems using optical flow techniques are most suitable in applications where the camera is moving and where it is not necessary to know the 3D position of an object. Giachetti et al [51] present a system that uses optical flow for road navigation. The camera is mounted on top of a vehicle, with its optical axis aligned along the longitudinal axis of the car. First, the optical flow is estimated and processed in order to compensate for variations in the optical flow pattern due to mechanical disturbances. Then, assuming that the vehicle is on a flat road and that the optical axis of the camera is parallel to the road surface, the egomotion is estimated from the optical flow. This provides information regarding the speed and rotational velocity of the vehicle. Finally, the optical flow is roughly segmented (by grouping similar optical flow vectors) to detect objects moving at different speeds. Approaching and departing (or overtaking) vehicles produce diverging and converging optical flow patterns, respectively. In order to refine the localisation of motion boundaries and to remove some of the incorrect optical flow vectors, intensity edge information is used. The system is tested with real road images and gives mixed results, mainly because of the effects of shocks and vibrations which result in unreliable optical flow estimates. Also, the system fails in a cluttered environment with several obstacles close to (or on) the road.

Another system that uses optical flow for the detection of obstacles on the road is proposed by Demonceaux and Akkouche [52]. A single forward-looking camera is mounted on a vehicle. The motion of the road is modeled by a 2D quadratic model and the optical flow of the image region corresponding to the road is estimated in each frame by means of wavelet analysis. Then, a hierarchical Markov model is used

to detect areas in the image whose optical flow does not conform with the motion of the road. These areas are segmented into individual obstacles which are tracked over time in order to detect outliers due to image noise and motion estimation errors. The system is tested in an actual road environment and is able to detect generic obstacles. However, no distance or TTC measurements are made.

Roderick et al [53] propose an optical flow algorithm for the purpose of obstacle avoidance of an autonomous aircraft in flight. Only the optical flow of certain image features is computed in order to reduce computation time. A control strategy is implemented which navigates the aircraft away from obstacles (represented by regions of high-magnitude optical flow) towards free space (represented by regions of low-magnitude optical flow). The navigation commands minimise a cost function which takes into account several criteria such as the location of feature points and their associated optical flow magnitude. Another control strategy is also implemented which is based on a scene reconstruction algorithm that is responsible for mapping the environment and planning a path for the aircraft to follow. However, the scene reconstruction algorithm runs at a much slower rate than the optical flow algorithm. During the flight, the system switches between these two control strategies. The benefit of navigating with the use of optical flow estimation and scene reconstruction techniques is demonstrated through a number of simulations.

In the monocular applications discussed so far, the camera is mounted on a moving platform. However, there are various other applications where the camera position and orientation do not change. In this case, if the motion of obstacles is constrained to a plane, it is possible to detect obstacles and determine their position with respect to the camera. This is done by finding a projective transformation (or *homography*) between the plane and its image projection. A system that makes use of this technique for the purpose of traffic surveillance is proposed by Coifman et al [54]. In this application, a camera is mounted above a highway and a homography is obtained between the road surface and its image projection. This is done by manually selecting a number of points or lines in the image whose 3D location is known. Once the

homography is estimated, it is possible to determine the 3D location of any image feature that corresponds to an object on the road. In each frame, corner features are detected and tracked. Then, corners that have similar motion characteristics and that satisfy a spatial proximity criterion, are assumed to belong to the same obstacle (vehicle) and are grouped together. Each detected vehicle is tracked over a distance of about 100m and its trajectory is recorded. Several traffic parameters, such as traffic flow and vehicle speed, are obtained. The system is implemented on a network of Digital Signal Processors (DSPs) and reaches an update rate of 7.5Hz in uncongested traffic and 2Hz in congestions. Good performance is achieved with real images in challenging traffic and illumination conditions.

Instead of detecting obstacles directly by tracking image features, another method that is commonly used in the case of stationary cameras is to obtain a model of the background. Since the camera does not move, the background is relatively constant. Therefore, foreground objects (including static and moving objects) can be detected indirectly by removing the background from the image. This technique is called *background subtraction* or *foreground segmentation*. In reality, the background is likely to change due to variations in lighting and weather conditions. Therefore, a static background model is not sufficient in most cases and has to be updated in each frame. For example, Manzarena and Richefeu [55] propose an adaptive background estimation technique for video surveillance applications. It is assumed that, at the pixel level, background intensities are present most of the time. Therefore, the algorithm records the temporal variations of the intensity of each image pixel and, from these statistical measures, it determines (for each pixel) whether a change in pixel intensity between consecutive frames is due to the presence of a foreground object (such as a vehicle) or not. This information is then used to update the background model. In order to make this process more robust, the spatial correlation between neighboring pixels is taken into account by using intensity edge information.

Vargas et al [56] argue that the algorithm proposed in [55] gives poor results in challenging urban traffic scenarios which are characterised by dense traffic flows,

congestions and queues. Thus, the authors propose an enhanced version of the algorithm presented in [55], which is able to handle cluttered scenes, including slow or stationary vehicles. This algorithm avoids integrating pixel intensities - belonging to foreground vehicles - into the background model, while at the same time preventing the model from becoming obsolete.

2.1.2.2 Polyocular Systems

The most common polyocular system is a binocular (stereo vision) system, consisting of two cameras that capture the same scene from two different (but overlapping) viewpoints. Features in the scene appear to move horizontally between the left and right images. The left and right image pixels corresponding to a 3D feature are found through a process called *correspondence*. Assuming that the stereo system is calibrated, the 3D location of the feature can be found by triangulation. Figure 2.3 shows a simple 2D example. A 3D point $P = (x, y, z)$ is projected onto left and right image pixels with column coordinates x_l and x_r respectively. Referring to Figure 2.3, PMC_l and p_lLC_l are similar triangles and, therefore, the following relationship is obtained:

$$\frac{x}{z} = \frac{x_l}{f} \tag{2.1.4}$$

where f is the lens focal length. A similar relationship is obtained from triangles PNC_r and p_rRC_r:

$$\frac{x-b}{z} = \frac{x_r}{f} \tag{2.1.5}$$

where b is the distance between the two cameras, also known as the *baseline distance*. From Equations (2.1.4) and (2.1.5), the following relationship is derived:

$$z = \frac{bf}{x_l - x_r} \tag{2.1.6}$$

where $x_l - x_r$ is the difference between the column coordinates of the pair of corresponding pixels. This quantity is defined as the *disparity*. The disparity values of different pixels can be plotted on a map (called the *disparity map*) which has the same dimensions as one of the stereo images.

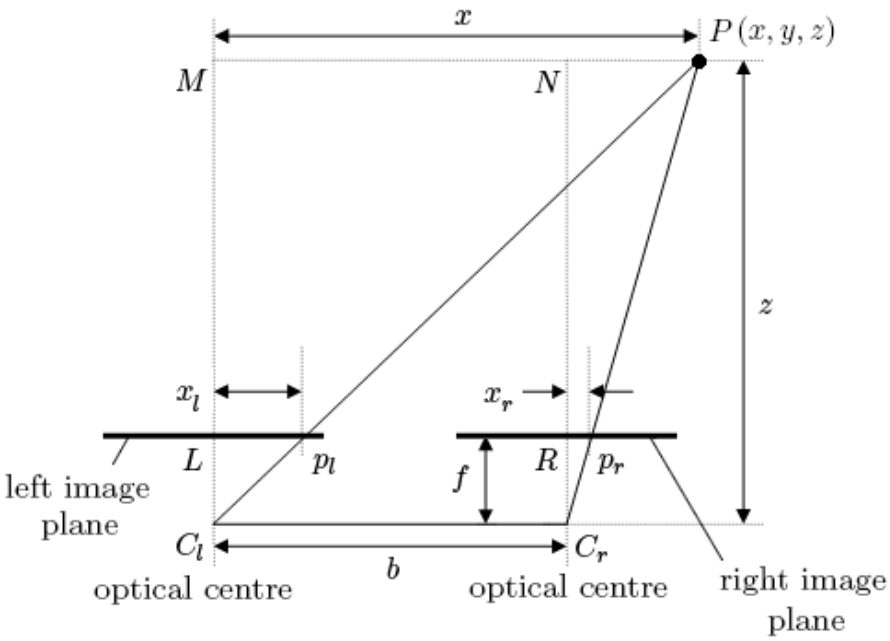


Figure 2.3: Projection of a scene point onto stereo images

Stereo vision can be used to detect both moving and stationary objects, even if the camera itself is stationary. Bertozzi et al [57] present a stereo vision system that is installed behind the windshield of a car in order to detect and track the preceding vehicle. Vehicle detection is carried out in the left image and is based on the fact that the rear of a vehicle is generally symmetrical in the vertical direction and can be characterised by a bounding box with specific aspect ratio constraints. Regions of vertical symmetry are detected by computing histograms of intensity edges and gray level intensities. After an area of symmetry is detected, the bounding box is formed by searching for the four corners of the rear of the vehicle. Then, an initial estimate of the distance to the vehicle is obtained from the position and size of the bounding box. This distance estimate is used to restrict the search for a similar symmetrical region and its bounding box in the right image. Once the right bounding box is obtained, the horizontal offset (disparity) between the left and right bounding boxes is found and the distance to the preceding vehicle is calculated using a single triangulation. The detected vehicle is then tracked in subsequent frames. One problem associated with this method of vehicle detection is that the assumption of symmetry can be violated

by the presence of strong reflections that reduce the vehicle's symmetry in the image plane. The proposed system runs in real-time on a general purpose processor.

Stereo vision is also used for the purpose of obstacle detection in rough terrain. Anderson et al [58] propose a system for the detection of obstacles around an autonomous vehicle in an offroad environment. Here, the ground surface cannot be modeled as a plane or ramp and obstacles are defined as any areas, such as steep slopes, that cannot be traversed by the vehicle. First, corresponding pairs of pixels are projected in 3D and a 3D point cloud is obtained. Then, obstacles are detected using the principle of 'compatible' points. Two 3D points are considered to be compatible if the difference in height between them is within certain limits and if the line joining the two points makes an angle with the horizontal plane which is larger than a certain threshold. Moreover, two 3D points are considered to belong to the same obstacle either if they are compatible or if there is a chain of compatible point pairs linking the two points. The limits and thresholds used for the detection of compatible points are related to certain physical properties and manoeuvring capabilities of the vehicle. For instance, the angular threshold represents the steepest slope that the vehicle is able to climb.

As mentioned previously, stereo vision configurations are used in most cases. However, there are certain applications that employ three or more cameras. These camera setups are referred to as *multibaseline* systems. One such system is implemented by Broggi et al [59]. Three cameras are mounted horizontally on a vehicle that is required to navigate autonomously in an offroad environment. Two cameras are placed 0.5m apart and the third camera is 1m away from the second camera. Only two of these cameras are used during any particular frame, depending on the speed of the vehicle. As observed from Equation (2.1.6), the larger the baseline distance, the larger the disparity associated with a particular object at a certain depth. This means that the system is more capable of detecting distant obstacles. As the speed of the vehicle increases, the braking distance required to stop the vehicle in the event of a conflict increases as well. Hence, the detection range

needs to be extended. Consequently, a larger baseline distance (up to a maximum of 1.5m when the outermost cameras are selected) is used when the vehicle speed increases. However, increasing the baseline distance results in greater differences in the appearance of objects between the left and right images and reduces the FOV of the system. For this reason, the baseline distance is reduced at lower vehicle speeds. The system runs at 15Hz on a general purpose processor.

A different trinocular system is proposed by Williamson [38]. The system is designed to detect very small obstacles (approx. 15cm high) up to a distance of around 100m in front of a vehicle. The cameras are mounted on top of the vehicle and are arranged in a triangular configuration, with one of the cameras situated vertically above the other two. All of the cameras are used in each frame. The advantage of using three cameras simultaneously is that more measurements can be made and, therefore, there is a better chance of obtaining reliable matches when locating corresponding features between each pair of cameras. This reduces the triangulation errors, improves the range accuracy, and increases the detection range. The use of a horizontal baseline and a vertical baseline means that the correspondence algorithm can take advantage of image texture in any direction. This system was implemented on a general purpose processor and had an update rate of about 1Hz.

2.1.2.3 Hybrid Systems

Certain systems use a combination of optical techniques, the most common being optical flow and stereo vision. For instance, Hrabar [60] proposes a system that uses optical flow and stereo vision to detect and avoid obstacles in the flight path of an autonomous helicopter during urban navigation. Forward-looking stereo cameras are mounted in front of the helicopter and a sideways-looking camera is installed on each side. The stereo cameras detect obstacles ahead of the helicopter whereas the other cameras detect obstacles to the sides, by measuring the optical flow on each side of the helicopter. The stereo vision and optical flow processes operate independently and their outputs are processed by two different control schemes, each of which produces

a turn rate command to move the helicopter away from any detected obstacles. The system determines which control command to use depending on the proximity of obstacles to the front and to the sides of the UAV. With this control strategy, the UAV is able to navigate through urban streets with various types of junctions. The system was tested using both simulations and real images.

Sull and Sridhar [61] propose a different method of combining optical flow and stereo vision. Their method is designed to detect obstacles on a runway during autonomous aircraft landings. The runway is modeled as a planar surface. In order to detect obstacles that are higher than a certain threshold, onboard sensors (such as the inertial navigation unit) are used to predict the optical flow and stereo disparity of obstacles with a height equal to the threshold. Then, for each left image pixel corresponding to the runway, the algorithm checks whether the magnitude of the predicted optical flow is greater than a certain limit. For those pixels whose predicted optical flow exceeds this limit, the algorithm uses the current frame and the previous frame in order to measure the actual optical flow. For the rest of the pixels, the algorithm can increase the temporal baseline (to improve the detection range of the system as explained in Section 2.1.2.1) until the predicted optical flow exceeds the desired limit. The larger temporal baseline is then used to measure the actual optical flow of these pixels. If certain pixels still do not have measurable optical flows, such as in the region close to the FOE, the algorithm can use the disparity provided by stereo vision.[3] Obstacles are detected by checking if the magnitude of the measured optical flow of each pixel exceeds the predicted optical flow. The system was tested with a number of real image sequences captured by stereo cameras mounted on a helicopter flying along a runway. The obstacles appearing in the images consisted of moving and stationary trucks. The results obtained show that the proposed algorithm is indeed capable of detecting distant obstacles by dynamically adjusting the temporal baseline. The algorithm runs on a four-processor Silicon Graphics® server at several

[3]At the time of the publication of [61], the part of the detection method based on stereo vision was not yet implemented.

frames per second.

In the system proposed by Hrabar [60], the outputs of optical flow and stereo vision are fused at a high level (the decision level). However, Mills [62] points out that optical flow and stereo vision are closely related to each another and that these two processes can be fused together at a lower level. Mills observes that there is a link between two corresponding pixels in a stereo image pair and their corresponding pixels in a subsequent frame. The corresponding left and right image pixels must undergo the same motion from one frame to another. Since the 3D position of points is obtained by stereo vision and these points are tracked using optical flow, it is possible to obtain a model to estimate the 3D motion of these points. In contrast, it is only possible to describe the 2D motion of image features when using a single camera. Therefore, the combination of stereo vision and optical flow avoids the loss of motion information.

The close link between optical flow and stereo vision is exploited by Franke and Heinrich [63]. The authors propose a fusion method for the purpose of the detection of moving obstacles on the road, such as vehicles and pedestrians. Images are captured by a pair of forward-looking cameras installed inside a vehicle. The fusion method introduces constraints in the obstacle detection process by taking advantage of the fact that stereo vision and optical flow are related by real-world depth. In fact, as can be observed from Equations (2.1.1) and (2.1.6), the magnitude of the optical flow of a pixel and the stereo disparity of the same pixel are both inversely proportional to depth. The authors derive expressions which directly relate the horizontal and vertical components of the optical flow with disparity. These expressions are used to construct a 'flow/depth' plane, where the value of each point on the plane is equal to the quotient of the optical flow and disparity of an image pixel corresponding to a stationary feature in the scene. The inclination of this plane is adjusted dynamically according to vehicle speed. In each frame, stereo correspondence and optical flow estimation are carried out and quotient values are obtained for each image pixel. Moving objects are easily detected because the quotient values of the pixels corresponding to such objects

deviate from the plane. The expressions relating optical flow and disparity are based on the assumption that the camera undergoes pure translational motion. However, in practice, the camera also undergoes some rotational motion. Therefore, the pitch and yaw movements are corrected online using an image stabilisation technique. The complete system runs in real-time on a general purpose processor. However, due to the optical flow technique used, it only works well at low vehicle speeds (less than 25km/h) and has a detection range of about 30m.

2.2 System Overview

2.2.1 The Selected Technology

For this application, the system needs to be able to detect both moving and stationary objects, irrespective of ownship speed. Also, it is necessary to determine the 3D position of obstacles with respect to the ownship in order to determine if the protection zone (defined in Section 1.3) is penetrated. A monocular system based on optical flow techniques would not provide satisfactory results in this case because of its dependence on the relative motion between the camera and obstacles. Obstacles near the FOE are very difficult to detect and limited positional information can be extracted from the optical flow pattern. Moreover, as explained in Section 2.1.2.1, optical flow techniques are very susceptible to shocks and vibrations, which degrade the quality of the optical flow pattern.

On the other hand, a polyocular system is able to detect both stationary and moving obstacles and can provide 3D positional information. Motion information can also be exploited by tracking the detected obstacles in 3D. The use of a trinocular system, such as the one proposed by Williamson [38], is not necessary in this case because the obstacles that need to be detected in this application are generally much larger than the smallest obstacles that need to be detected in [38]. As a result, for this application, the detection range of a stereo vision system could potentially be similar to that of the system proposed in [38]. The use of an alternate multibaseline system,

where the baseline distance increases or decreases with vehicle speed (such as the one proposed by Broggi et al [59]), is also not preferred because, in this application, it is important to maximise the detection range at all times in order to be able to detect potential collision threats as early as possible, before obstacles penetrate the protection zone. Therefore, the use of a stereo vision system, with the largest affordable baseline distance, is preferred.

On the basis of the arguments presented above, it was decided to implement a stereo vision-based obstacle detection and tracking system for this application.

2.2.2 System Functionalities

Figure 2.4 represents the main functional blocks of the stereo vision system and the following is an overview of each of these blocks.

Calibration is an offline process that is composed of three stages: intrinsic calibration, relative extrinsic calibration, and absolute extrinsic calibration. Intrinsic calibration determines the lens parameters of each camera whereas relative extrinsic calibration determines the geometry between the two cameras. Absolute extrinsic calibration determines the geometry between the cameras and a common reference frame within which the position of obstacles is reported.

Rectification is a preprocessing stage that removes lens distortion and compensates for any misalignments between the cameras. The rectification parameters are fixed for a particular stereo setup and are calculated offline as part of the calibration process. The main aim of rectification is to simplify the correspondence problem by ensuring that corresponding pixels lie on the same row of the left and right images.

Correspondence is the process that finds corresponding pixels in the left and right images. Since the images are rectified, correspondence is reduced to a 1D problem. Correspondence can be carried out on the whole image or on specific features. The output of this process is a disparity map.

Reconstruction involves the determination of the coordinates of a 3D point from the 2D coordinates of its projection in the left and right images. This process uses

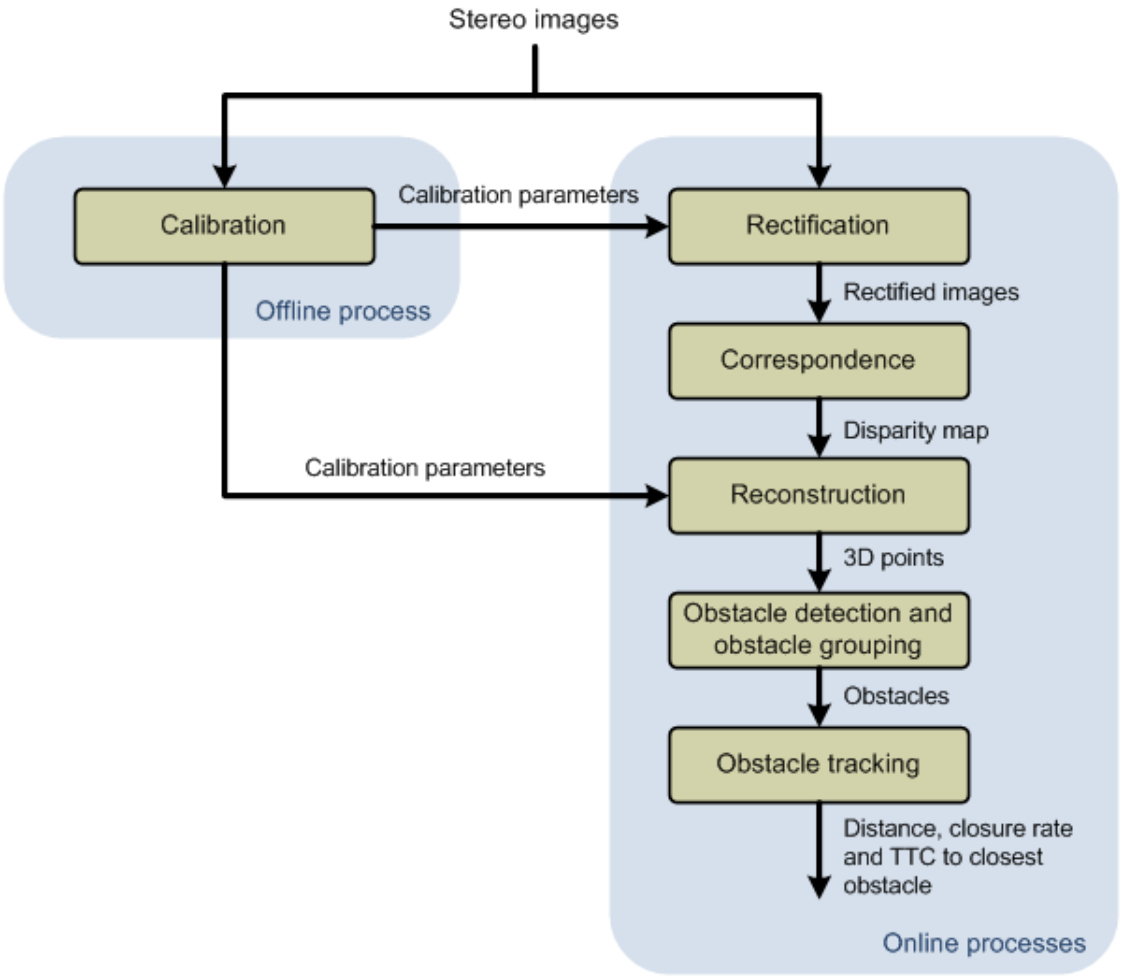


Figure 2.4: Functional block diagram of the stereo vision system

knowledge of the stereo geometry (obtained during calibration) in order to recover 3D information. The output of reconstruction is a 3D point cloud.

Obstacle detection is the process of classifying 3D points as either belonging to ground features or to any obstacles above the ground. The detected obstacle points are grouped into obstacle regions (through a process called *clustering*) on the basis of different criteria such as spatial proximity.

Obstacle tracking involves monitoring the state of obstacles over time. The main objectives of this process are to obtain better position estimates, to determine the closure rate between the ownship and obstacles, and to predict the TTC if a collision is imminent.

The functions described above are typical of many stereo vision systems. However, the implementation of each function can differ significantly from one system to another, depending on the application. The choice of a particular algorithm depends on several factors, including:

- assumptions about the operational environment
- quantity of obstacles and obstacle properties
- platform speed and position (fixed or moving, on the ground or in the air)
- algorithm complexity and available processing power
- required system update rate, detection rate and accuracy

Research on stereo vision to date has resulted in many algorithms being developed to implement different stages of an obstacle detection and tracking system. However, the specific application of obstacle detection (particularly of aircraft extremities) in ramps and taxiways for the purpose of wingtip collision prevention, has not yet been addressed in the literature. As is discussed further in the rest of the thesis, this application poses several challenges due to the wide variety of obstacle shapes and sizes that need to be detected. One of the biggest challenges is to reliably detect aircraft extremities, such as wings and wingtips, when viewing them from different directions. Therefore, due to the nature of the application, the processing blocks of the stereo vision system could not be implemented simply by applying methods and techniques that are already available in the literature. Instead, whenever existing methods and techniques were applied, these had to be modified and adapted to meet the specific requirements of the application.

Major modifications as well as new developments were carried out on two particular algorithms: correspondence and clustering. The correspondence algorithm uses area-based correlation techniques and computes the disparity of edge pixels with sub-pixel precision. Several constraints and confidence criteria are applied to ensure that the disparity values obtained are reliable. In order to reduce the

computation time, a modified multiresolution scheme is proposed in this work. This reduces the processing time (in comparison with the time taken to process the full resolution images directly) while avoiding the problems associated with a standard multiresolution approach.

The clustering algorithm is based on an agglomerative, hierarchical clustering technique. A new method of grouping and filtering obstacle points, on the basis of multiple weighted criteria, is proposed. This method makes use of thresholds that adjust dynamically according to the distance of obstacles from the cameras, in such a way as to increase the detection range of the system while preventing false detections.

2.2.3 Camera Placement

Camera placement is a very important consideration as it affects the overall performance of the system. Naturally, the cameras are required to cover the intended protection zone around the wingtips. Consequently, two main configurations were considered as depicted in Figure 2.5:

1. Placing a pair of cameras on each wingtip (Figure 2.5(a)): One of the advantages of this setup is that the positional accuracy of the system increases as an obstacle get closer to the wingtips. At the same time, the apparent size of the obstacle in the image will increase, improving its detectability. The downside of this is that part of the obstacle can fall outside the common camera FOV, depending on its actual size.

   Since the cameras are located on the wingtips, it is possible to detect obstacles beyond the range defined by the protection zone. This can be particularly advantageous for obstacle tracking.

   Another advantage is that, when the obstacle is an aircraft, it is generally easier to detect the aircraft's extremities. This is because, due to the geometry of the most common conflict scenarios (as shown in Figure 1.3), the image of the wing, wingtip or tail is likely to contrast with the background. An example of this is

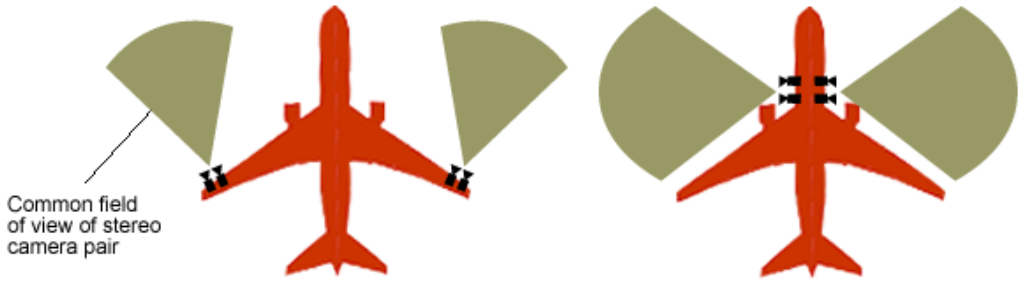
shown in Figure 2.6(a) where the ownship is moving parallel to another aircraft.

One of the main disadvantages of this setup is that the cameras are prone to random vertical movements due to wing bending. This can affect the performance of the system. Also, a logistical problem can arise as the wings may have very little space available due to equipment such as wingtip lights and moving surfaces. This may introduce limitations on the baseline distance and on camera placement on the wing.

2. Placing a pair of cameras on either side of the fuselage (Figure 2.5(b)): One of the advantages of this setup is that there is more flexibility in camera placement and in the selection of baseline distance. Another advantage is that, since the fuselage is more rigid than the wings, the cameras will be less susceptible to fluctuations.

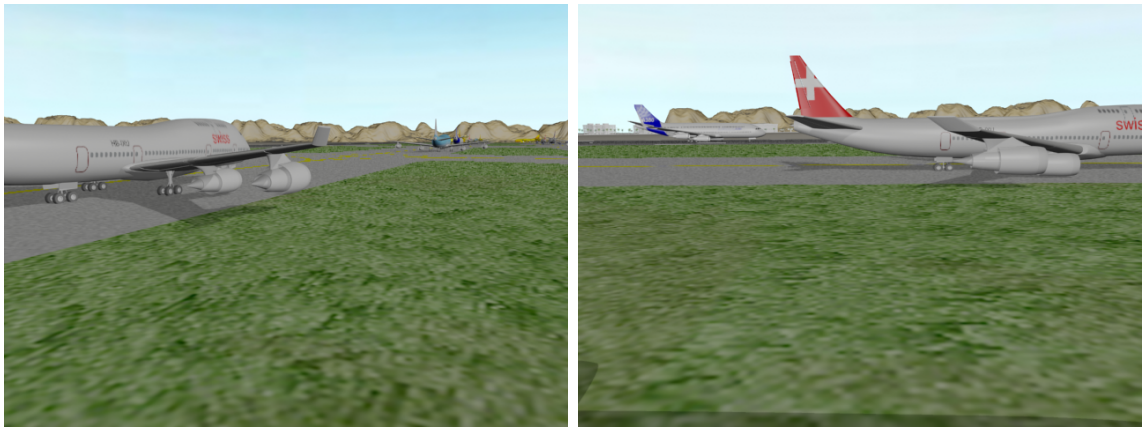
   A disadvantage of this setup, however, is that in certain scenarios where the obstacle is an aircraft, the aircraft's extremities can be hard to detect. For instance, if the ownship is moving parallel to another aircraft, the image signal corresponding to the wing and wingtip of the other aircraft will be smaller than that obtained with the wing-mounted camera setup and it is very likely that the image background of the wing will be the fuselage (Figure 2.6(b)). This will reduce image contrast.

   Another drawback of this configuration is that, since the cameras are further away from the protection zone than in the first case, a larger baseline distance and/or focal length will be required to achieve the same range resolution and positional accuracy as with the first setup. Increasing the baseline distance or focal length will reduce the common camera FOV and potentially affect the ability of the system to monitor the whole of the protection zone, particularly the area directly surrounding the wingtips. This problem also increases with wing sweepback.



(a) (b)

Figure 2.5: Camera placement options: (a) wingtips and (b) fuselage



(a) (b)

Figure 2.6: Images of an aircraft from two different viewpoints: (a) wingtip camera and (b) camera mounted on the fuselage

After weighing the advantages and disadvantages offered by these options, the first option was selected, with a pair of cameras mounted close to each wingtip. As each camera pair is intended to operate independently, the rest of this thesis focusses on one (the left) wingtip stereo vision system. Table 2.2 contains the stereo vision parameters that were selected for this application. The selection of the baseline distance and the lens focal length are discussed in detail in Chapter 5, as different combinations of baseline distance and focal length were tested to determine the most suitable configuration.

Table 2.2: Stereo vision parameters

Baseline distance	$1.5m$
Camera height above the ground	$8m$
Lens focal length	$32mm$
Horizontal FOV	$60°$
Vertical FOV	$47°$
Image resolution	640x480 pixels
Frame rate	15 frames/second

Most of the visible cameras used in computer vision systems are monochrome cameras that produce grayscale images. These cameras are generally faster and have a higher resolution than color cameras with the same price tag. Also, grayscale images can be processed much more quickly than color images. Therefore, color cameras are normally limited to applications where color detection can significantly improve the performance of the system. Due to the wide use of monochrome cameras, a lot of research is focussed on grayscale images and most computer vision algorithms are tailored for this type of image. For these reasons, the cameras used in this work are also monochrome.

2.2.4 Synthetic Image Generation

In order to be able to test the individual stages of the stereo vision system and to check the performance of the overall system, a simulation environment was set up to replicate a typical aerodrome, particularly the ramp and taxiway regions. This was done using Autodesk 3ds Max® which is a 3D modeling, animation and rendering software package. The ownship was simulated by using a model of an A380 and a pair of virtual cameras (with exactly the same parameters as those given in Table 2.2) were placed close to the left wingtip of the ownship.

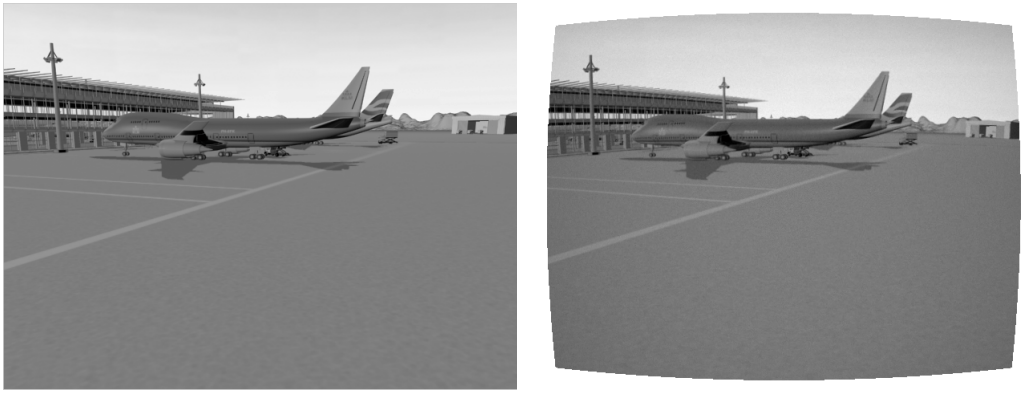
The left and right stereo images captured by the cameras were post-processed in order to add some of the characteristics of camera sensors and lenses that degrade image quality, namely:

- **Lens distortion:** Radial lens distortion was applied in different quantities to

the left and right stereo images. The distortion parameters were determined during calibration (Refer to Table 3.1).

- **Vignetting:** This is the gradual reduction of pixel brightness away from the centre of an image. It is mostly observed at the periphery of an image because the light rays are spread over a larger sensor area than the rays in the middle (especially in the case of large apertures). Vignetting was added by multiplying the image pixel values by weights obtained using the cos^4 law described in [64].

- **Temporal image noise:** This is caused by sensor noise and results in fluctuations of the intensity value of each pixel. Image noise was introduced by adding Additive White Gaussian Noise (AWGN) with a mean of 0 and a standard deviation of 3 intensity levels.[4] To ensure that the left image noise was uncorrelated with the right image noise, the random number generator was initialised to a different state for each image.



(a) (b)

Figure 2.7: An image (a) before post-processing and (b) after post-processing

Figure 2.7 shows an image before and after post-processing. By comparing both images, it is difficult to notice the temporal noise in the post-processed image, due to the relatively small amplitude of the noise. However, it is possible to observe that the

[4]The maximum intensity level of a pixel in a grayscale image is 255.

post-processed image is warped due to radial lens distortion. The subtle darkening of pixels due to vignetting is also visible, particularly at the corners of the image.

Unless specified otherwise, the stereo images used throughout this thesis were generated using the simulation environment and virtual camera setup described in this section. Consequently, most of the results presented in the next chapters are based on simulation experiments.

Chapter 3

Calibration

This chapter addresses the calibration process designed and implemented for the system. Section 3.1 introduces the different reference frames and the relationship between them and describes the camera model used for calibration. Section 3.2 discusses the algorithms that are used for intrinsic and relative extrinsic calibration and presents some calibration results. Similarly, Section 3.3 describes the algorithm that is used for absolute extrinsic calibration and presents more calibration results.

3.1 Reference Frames and their Relationships

There are four different coordinate frames which are referred to throughout this thesis, namely: the World Reference Frame (WRF), Camera Reference Frame (CRF), Aircraft Reference Frame (ARF) and Image Reference Frame (IRF). Section 3.1.1 describes the relationship between the first three of these reference frames whereas Section 3.1.2 introduces the camera model and describes the relationship between the IRF, CRF and WRF.

3.1.1 The WRF, CRF and ARF

Figure 3.1 shows the WRF, CRF and ARF. These are all right-handed Cartesian coordinate systems with orthogonal axes. The origin of the ARF is located on the ground surface, right in the middle of the aircraft. The x axis is aligned horizontally across the aircraft and the z axis is aligned along the fuselage. The y axis points

vertically downwards.[1] The position of obstacles (detected by the cameras) is reported in the ARF.

The origin of the WRF is also a point on the ground, with the xz plane parallel to the ground surface and the y axis pointing vertically downwards. For convenience, the origin of the WRF is selected to be the point on the ground halfway between the left and right cameras as shown in Figure 3.2. However, in practice, the WRF can be made to coincide with the ARF by shifting the former's origin. It is important to note that the origin of the WRF is not a fixed point on the aerodrome surface but moves along with the ownship.

The origin of the left and right CRFs is located in the left and right cameras respectively. The CRF is explained in more detail in Section 3.1.2.

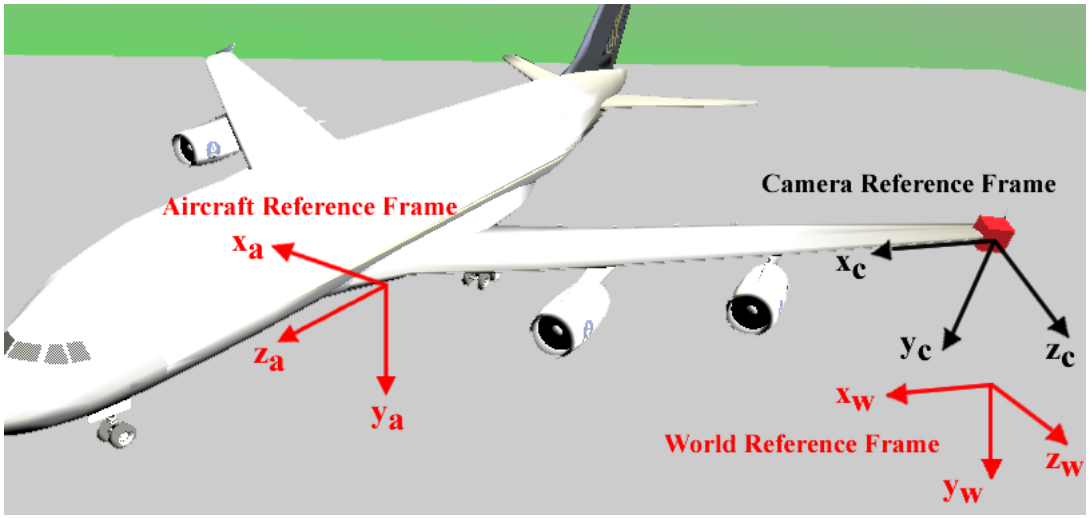


Figure 3.1: Reference frames (1)

The relationship between a point $P_l = (X_c, Y_c, Z_c)^T$ in the left CRF and a point $P_w = (X_w, Y_w, Z_w)^T$ in the WRF is given by the following rigid motion transformation:

$$\begin{aligned} P_l &= R_l P_w + T_l \\ \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} &= \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} + \begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix} \end{aligned} \tag{3.1.1}$$

[1] The positive y axis is below the ground surface.

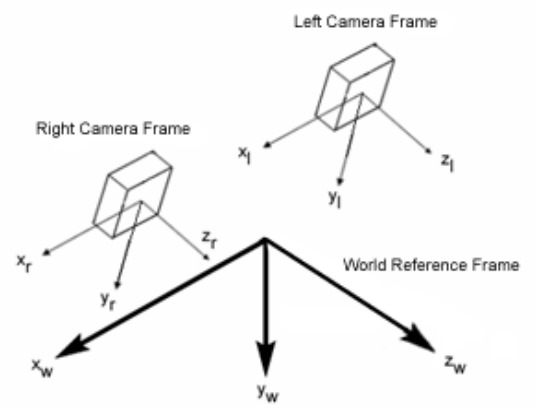


Figure 3.2: Reference frames (2)

where R_l and T_l are the rotation matrix and translation vector respectively. Similarly, the relationship between a point P_r in the right CRF and point P_w in the WRF is given by:

$$P_r = R_r P_w + T_r \tag{3.1.2}$$

The relationship between the left and right cameras is derived from Equations (3.1.1) and (3.1.2), by eliminating P_w as follows:

$$\begin{aligned} P_r &= R_r[R_l^{-1}(P_l - T_l)] + T_r \\ &= R_r R_l^{-1} P_l - R_r R_l^{-1} T_l + T_r \end{aligned}$$

Let $R_{rel} = R_r R_l^{-1}$ and $T_{rel} = T_r - R_{rel} T_l$.

$$\implies P_r = R_{rel} P_l + T_{rel} \tag{3.1.3}$$

T_{rel} and R_{rel} describe the position and orientation of the left camera with respect to the right camera, respectively.

The relationship between a point P_l in the left CRF and a point $P_a = (X_a, Y_a, Z_a)^T$ in the ARF is given by:

$$P_a = R_a P_l + T_a \tag{3.1.4}$$

where R_a and T_a are the rotation matrix and translation vector respectively. Using Equation (3.1.1) and eliminating P_l from Equation (3.1.4), the ARF and WRF are

related as follows:

$$\begin{aligned} P_a &= R_a(R_l P_w + T_l) + T_a \\ &= R_a R_l P_w + R_a T_l + T_a \end{aligned}$$

Let $R_2 = R_a R_l$ and $T_2 = R_a T_l + T_a$.

$$\implies P_a = R_2 P_w + T_2 \tag{3.1.5}$$

Each rotation matrix mentioned in this section is a Direction Cosine Matrix (DCM) of the following form:

$$\begin{aligned} R &= R(\psi)R(\phi)R(\theta) \\ &= \begin{pmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} cos\phi & 0 & sin\phi \\ 0 & 1 & 0 \\ -sin\phi & 0 & cos\phi \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{pmatrix} \\ &= \begin{pmatrix} cos\psi cos\phi & cos\psi sin\phi sin\theta - sin\psi cos\theta & cos\psi sin\phi cos\theta + sin\psi sin\theta \\ sin\psi cos\phi & sin\psi sin\phi sin\theta + cos\psi cos\theta & sin\psi sin\phi cos\theta - sin\theta cos\psi \\ -sin\phi & sin\theta cos\phi & cos\phi cos\theta \end{pmatrix} \end{aligned} \tag{3.1.6}$$

where R represents a rotation of θ about the x axis, followed by a rotation of ϕ about the y axis, followed by a rotation of ψ about the z axis. Since the reference frames satisfy the right-hand rule, R corresponds to anticlockwise rotations about the x, y and z axes respectively.

3.1.2 The Camera Model

Figure 3.3 shows the geometry of a *pinhole* camera model. This model assumes that no lenses are used and that the camera aperture is a point. It can be noted that the origin of the CRF is the location of the pinhole (the projection centre) whilst the origin of IRF is at the upper left corner of the image that is generated by the camera sensor. The z axis of the camera frame is perpendicular to the image plane and the point of intersection between the two is called the *principal point*. The point $P_l = (X_c, Y_c, Z_c)^T$

in the CRF and the point $p_l = (x_p, y_p)^T$ in the IRF are related through perspective projection as follows (refer to plan view and side elevation respectively):

$$x_p = \frac{f_x X_c}{Z_c} + c_x = f_x x_n + c_x$$

$$y_p = \frac{f_y Y_c}{Z_c} + c_y = f_y y_n + c_y \tag{3.1.7}$$

where:

$f = (f_x, f_y)$ is the focal length in terms of pixel dimensions in the x and y directions,

$c = (c_x, c_y)$ is the principal point expressed in pixel coordinates,

(x_n, y_n) is the normalised image projection.

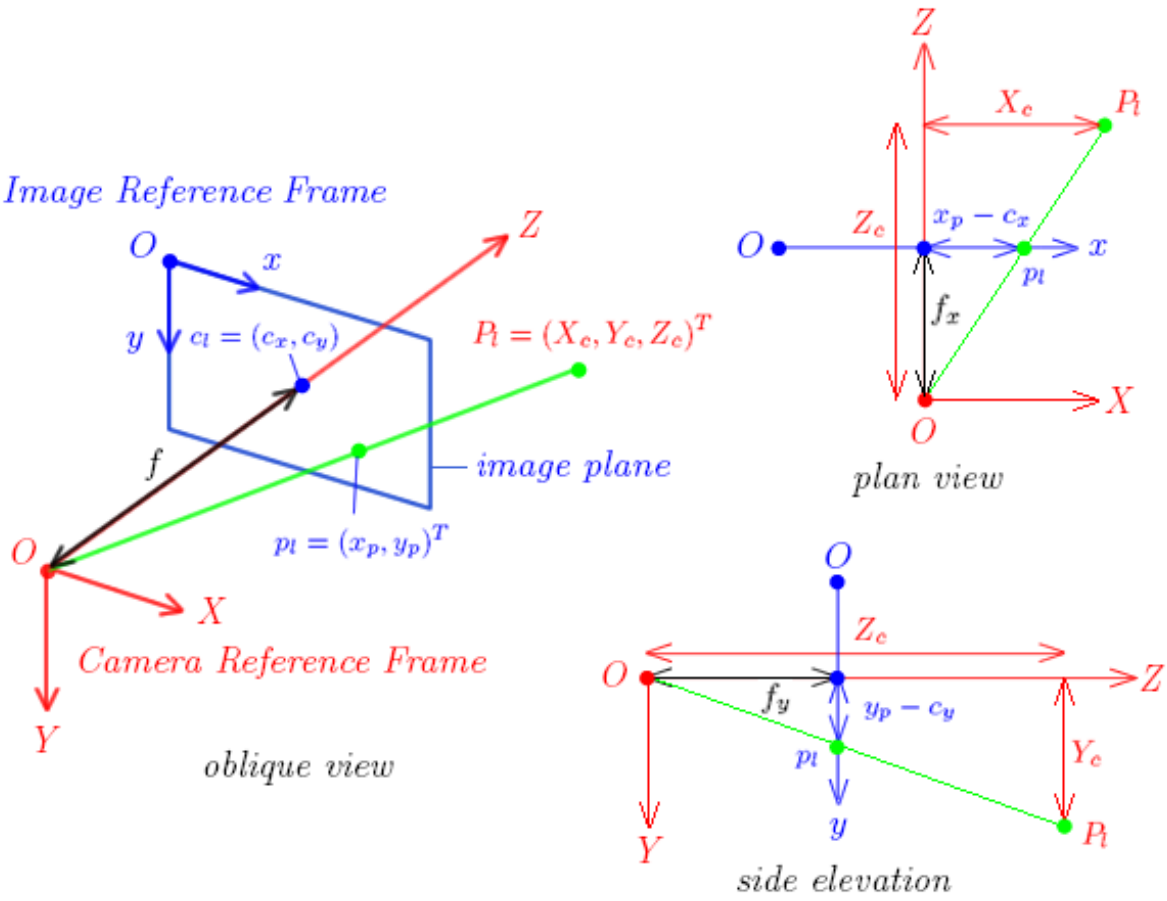
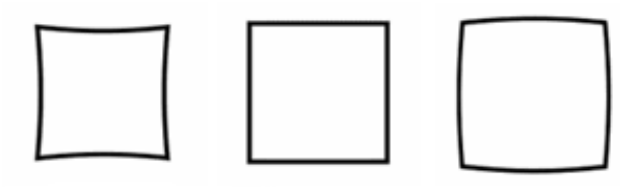


Figure 3.3: The pinhole camera model

A more accurate camera model is obtained from the pinhole camera model by taking lens distortion into account. Lens distortion has two components: radial and

tangential. Radial distortion bends light rays depending on their distance from the principal point. It can be subdivided into two types: barrel distortion and pincushion distortion (Figure 3.4). Tangential distortion is due to non-collinearity of the centre of curvature of the lens surfaces and results in decentring.



(a) (b) (c)

Figure 3.4: Radial lens distortion: (a) pincushion distortion, (b) no distortion, (c) barrel distortion

Let $r^2 = x_n^2 + y_n^2$. The distorted normalised projection (x_d, y_d) is obtained from the normalised image projection as follows [65, 66]:

$$\begin{aligned} x_d &= (1 + k_1 r^2 + k_2 r^4 + k_5 r^6) x_n + 2k_3 x_n y_n + k_4 (r^2 + 2{x_n}^2) \\ y_d &= (1 + k_1 r^2 + k_2 r^4 + k_5 r^6) y_n + k_3 (r^2 + 2{y_n}^2) + 2k_4 x_n y_n \end{aligned} \tag{3.1.8}$$

where $k = (k_1..k_5)$ is a vector of the radial and tangential distortion coefficients. Another parameter that has to be considered is the skew α. This parameter varies as a function of the angle between the x and y axes of the image plane. α is 0 if the x and y axes are perfectly orthogonal. Substituting (x_d, y_d) for (x_n, y_n) in Equation (3.1.7) and taking skew into account, we obtain:

$$\begin{aligned} x_p &= f_x(x_d + \alpha y_d) + c_x \\ y_p &= f_y y_d + c_y \end{aligned} \tag{3.1.9}$$

Equation (3.1.9) can be expressed in matrix form as follows:

$$\begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & \alpha f_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_d \\ y_d \\ 1 \end{pmatrix} = A \begin{pmatrix} x_d \\ y_d \\ 1 \end{pmatrix} \tag{3.1.10}$$

where A is known as the *intrinsic camera matrix*. Assuming that the lens distortion is 0, Equation (3.1.8) is reduced to $x_d = x_n$ and $y_d = y_n$. Then, Equation (3.1.10) becomes:

$$\begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = A \begin{pmatrix} x_n \\ y_n \\ 1 \end{pmatrix} = A \begin{pmatrix} \frac{X_c}{Z_c} \\ \frac{Y_c}{Z_c} \\ 1 \end{pmatrix} \tag{3.1.11}$$

By combining Equations (3.1.1) and (3.1.11), the CRF can be bypassed to provide a direct relationship between the IRF and the WRF:

$$\begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = A \begin{pmatrix} \frac{X_c}{Z_c} \\ \frac{Y_c}{Z_c} \\ 1 \end{pmatrix} = A \begin{pmatrix} \frac{r_{11}X_w + r_{12}Y_w + r_{13}Z_w + T_x}{r_{31}X_w + r_{32}Y_w + r_{33}Z_w + T_z} \\ \frac{r_{21}X_w + r_{22}Y_w + r_{23}Z_w + T_y}{r_{31}X_w + r_{32}Y_w + r_{33}Z_w + T_z} \\ 1 \end{pmatrix} \tag{3.1.12}$$

Equation (3.1.12) can be rearranged as follows:

$$s \begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = A(R_l|T_l) \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = AE \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = H \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \tag{3.1.13}$$

where:

s is a non-zero scalar factor equal to $r_{31}X_w + r_{32}Y_w + r_{33}Z_w + T_z$,

E is known as the *extrinsic matrix*,

H is known as the *projection matrix* or *homography*.

3.2 Intrinsic and Relative Extrinsic Calibration

The objective of intrinsic calibration is to determine the camera parameters in intrinsic matrix A and the lens distortion parameters $k_i|i = 1..5$. On the other hand, the objective of relative extrinsic calibration is to find the relative position T_{rel} and orientation R_{rel} between the left and right cameras.

In order to find the intrinsic and relative extrinsic parameters, plane-based calibration is used. This type of calibration is widely used by the computer vision community and several implementations are available [67–70]. The one used in this

work is Bouguet's camera calibration toolbox [66]. This toolbox supports stereo calibration and rectification and can estimate all of the intrinsic and relative extrinsic camera parameters. It is very well documented and is easy to use. The toolbox and its source code are freely available online. The mathematics presented in this section describes algorithms which are part of this toolbox and have been documented by others.

To carry out calibration, a planar checkerboard pattern (shown in Figure 3.5) is used. The x and y axes of this calibration object are aligned with the vertical and horizontal edges of the pattern respectively and the z axis is perpendicular to the surface. The origin is located at the top-left corner of the pattern. The squares on the pattern are identical to each other and the corners are used as control points.

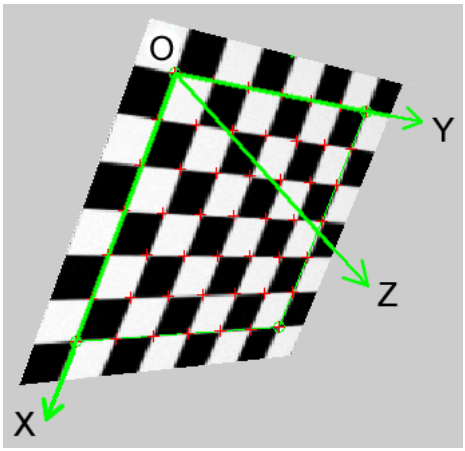


Figure 3.5: 2D calibration pattern

The input to the calibration routine consists of a number of images that are obtained by viewing the planar pattern from different positions and orientations with respect to the stereo setup (Figure 3.6). In order to extract the control points from each of the calibration images, a software routine is used. First, the user selects the boundary of the calibration pattern in each image. Then, the software routine uses a corner detector to detect control points within the selected boundaries and to determine their pixel coordinates. The user then inputs the dimensions (in mm) of one of the squares on the pattern. These are used to express the coordinates of the control points in the coordinate system of the calibration object.

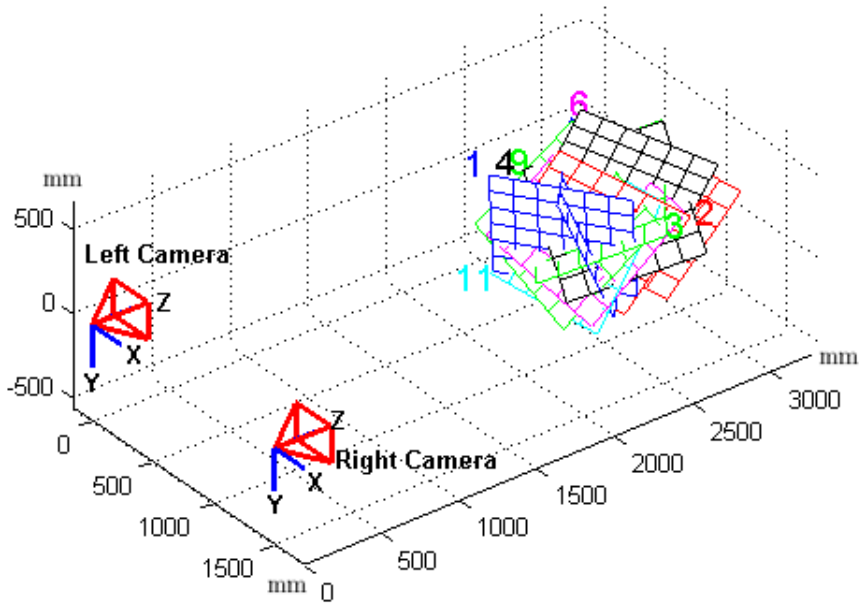


Figure 3.6: Arrangement of stereo cameras and calibration images

The left and right cameras are calibrated separately. However, in order to be able to determine the relative position and orientation between them, it is necessary that corresponding pairs of calibration images are used for the left and right cameras.

3.2.1 Intrinsic Calibration

The intrinsic calibration routine consists of two phases: an initialisation stage and an optimisation stage.

3.2.1.1 Initialisation of intrinsic parameters

The intrinsic parameters are initialised as follows:

- Principal point - This is initialised at the centre of the image.
- Distortion coefficients - It is assumed that the camera has no lens distortion. Therefore, the distortion coefficients are set to 0.
- Skew - It is assumed that the image axes are perpendicular. Therefore, the skew is set to 0.
- Focal length - This is estimated in a two-step process:
  1. The planar homography between the calibration object and each calibration image is estimated as proposed in [67] and as explained below.
  2. From the computed homographies, the focal length is found using the principle of orthogonality of vanishing points as proposed in [71]. This method is described in Appendix B.1.

The z coordinate of the control points on the calibration pattern is 0. Therefore, from Equation (3.1.13), the following relationship is obtained between a point $\widetilde{m} = (x_p, y_p, 1)^T$ in the image plane and a point $\widetilde{M} = (X, Y, Z, 1)^T$ defined in the coordinate system of the calibration object:

$$s \begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = A \begin{pmatrix} r_1 & r_2 & r_3 & T \end{pmatrix} \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} = A \begin{pmatrix} r_1 & r_2 & T \end{pmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = H \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{3.2.1}$$

where r_i is the i^{th} column of the rotation matrix.

The homography H is estimated using a technique based on the maximum likelihood criterion. Ideally, image points m_i and calibration object points M_i satisfy Equation (3.2.1). However, in practice, this does not occur because the image points are corrupted by noise. Assuming that the points are corrupted by noise with mean 0 and covariance matrix Λ_{m_i}, the maximum likelihood estimation of H is obtained by minimising the following:

$$\sum_i (m_i - \hat{m}_i)^T \Lambda_{m_i}^{-1} (m_i - \hat{m}_i)$$

where:

$\hat{m}_i = \frac{1}{\bar{h}_3^T M_i} \begin{pmatrix} \bar{h}_1^T M_i \\ \bar{h}_2^T M_i \end{pmatrix}$,

$\bar{h}_i$ is the i^{th} row of H normalised by s (This is the scalar factor s defined in Equation (3.1.13)).

In practice it is assumed that $\Lambda_{m_i} = \sigma^2 I$ for all i (where σ is the standard deviation of the noise). Therefore, the above problem becomes a nonlinear least-squares one, i.e. $min_H \sum_i \|m_i - \hat{m}_i\|^2$. The nonlinear minimisation is performed using the Levenberg-Marquardt algorithm. This requires an initial guess for H (in order to find $\hat{m}_i$) which is obtained as follows.

Let $x = (\bar{h}_1^T, \bar{h}_2^T, \bar{h}_3^T)^T$. Then, Equation (3.2.1) can be rewritten as:

$$\begin{pmatrix} \widetilde{M}^T & 0^T & -x_p \widetilde{M}^T \\ 0^T & \widetilde{M}^T & -y_p \widetilde{M}^T \end{pmatrix} x = 0 \tag{3.2.2}$$

For n points there are n equations of the form of (3.2.2) which can be written in matrix form as $Lx = 0$, where L is a $2n \times 9$ matrix. x encapsulates the value s defined in Equation (3.1.13) and is therefore defined up to a scale factor. Hence, the solution (H) is the right singular vector of L associated with the smallest singular value (or the eigenvector of $L^T L$ associated with the smallest eigenvalue). The solution is obtained using Singular Value Decomposition (SVD) [72].

3.2.1.2 Initialisation and refinement of extrinsic parameters

For each calibration image, initial estimates of the extrinsic parameters are obtained by estimating the homography between the normalised image coordinates of the

control points and the coordinates of the points defined in the coordinate system of the calibration object. The homography is computed using the technique (based on the maximum likelihood criterion) just described. The rotation matrix R and translation vector T are extracted from the homography and used as initial values.

R and T are refined by minimising the following function:

$$\sum_{j=1}^{b} \|m_j - \hat{m}(A, k, R, T, M_j)\|^2 \tag{3.2.3}$$

where:

b is the number of control points,

$\hat{m}(A, R, T, M_j)$ is the projection of 3D point M_j in the image according to Equation (3.2.1).

This function is minimised through gradient descent as follows:

1. The corner points of the calibration object are projected from 3D space onto the image plane using Equation (3.2.1). The initial values of the extrinsic parameters are used in the first iteration.

2. The error $\mathbf{e}$ between the actual and projected pixel coordinates of the control points is calculated.[2]

3. The change δ in the extrinsic parameters is found using Equation (3.2.4):

$$\delta = \lambda(J^T J)^{-1} J^T e \tag{3.2.4}$$

   where:

   J is a Jacobian matrix expressing the change in the location of each of the projected control points for each of the extrinsic parameters,

   λ is a variable damping parameter.

4. New estimates of the extrinsic parameters are calculated from δ.

[2]The actual pixel coordinates are the coordinates of the control points detected by the corner detector (as explained at the beginning of Section 3.2).

5. Steps (1)-(4) are repeated until δ is less than a certain threshold (implying convergence to an acceptable level of tolerance) or a predefined number of iterations is exceeded.

3.2.1.3 Main optimisation

The main calibration routine refines all the intrinsic and extrinsic parameters by minimising the following function:

$$\sum_{i=1}^{n}\sum_{j=1}^{b} \|m_{ij} - \hat{m}(A, k, R_i, T_i, M_j)\|^2 \tag{3.2.5}$$

where:
n is the number of calibration images,
$\hat{m}(A, R_i, T_i, M_j)$ is the projection of 3D point M_j in image i according to Equation (3.2.1).
This function is minimised through gradient descent as described above. The difference is that the jacobian matrix is computed for all of the calibration parameters over all of the calibration images.

3.2.2 Relative Extrinsic Calibration

Assuming that corresponding images are used to calibrate the left and right cameras, the relative position T_{rel} and orientation R_{rel} between the cameras can be found. As explained in Section 3.1.1, the left and right cameras are related through Equation (3.1.3). Substituting the refined values of R_i and T_i ($i = 1..n$) for the left and right cameras into this equation, the relative position and orientation is calculated for each pair of images. Then, the median of these values is chosen as the initial estimate of R_{rel} and T_{rel}. Refined values of R_{rel} and T_{rel} are then obtained using gradient descent as follows:

1. The corner points of the calibration pattern are projected from 3D space onto the left image plane using Equation (3.2.1) and the projection error is found.

2. Using the estimated homography for the left camera and the estimated relative position and orientation between the cameras, the homography for the right camera is estimated.

3. The corner points of the calibration pattern are projected from 3D space onto the right image plane and the projection error is found.

4. Steps (1)-(3) are repeated for each calibration image.

5. The jacobian matrix is constructed.

6. The change in the calibration parameters is computed using Equation (3.2.4).

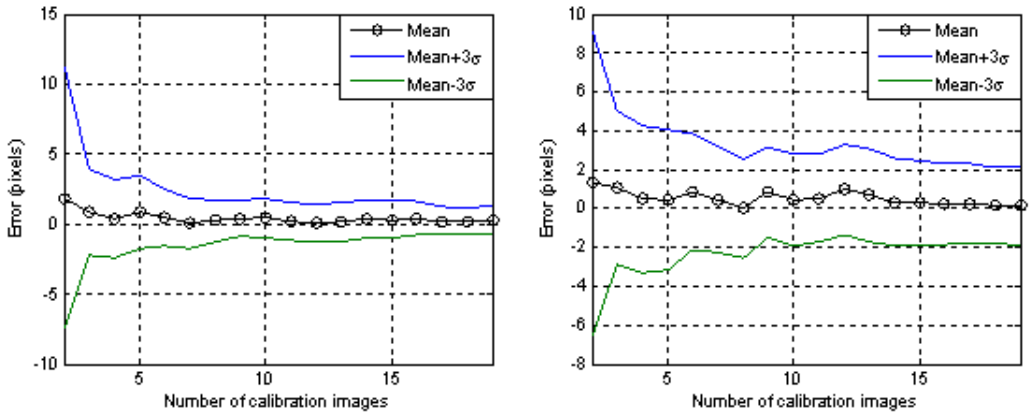
7. Steps (1)-(6) are repeated until the change in the parameters is less than a certain threshold or a predefined number of iterations is exceeded.

The process described above not only refines R_{rel} and T_{rel} but also re-estimates the intrinsic and extrinsic parameters of each camera.

3.2.3 Calibration Results

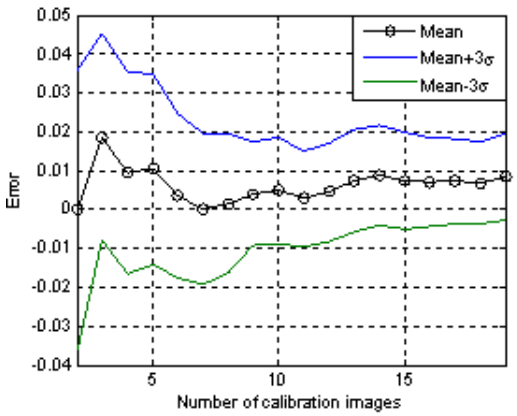
The number of calibration images is likely to affect the outcome of calibration. In order to test this assumption, the intrinsic and relative extrinsic parameters were determined by varying the number of calibration images from 2 to 19. Figure 3.7 shows how the calibration error varies for different parameters when using different numbers of images.[3] The skew α and the rotation R_{rel} are not shown because the error of these parameters is insignificant. In general it can be observed that the calibration error decreases with the number of images. In this setup, the effect is largest on the focal length, the principal point and T_x (component of T_{rel}). Beyond approximately 10 calibration images, the error remains relatively constant. For this reason, it is evident that 10 images are sufficient to calibrate the cameras. Tables 3.1 and 3.2 show the intrinsic and relative extrinsic calibration parameters obtained when using 10 images.

[3]The results shown in Figure 3.7 are also presented in tabular form in Appendix B.2.

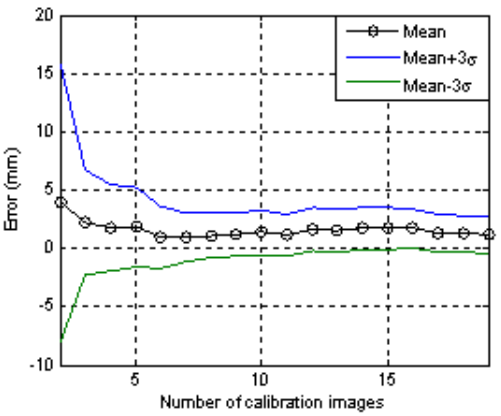


(a) Focal length

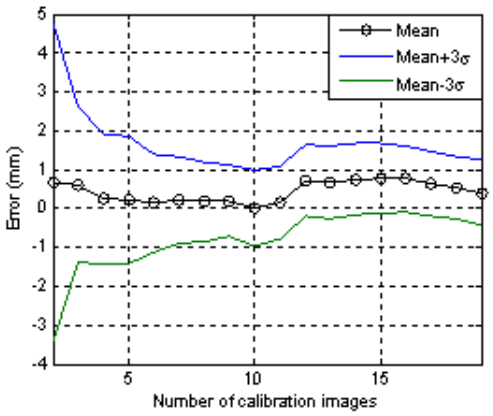
(b) Principal point



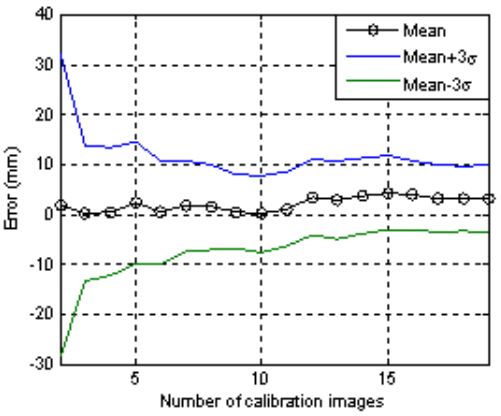
(c) Lens distortion coefficient (k_1)



(d) Translation (T_x)



(e) Translation (T_y)



(f) Translation (T_z)

Figure 3.7: Variation of error of calibration parameters with number of calibration images

Table 3.1: Intrinsic calibration results (These results were obtained for the simulated camera setup described in Section 2.2.3)

Calibration parameters		Left camera		Right camera	
		Actual	Estimated ($\mu \pm 3\sigma$)	Actual	Estimated ($\mu \pm 3\sigma$)
Focal length (pixels)	f_x	554.26	553.64 ± 1.71	554.26	553.84 ± 1.41
	f_y	554.26	554.02 ± 1.1	554.26	553.75 ± 0.96
Principal point (pixels)	c_x	320	319.51 ± 2.44	320	318.32 ± 2.37
	c_y	240	239.64 ± 2.31	240	239.38 ± 1.79
Skew	α	0	0	0	0
Lens distortion coefficients	k_1	-0.25	-0.25 ± 0.01	-0.20	-0.20 ± 0.01
	k_2	0.12	0.09 ± 0.02	0.12	0.04 ± 0.02
	k_3	0	0	0	0
	k_4	0	0	0	0
	k_5	0	0	0	0

Table 3.2: Relative extrinsic calibration results (These results were obtained for the simulated camera setup described in Section 2.2.3)

Calibration parameters		Stereo	
		Actual	Estimated ($\mu \pm 3\sigma$)
Rotation (°)	θ	0	0 ± 0.30
	ϕ	0	0 ± 0.32
	ψ	0	0 ± 0.05
Translation (mm)	T_x	-1500	-1498.64 ± 1.95
	T_y	0	0.02 ± 0.98
	T_z	0	-0.02 ± 7.72

3.3 Absolute Extrinsic Calibration

Absolute extrinsic calibration determines the position and orientation of each camera with respect to the WRF. As for intrinsic and relative extrinsic calibration, the absolute extrinsic parameters are estimated by minimising the projection error of a number of control points in the WRF with known 3D coordinates against their detected position in the IRF. However, in this case, the control points are spread over an area that is similar to the desirable region of obstacle detection. This is done so that the calibration errors are minimised over the whole region of interest of the application.

In this work, the method described in [73] is used.[4] A number of 'X'-shaped targets are spread uniformly over a rectangular region measuring 20m by 50m (the size of a typical protection zone) as shown in Figure 3.8. The centre of each target forms a control point and each camera is calibrated separately.

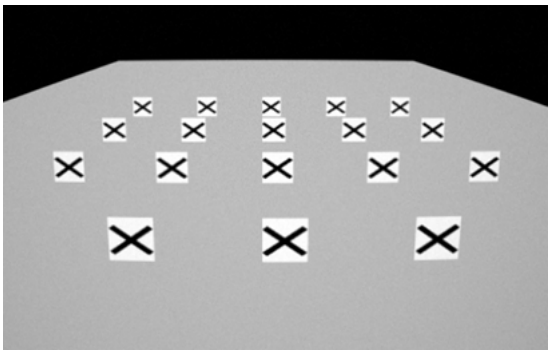


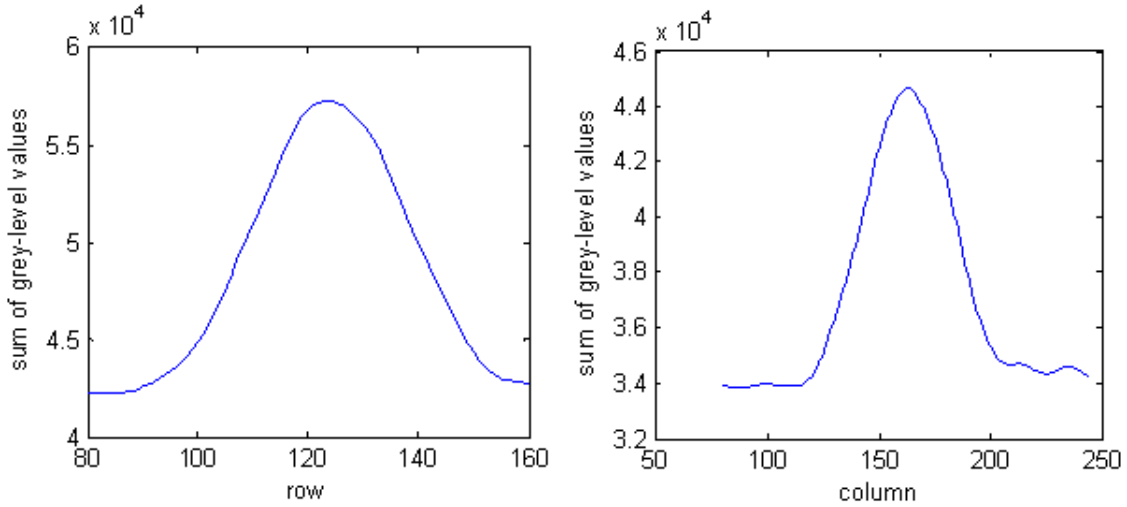
Figure 3.8: Extrinsic calibration scene setup

3.3.1 The Calibration Routine

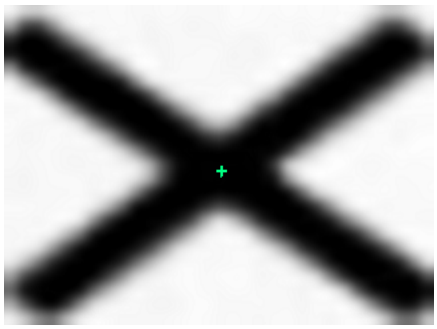
The input to the calibration routine consists of the actual coordinates of the control points in the WRF and the pixel coordinates of their projection in the IRF. The pixel coordinates of the control points are extracted from the stereo images using a software routine. First, the user selects the upper left and bottom right corners of each target to define target subimages. Each of these subimages is then scaled by bicubic interpolation. Then, for each target, two grey-level histograms are obtained: one along the rows (Figure 3.9(a)) and one along the columns (Figure 3.9(b)). The coordinates of the centre of each target are given by the midpoint of the interval with intensities within 10% of the peak of each histogram.

Before the control points are selected, the left and right calibration images are rectified.[5] One of the effects of this process is that the lens distortion is removed and the skew α is set to 0. Therefore, according to Equation (3.1.12), a point in the WRF

[4]Section 3.3.1 presents a mathematical description of this method.
[5]Rectification is discussed in detail in Chapter 4.



(a) Histogram of grey-level values along the rows (b) Histogram of grey-level values along the columns



(c) Resized target and detected centre

Figure 3.9: Detection of target centre

is projected onto the IRF as follows:

$$x_p = f_x \frac{X_c}{Z_c} + c_x = f_x \frac{r_{11}X_w + r_{12}Y_w + r_{13}Z_w + T_x}{r_{31}X_w + r_{32}Y_w + r_{33}Z_w + T_z} + c_x$$

$$y_p = f_y \frac{Y_c}{Z_c} + c_y = f_y \frac{r_{21}X_w + r_{22}Y_w + r_{23}Z_w + T_y}{r_{31}X_w + r_{32}Y_w + r_{33}Z_w + T_z} + c_y \tag{3.3.1}$$

As can be observed, there are 12 unknowns consisting of a rotation matrix R and a translation vector T. Since R is orthogonal, it has three degrees of freedom. Hence,

six constraint equations are obtained on R:

$$\begin{aligned} r_{11}r_{11} + r_{12}r_{12} + r_{13}r_{13} - 1 = 0 \\ r_{21}r_{21} + r_{22}r_{22} + r_{23}r_{23} - 1 = 0 \\ r_{31}r_{31} + r_{32}r_{32} + r_{33}r_{33} - 1 = 0 \\ r_{11}r_{21} + r_{12}r_{22} + r_{13}r_{23} - 1 = 0 \\ r_{11}r_{31} + r_{12}r_{32} + r_{13}r_{33} - 1 = 0 \\ r_{21}r_{31} + r_{22}r_{32} + r_{23}r_{33} - 1 = 0 \end{aligned} \tag{3.3.2}$$

Each control point provides two equations of the form of (3.3.1). Therefore, with a number of targets $n >= 3$, a nonlinear over-determined system of $2n + 6$ equations is formed:

$$F(u) = 0 \tag{3.3.3}$$

where $u = (T_x, T_y, T_z, r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33})$ is the vector of unknowns.

This system is solved using the Gauss-Newton iterative method. The rotation matrix is initialised as the identity matrix whereas the translation vector is initialised as a null vector. In practice, the translation vector should preferably be initialised to a value that is close to the actual value, as this would enable faster convergence.[6] At each iteration of the Gauss-Newton method, the calibration parameter vector u is updated as described below.

$$Jdu = F(u) \tag{3.3.4}$$

where J is the Jacobian matrix associated with $F(u)$.

$$\implies du = -(J(u_i)^T J(u_i))^{-1} J(u_i)^T F(u_i) \tag{3.3.5}$$

Let $A = J(u_i)^T J(u_i)$. Then,

$$du = -A^{-1} J(u_i)^T F(u_i) \tag{3.3.6}$$

[6]The reason why this was not done here was to check that the calibration routine converges successfully, irrespective of the initial conditions.

A is decomposed by QR decomposition into an orthogonal matrix Q and an upper triangular matrix D.

$$\implies du = -D^{-1}Q^T J(u_i)^T F(u_i) \tag{3.3.7}$$

The new estimate of the unknown vector is then computed:

$$u_{i+1} = du + u_i \tag{3.3.8}$$

This process is repeated by looping through Equations (3.3.5)-(3.3.8) until the norm of the residual du is below a certain threshold or a predefined number of iterations is exceeded.

3.3.2 Calibration Results

The results obtained from the absolute extrinsic calibration are presented in Table 3.3. For this experiment, the 3D coordinates of the control points were known precisely. In practice, due to measurement errors, this is not the case. This issue can affect the accuracy of calibration and is explored in greater detail in Chapter 7.

Table 3.3: Absolute extrinsic calibration results (These results were obtained for the simulated camera setup described in Section 2.2.3)

Calibration parameters		Left camera		Right camera	
		Actual	Estimated	Actual	Estimated
Rotation (°)	θ	-10	-9.908	-10	-9.884
	ϕ	0	0	0	-0.086
	ψ	0	0	0	0.006
Translation (m)	T_x	0.75	0.76	-0.75	-0.74
	T_y	7.88	7.93	7.88	7.93
	T_z	1.39	1.42	1.39	1.41

Chapter 4

Rectification and Correspondence

Rectification and correspondence are closely linked problems. As mentioned previously, 3D points in the scene are projected onto the stereo images. Rectification warps the images such that a pair of left and right image pixels, corresponding to the same 3D point, lie on the same row. Then, correspondence searches for matching pairs of image pixels. If the input images are processed directly (without rectification), the search for corresponding pixels has to be carried out over the whole image. However, by first rectifying the images, correspondence is reduced to a 1D search problem.

Section 4.1 introduces the epipolar geometry and explains how rectification modifies this geometry to simplify correspondence. Then, the rectification algorithm used in this research is discussed. Section 4.2 begins with an overview of correspondence algorithms and discusses several issues related to correspondence. Then, the selection and implementation of the correspondence algorithm used in this research are discussed in detail. Several results are presented and discussed.

4.1 Image Rectification

4.1.1 Epipolar Geometry

Rectification is best understood in terms of epipolar geometry which is modeled in Figure 4.1.[1] A point P_w in the WRF is projected onto point p_l in the left IRF and point p_r in the right IRF. O_l and O_r are the optical centres (centres of projection)

[1]The pinhole camera model still applies here.

of the left and right cameras respectively. P_w, p_l, p_r, O_l and O_r all lie on the same plane, known as the *epipolar plane* η. This plane intersects the left and right image planes at lines l and l' respectively. These are known as *epipolar lines*. Every 3D point on the ray passing through O_r and p_r is projected onto a 2D point on epipolar line l in the left image. Similarly, every 3D point on the ray passing through O_l and p_l is projected onto a 2D point on epipolar line l' in the right image. Therefore, it follows that l and l' are images of the rays passing through O_r and p_r and O_l and p_l respectively. The image in one view of the centre of projection of the other view is called the *epipole*. Since all the rays pass through a centre of projection, all the epipolar lines pass through an epipole (e_l or e_r).

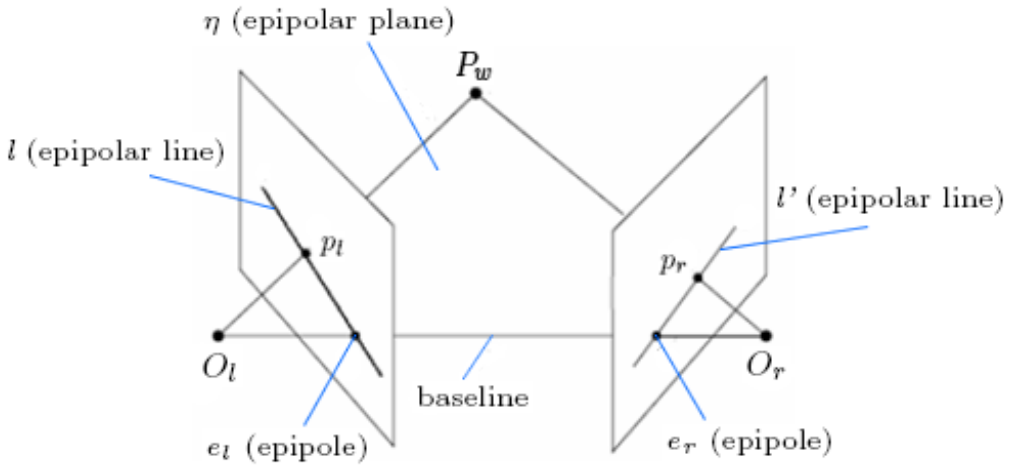


Figure 4.1: Epipolar geometry

The epipolar geometry imposes an important constraint on corresponding points. Given point p_l, the corresponding point p_r must lie on the epipolar line l'. This is known as the *epipolar constraint*. Using this constraint, it can be shown [74] that corresponding points are related as follows:

$$p_r{}^T F p_l = 0 \tag{4.1.1}$$

where:

F is known as the *fundamental matrix*,

p_l and p_r are expressed in pixel coordinates.

From the epipolar geometry it is observed that the correspondence search problem can be simplified. This is because, given a point in one image, the corresponding point in the other image has to lie on the epipolar line. Therefore, if the setup of the stereo cameras is known, it is not necessary to search over the whole image. This is a key simplification that enables higher processing speeds, a fundamental requirement for the real-time operation of the application of interest of this work.

Rectification modifies the fundamental matrix such that (a) the epipoles are shifted to infinity and the epipolar lines become parallel to each other and (b) the epipolar lines become parallel to the horizontal axis of the IRF. This means that corresponding points will have the same row coordinate.

Rectification is a necessary preprocessing step in practice because of the difficulty of physically aligning the cameras. The images obtained through rectification are the same as those obtained using parallel cameras. Therefore, rectification effectively alters the stereo geometry and eliminates alignment errors.

4.1.2 The Rectification Algorithm

Several algorithms are available to perform stereo rectification such as those identified in [75, 76]. For this research, rectification was carried out using the same toolbox used for calibration [66]. Besides rectifying the images, the rectification algorithm also removes lens distortion.

Let us assume that the stereo geometry is as shown in Figure 4.2(a). First, a rotation is applied to both cameras in order to bring them in the same orientation as shown in Figure 4.2(b). The rotations R_l and R_r applied to the left and right cameras are obtained from the relative camera orientation $R_{relOld} = R(z, \psi)R(y, \phi)R(x, \theta)$ determined during calibration:

$$\begin{aligned} R_r &= R(z, -\psi/2)R(y, -\phi/2)R(x, -\theta/2) \\ R_l &= {R_r}^T \end{aligned} \tag{4.1.2}$$

The relative translation vector $\boldsymbol{t}$ between the cameras then becomes

$$\boldsymbol{t} = R_r T_{relOld} \tag{4.1.3}$$

where T_{relOld} is the relative translation vector also determined during calibration.

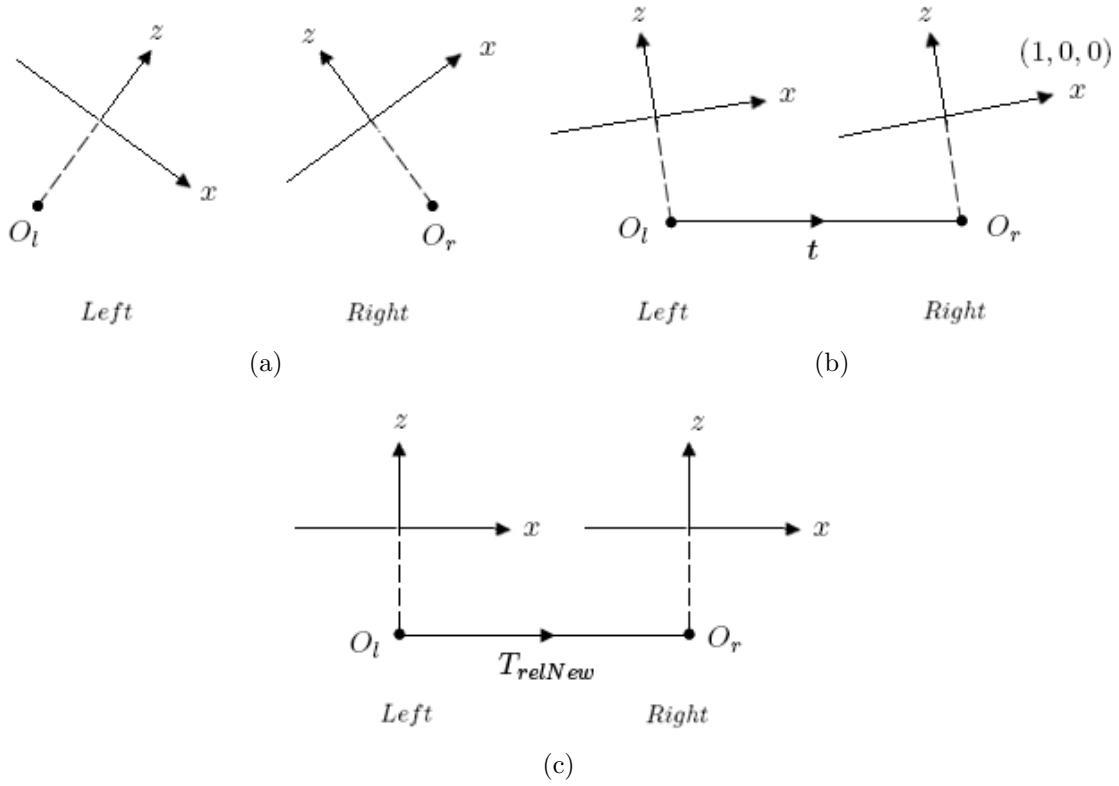


Figure 4.2: Modifying the stereo geometry: (a) plan view of original stereo setup, (b) plan view of the cameras in the same orientation, (c) plan view of the cameras with the epipolar lines parallel to the horizontal axis of the IRF

Next, the cameras are rotated such that the relative translation vector becomes parallel to the x axis of the IRF (axis $(1, 0, 0)$) as shown in Figure 4.2(c). This effectively shifts the epipoles to infinity and makes the epipolar lines parallel to the horizontal axis of the IRF. The axis of rotation (specified by unit vector $\hat{\boldsymbol{w}}$), angle of rotation φ and rotation vector $\boldsymbol{r}$ are given by

$$\begin{aligned} \hat{\boldsymbol{w}} &= \frac{\boldsymbol{t} \otimes \boldsymbol{u}}{\|\boldsymbol{t} \otimes \boldsymbol{u}\|} \\ \varphi &= cos^{-1}\left(\frac{|\, \boldsymbol{t}.\boldsymbol{u} \,|}{|\, \boldsymbol{t} \,||\, \boldsymbol{u} \,|}\right) \\ \boldsymbol{r} &= \hat{\boldsymbol{w}}\varphi \end{aligned} \tag{4.1.4}$$

where $\boldsymbol{u} = (1, 0, 0)$.

A rotation matrix $R2$ is obtained from rotation vector $\boldsymbol{r}$ using Rodrigues' rotation formula [77]. Therefore, the global rotations R_{left} and R_{right} that have to be applied to the cameras are:

$$\begin{aligned} R_{left} &= R2R_l \\ R_{right} &= R2R_r \end{aligned} \tag{4.1.5}$$

After applying these global rotations, the cameras become perfectly aligned. Therefore, the new relative orientation R_{relNew} is the identity matrix. The new relative translation vector T_{relNew} is of the form of $(a, 0, 0)$ where a is the horizontal displacement (baseline distance) between the cameras. T_{relNew} is given by

$$T_{relNew} = R_{right}T_{relOld} \tag{4.1.6}$$

After estimating the global rotation that has to be applied to each camera, 'new' intrinsic parameters are computed for each camera as follows:

- Focal length - This is set to a value that retains as much information (contained in the original distorted images) as possible. The focal lengths of both cameras are set to equal values.[2]
- Skew - This is set to 0 in order to prevent skew in the rectified images.
- Distortion coefficients - These are set to 0 to prevent lens distortion in the rectified images.
- Principal point - This is set to a value that maximises the visible area in the rectified images. The principal point coordinates of both cameras are set to equal values.

The next step is to warp the original distorted images in order to obtain the rectified images. In order to do this it is necessary to establish the relationship

[2]Refer to Appendix C.1 for a description of the methods used to set the focal length and principal point of the rectified stereo cameras.

between pixel coordinates (x_{p1}, y_{p1}) in the rectified image and corresponding pixel coordinates (x_{p2}, y_{p2}) in the distorted image.

Given (x_{p1}, y_{p1}) and using Equation (3.1.10), the distorted normalised projection (x_{d1}, y_{d1}) in the rectified image is given by:

$$\begin{pmatrix} x_{d1} \\ y_{d1} \\ 1 \end{pmatrix} = {A_{new}}^{-1} \begin{pmatrix} x_{p1} \\ y_{p1} \\ 1 \end{pmatrix} \tag{4.1.7}$$

where A_{new} is the 'new' intrinsic camera matrix. Since there is no distortion in the rectified image, the normalised projection (x_{n1}, y_{n1}) is identical to (x_{d1}, y_{d1}). Therefore,

$$\begin{pmatrix} x_{n1} \\ y_{n1} \\ 1 \end{pmatrix} = {A_{new}}^{-1} \begin{pmatrix} x_{p1} \\ y_{p1} \\ 1 \end{pmatrix} \tag{4.1.8}$$

The normalised projection (x_{n1}, y_{n1}) is rotated by the global rotation determined earlier, as follows:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = {R_{global}}^{T} \begin{pmatrix} x_{n1} \\ y_{n1} \\ 1 \end{pmatrix} \tag{4.1.9}$$

where R_{global} is equal to R_{left} or R_{right} for the left and right images respectively. The normalised projection (x_{n2}, y_{n2}) in the original distorted image is given by:

$$\begin{aligned} x_{n2} &= \frac{x}{z} \\ \\ y_{n2} &= \frac{y}{z} \end{aligned} \tag{4.1.10}$$

The normalised distorted projection (x_{d2}, y_{d2}) in the original distorted image is obtained by substituting (x_{n2}, y_{n2}) and the 'old' distortion coefficients $k_1..k_5$ in Equation (3.1.8). Finally, the distorted pixel coordinates (x_{p2}, y_{p2}) are obtained using Equation (3.1.10):

$$\begin{pmatrix} x_{p2} \\ y_{p2} \\ 1 \end{pmatrix} = A_{old} \begin{pmatrix} x_{d2} \\ y_{d2} \\ 1 \end{pmatrix} \tag{4.1.11}$$

where A_{old} is the 'old' intrinsic camera matrix. The coordinates (x_{p2}, y_{p2}) are unlikely to be integers. Therefore, the intensity of the distorted image at this pixel position is calculated as a function of the intensity values of the four closest pixels (Figure 4.3).

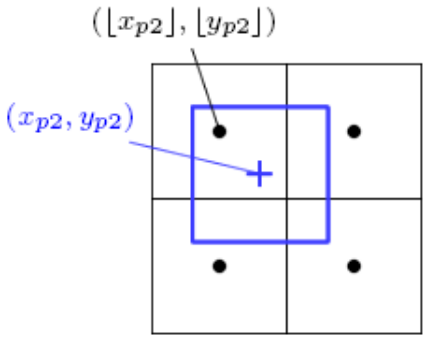


Figure 4.3: The 4-pixel neighborhood of a (fictitious) pixel with non-integer coordinates (x_{p2}, y_{p2})

The algorithm generates four blending coefficients $a_1..a_4$ and four array indices $ind_1..ind_4$ corresponding to the nearest pixels:

$$\begin{aligned} a_1 &= (1-\alpha_y)(1-\alpha_x) \\ a_2 &= (1-\alpha_y)\alpha_x \\ a_3 &= \alpha_y(1-\alpha_x) \\ a_4 &= \alpha_y\alpha_x \\ ind_1 &= x_{p3}nr + y_{p3} + 1 \\ ind_2 &= (x_{p3}+1)nr + y_{p3} + 1 \\ ind_3 &= x_{p3}nr + y_{p3} + 2 \\ ind_4 &= (x_{p3}+1)nr + y_{p3} + 2 \end{aligned} \tag{4.1.12}$$

where:

$(x_{p3}, y_{p3}) = (\lfloor x_{p2} \rfloor, \lfloor y_{p2} \rfloor)$,

$\alpha_x = x_{p2} - x_{p3}$,

$\alpha_y = y_{p2} - y_{p3}$,

nr is the number of image rows.

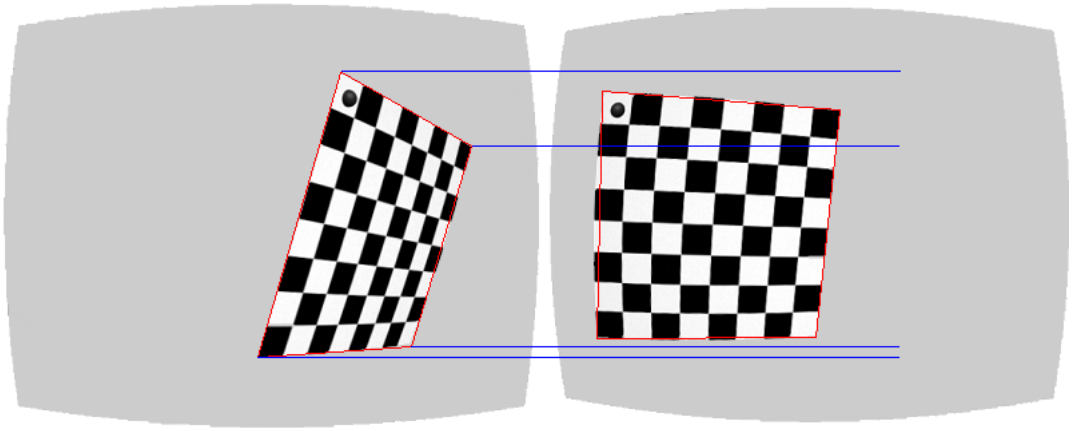
The intensity I of the distorted image at position (x_{p2}, y_{p2}) is then found using Equation (4.1.13):

$$I = a_1 Im(ind_1) + a_2 Im(ind_2) + a_3 Im(ind_3) + a_4 Im(ind_4) \tag{4.1.13}$$

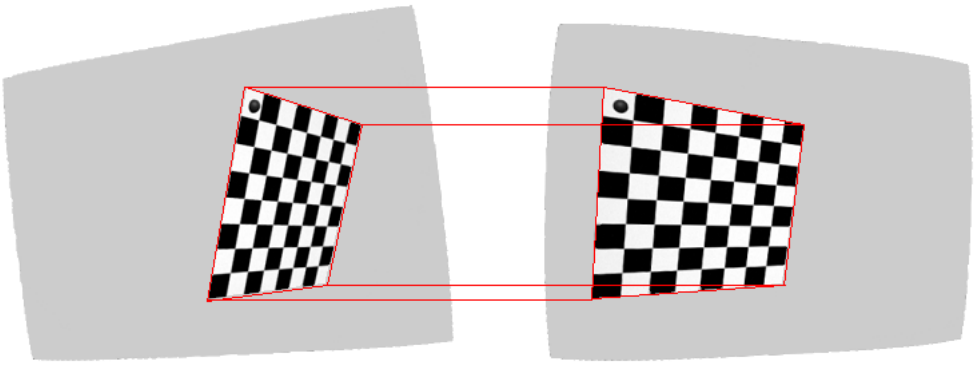
where Im is a 2D array representing the distorted image. The intensity value I is finally assigned to the pixel with coordinates (x_{p1}, y_{p1}) in the rectified image.

Blending coefficients and array indices are generated for every pixel position in the rectified images. Since this process relies only on knowledge of the calibration parameters, it is only carried out offline. The 'new' calibration parameters as well as the array indices and blending coefficients are then stored for direct use online according to Equation (4.1.13).

Figure 4.4 shows a pair of stereo images before and after rectification. For the example shown in this figure, the stereo cameras were deliberately misaligned in orientation and significant radial lens distortion was applied to both cameras. From Figure 4.4(a) it is clear that corresponding points in the left and right images have different row coordinates. Also, lens distortion is emphasised by the straight red lines superimposed on the calibration object. In Figure 4.4(b) it is observed that, as expected, the rectification algorithm successfully removes lens distortion and ensures that corresponding points lie on the same row.



(a) Distorted stereo images



(b) Rectified stereo images

Figure 4.4: Rectification: In the original distorted images, corresponding points in the left and right images (such as the corners of the calibration object) have very different row coordinates. Due to radial lens distortion, the square calibration object appears to be curved. This is quite clear from the right distorted image and is emphasised by the straight red lines superimposed on the calibration object. In the rectified images, corresponding points lie on the same row and lens distortion is removed. As a result, the lines of the calibration pattern appear to be straight.

4.2 Correspondence

4.2.1 Background

4.2.1.1 Correspondence issues

Before looking at the different correspondence methods available in the literature, it is relevant to discuss a number of issues associated with stereo images and which can complicate the search for corresponding points:

- **Occlusions:** These are areas in the scene which are either visible in one of the images (in which case they are referred to as partial occlusions (Figure 4.5(a))) or in neither of them (in which case they are referred to as full occlusions (Figure 4.5(b))). Most occlusions are caused by depth discontinuities that occur at the boundary between objects and the background. Occlusion zones are more likely to increase with distance between the cameras. It is not possible to find corresponding pixels in occluded areas; however, it is possible to detect occlusions and compensate for them, as explained later on in this section and in Section 4.2.3. This can be useful to detect incorrect correspondences or match ambiguities.

Figure 4.5: Examples of occlusion: (a) partial occlusion (the cube is partly hidden in the left image) and (b) full occlusion (the cone is partly hidden in both images)

- **Photometric distortion:** Corresponding pixels may have different intensities in the left and right images. This is more evident in outdoor applications and is mainly caused by differences in light reflections and, consequently, in the intensity of light entering each camera, as well as differences in camera settings

(such as camera gain and bias). Different ways of compensating for this type of distortion exist and one of the methods is described in Section 4.2.2.

- **Projective distortion:** Due to the fact that the scene is captured from different viewpoints, the same scene object appears differently in the left and right images. The difference in appearance increases with distance between the cameras. As discussed in Chapter 5, projective distortion is an important issue that needs to be considered when choosing the baseline distance of the stereo vision system.

- **Image noise:** Image noise depends on the quality of the camera sensor and scene lighting and it increases the difference between corresponding points. Filtering and other measures are available to compensate for image noise. These methods are described further in Sections 4.2.2 and 4.2.3.

4.2.1.2 Constraints and assumptions

In order to detect false matches, outliers and ambiguities caused by the problems discussed in Section 4.2.1.1, most correspondence algorithms implement a number of constraints and make certain assumptions. Apart from the epipolar constraint mentioned in the previous section, the most common constraints and assumptions are:

- **Smoothness:** This is the assumption that disparity varies smoothly everywhere in the scene except at object boundaries. This means that the disparity of a pixel is generally closely related to that of its neighbours. This assumption can be used to prevent sudden changes in disparity due to image noise and other factors.

- **Uniqueness:** A point in the scene is projected onto a single pixel in the left and right images. This means that each pixel should have a unique corresponding match. If, during correspondence, a pixel in one image matches with more

than one pixel in the other image, the uniqueness constraint is violated. This constraint is described in more detail in Section 4.2.4.

- **Ordering (Monotonicity):** This constraint requires that the ordering of features along a row (scanline) is preserved between the left and right stereo images. If a and a' and b and b' are two pairs of corresponding pixels and a is to the left of b, then a' should also be to the left of b' and vice versa. Figure 4.6(a) illustrates the ordering constraint for two 3D points, A and B. This constraint fails when a 3D point falls within the *forbidden zone* of one or more 3D points. For example, in Figure 4.6(b), point C lies within the forbidden zone of point A. Hence, the order of the pixels corresponding to points A and C is reversed between the left and right images. The ordering constraint tends to fail mostly when the scene contains narrow foreground objects because points on these objects are more likely to fall within forbidden zones.

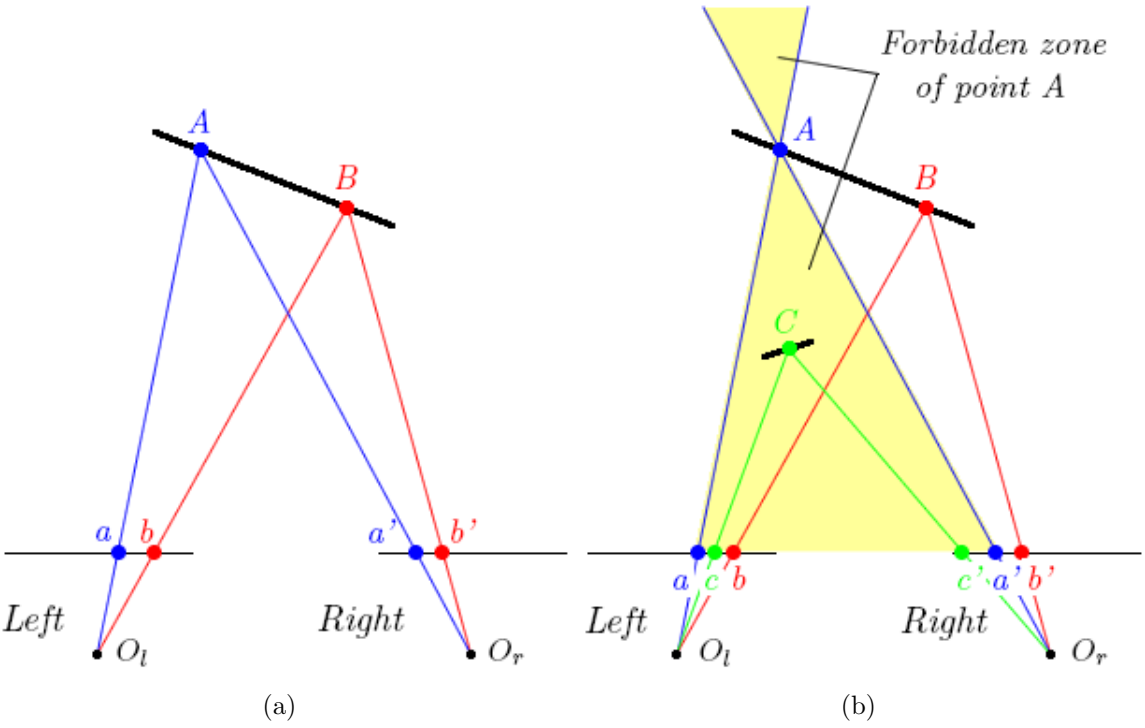


Figure 4.6: The ordering constraint (a) and violation of the constraint (b)

- **Left-right consistency:** During correspondence, one of the images is used as a reference and the disparity of pixels in this image is calculated by finding corresponding pixels in the other image. The left-right consistency constraint states that a pixel a in the left image must correspond to a pixel b in the right image irrespective of whether it is the left image or the right image that is used as a reference. Left-right consistency provides one way of detecting occlusions. However, two disparity maps need to be computed to enable this, one for each reference image.

- **Inter-scanline consistency:** Certain features, such as edges, are preserved between the left and right images. This means that if a group of pixels form an edge in the left image, their corresponding pixels in the right image will also form an edge. This implies that the disparities of the pixels making up these features are related to each other. Since a feature can span several scanlines, this concept is known as *inter-scanline consistency*. Therefore, the reliability of the disparity of such pixels can be determined by checking whether it is consistent with the disparity of the rest of the pixels making up the feature.

4.2.1.3 Correspondence methods

Correspondence methods can be broadly classified into two categories: those that produce a dense disparity map and those that produce a sparse disparity map. Correspondence methods that produce a dense disparity map use intensity information and correlation techniques to compute the disparity of every pixel in the image. These methods are further divided into local and global optimisation methods.

As the name implies, local methods use local information in order to find corresponding points. These methods are also commonly known as *window methods*. The disparity of a pixel of interest in one image is found by comparing a small region (window) around the pixel with similarly sized regions around candidate pixels in the other image. The window of the second image that best correlates with the window (around the pixel of interest) of the first image is used to determine the candidate

pixel that best corresponds with the pixel of interest.

Global correspondence methods differ from local methods in that they use information from a larger region of the image in order to find the disparity at each pixel. Also, instead of computing the disparity of each pixel directly and separately, these methods compute the disparity by minimising a global cost function. This cost function consists of a matching cost and a smoothness cost. There are several global methods including dynamic programming, graph-cuts, simulated annealing, and probabilistic diffusion. Dynamic programming is one of the more common global methods used. It computes the minimum cost path between a pair of corresponding scanlines. It can detect partial occlusions and assigns a fixed cost to them. However, dynamic programming enforces the ordering constraint which, as explained earlier in this section, can be violated. This results in local errors tending to propagate along a scanline, corrupting other potentially good matches and resulting in horizontal streaks in the disparity map.

A comprehensive review and evaluation of dense correspondence methods can be found in [78].

Correspondence methods that produce a sparse disparity map are also known as *feature-based correspondence methods*. This is because they only compute the disparity of particular low-level image features such as edges and corners, or higher-level features such as lines, curves and circles. Naturally, these methods require a feature detector. Once the features are detected in the stereo images, corresponding features can be found either by using intensity information around each feature or by using feature descriptors. For example, an edge feature can be described in terms of its strength (magnitude) and direction. This information is stored in a feature vector. Then, the distance between the feature vector of a feature of interest and the feature vector of candidate features is measured. The best match is given by the candidate feature which results in the shortest distance between the feature vectors.

Feature-based correspondence methods are suitable for applications that do not require a complete 3D reconstruction of the scene, such as navigation or obstacle

detection and avoidance. Since only certain features are processed, the computation time is significantly reduced. The same correspondence methods that are used to obtain a dense disparity map can also be used to process only certain features. For example, a local (window-based) correspondence method is used to process vertical edges in [59, 79]. In [80, 81], edges are matched using dynamic programming, with the matching cost in [81] being based on edge gradient vectors. The correspondence methods cited here are all used in outdoor obstacle detection applications.

Correspondence is the most time-consuming task in the stereo vision process. Therefore, in most cases, it is necessary to find a compromise between complexity, computation time and accuracy or to sacrifice one property in favour of another. Some systems achieve real-time performance by using customised hardware, Digital Signal Processors (DSPs) or Field-Programmable Gate Arrays (FPGAs). However, it is also possible to achieve this kind of performance on a general purpose processor by parallelising the computationally intensive section of the code. For example, in the case of window-based methods, the same instruction is essentially repeated over different pixels in the image. Each pixel can be processed independently of the rest; hence, it is possible to process several pixels simultaneously. This can be done using Single Instruction Multiple Data (SIMD) techniques. Different instruction sets for general purpose processors have been developed to support SIMD programming. These include MultiMedia eXtensions (MMX) and Streaming SIMD Extensions (SSE). In [82], the authors present an efficient implementation of window-based correspondence which achieves real-time performance not only by using SIMD techniques but also by taking advantage of the data redundancy which is inherent in the algorithm.

4.2.2 The Correspondence Algorithm

In this work the aim is not to reconstruct the whole scene but to detect obstacles within the scene. For this reason it is only necessary to find the disparity of subsets of the image that are likely to correspond to obstacles. Therefore, a feature-based

correspondence method using intensity information has been adopted.

The image features that are processed are edge pixels. There are several reasons for processing edges. Firstly, edges contain the most important structural information about an image. They normally occur in textured image regions and are more likely to provide a reliable match. Most importantly, edges are a key feature of obstacles in the context of this application. Moreover, since edge pixels account for a small percentage of the whole image, computation time can be significantly reduced by using this approach. An analysis of several images captured with real cameras in ramp areas and taxiways showed that, on average, edge pixels account for less than 12% of the whole image.[3] The main reason for this is that a large proportion of each image consists of the ground surface which normally has low texture.

Several edge detection methods are available. Figure 4.7 shows the results obtained when applying three different edge detectors to a noisy synthetic image of a typical aerodrome scene. The poorest performance is obtained from the Roberts edge detector, which only suppresses noise at the expense of removing true but weak edges such as the ground markings. This results in the fragmentation of edge contours (Figure 4.7(b)). The Prewitt detector performs better than the Roberts detector, although some edge fragmentation is still produced (Figure 4.7(c)). The Canny detector, on the other hand, removes noisy edges while preserving continuous edge contours (Figure 4.7(d)). Due to these superior properties, the Canny edge detection method [74] was selected for this work.

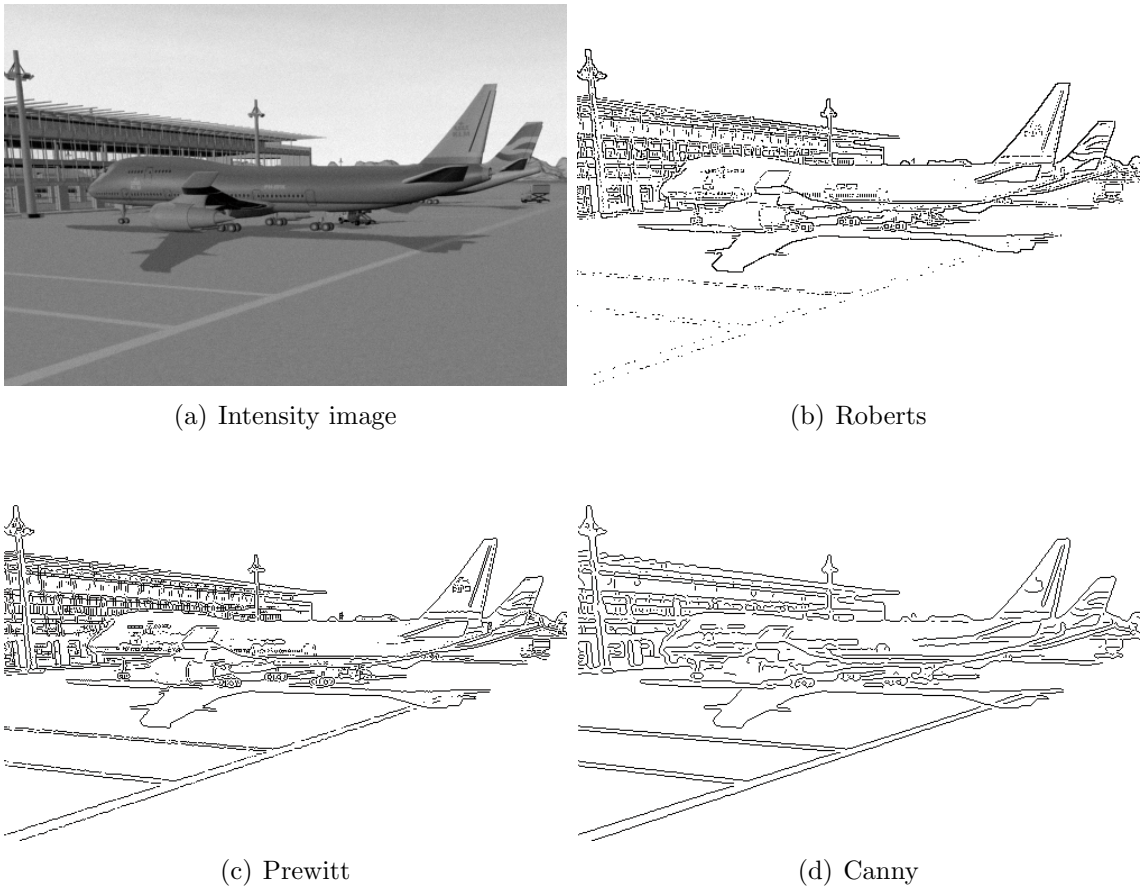
Canny edge detection is carried out in four steps as follows:

1. **Image filtering**: The intensity image is smoothened with a Gaussian filter. This suppresses noise and blurs the image slightly.

2. **Generation of edge map**: The strength (magnitude) and orientation (angle) of the edge normal is determined at each pixel.

3. **Non-maximum suppression**: The edge map produced in Step (2) may have

[3]Refer to Appendix C.2 for the results of this analysis.



(a) Intensity image (b) Roberts

(c) Prewitt (d) Canny

Figure 4.7: Edge detection results: noisy intensity image (a) and edge maps obtained using the Roberts (b), Prewitt (c) and Canny (d) edge detection techniques

edges that are several pixels thick. In order to reduce edge thickness to a single pixel, a search is carried out in the direction of the edge normal for each edge pixel. If the strength of the edge at the pixel is greater than at least one of its two neighbours, the edge pixel is retained; otherwise, it is discarded. This is known as *non-maximum suppression*.

4. **Hysteresis thresholding and edge linking**: Other edge detectors (such as the Roberts and Prewitt detectors) use a single threshold to separate edge pixels from non-edge pixels. If the threshold is low, true weak edges as well as noisy edges are retained, potentially resulting in false edge contours. If the threshold is high, noise is removed but edge contours become fragmented. This is because

true edge maxima tend to fluctuate above and below the threshold due to noise (This explains the results obtained in Figures 4.7(b) and 4.7(c)). The Canny detector solves these problems using hysteresis thresholding. Two thresholds t_1 and t_2, where $t_1 < t_2$, are applied. Strong edges, whose magnitude exceeds t_2, are automatically classified as edge pixels whereas edges whose strength is below t_1 are automatically removed. Weak edges, whose strength lies between the two thresholds, are only marked as edge pixels if there is a continuous chain of edges linking them to strong edges (with a magnitude greater than t_2).

The output of edge detection is a binary edge map with pixels either classified as edges (1) or non-edges (0). For each edge pixel, the disparity is found using a local, window-based method. This matches edge pixels in the left (reference) image to corresponding edge pixels in the right image as follows (refer to Figure 4.8):

1. A square window of pixels is selected around a pixel $p_l(x_1, y_1)$ in the left image.[4]
2. A similar window is selected around a candidate edge pixel, with the same row coordinate, in the right image.
3. The matching cost between the left and right image windows is computed.
4. Steps (2) and (3) are repeated for every edge pixel that is within the disparity search region.[5]
5. The right edge pixel $p_r(x_2, y_1)$ corresponding to the minimum matching cost (global minimum) is identified.
6. The disparity d of $p_l(x_1, y_1)$ is given by:

$$d = x_1 - x_2 \tag{4.2.1}$$

[4]The disparity is assumed to be constant over the window. Hence, local methods make an implicit smoothness assumption.
[5]The disparity search region lies within a single pixel row due to rectification.

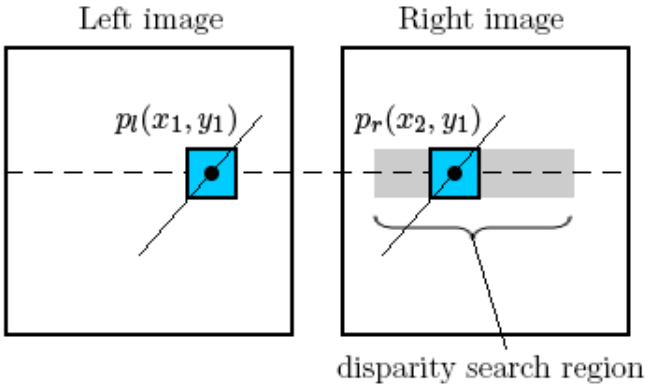


Figure 4.8: Window-based correspondence

There are several matching costs that can be used to compare intensity regions (windows) in the left and right images. The most common are the Sum of Squared Differences (SSD), the Sum of Absolute Differences (SAD) and Cross-Correlation (CC) methods. The SAD matching cost is used here because it is more computationally efficient than SSD and CC and still performs well in this application.

Since the stereo vision system is to be used in an outdoor environment, it is important to compensate for photometric distortion.[6] This is done by normalising the intensity windows. The average window intensity is subtracted from the pixel intensities in each window and the result is divided by the standard deviation. The matching cost between left and right image regions is therefore given by:

$$C(m) = \sum_{(x,y)\in W} \left| \left(\frac{I_l(x,y) - \mu_l}{\sigma_l} \right) - \left(\frac{I_r(x+m,y) - \mu_r}{\sigma_r} \right) \right| \tag{4.2.2}$$

where:

$d_{min} \leq m \leq d_{max}$ is the disparity search range,

$C(m)$ is the matching cost at disparity m,

W represents the left and right image regions,

$I_l(x,y)$ and $I_r(x+m,y)$ are the intensities of pixels within the left and right windows respectively,

μ_l and μ_r are the average intensities of the left and right image regions respectively,

[6] Photometric distortion is defined in Section 4.2.1.

σ_l and σ_r are the standard deviations of the intensities of the left and right image regions respectively.

4.2.3 Selection of Window Size and Number of Windows

One important issue that can affect the quality of the disparity map is the size of the window used during the correspondence process. A small window is prone to noise but is computationally quick and all the pixels within the window are likely to be at the same depth (i.e. the disparity will be constant within the window). On the other hand, a larger window has a better Signal-to-Noise Ratio (SNR) but increases the processing time. Also, as the window size is increased, it is more likely that the disparity changes within the window. This means that a larger window has a higher probability of containing occluded pixels which lead to increased differences between the left and right image regions used for matching. Therefore, an optimal window size has to be determined through compromise.

To choose a suitable window size, the correspondence algorithm was tested on 8 stereo images, the ground truth disparity maps of which were known. These are standard test stereo images and are available in [83]. The images were corrupted with AWGN with 0 mean and a standard deviation of 3 intensity levels. The details of the test images are given in Table 4.1 and the images are presented in Appendix C.3. Although these images are not directly related to the application of interest of this work, they are still valid for the purpose of this particular experiment because they share several of the typical characteristics of images acquired in ramps and taxiways, such as: occlusions, depth discontinuities, and generic shape information.

Only the edge pixels were processed in each image. Figure 4.9 shows how the processing time[7] and the percentage of correct disparities[8] change when increasing the window size from 3x3 pixels to 15x15 pixels. It can be observed that the greatest

[7]The percentage increase in processing time is measured with respect to the processing time when using a window size of 3x3 pixels.

[8]A disparity value is considered to be correct if the difference between the computed and correct disparities is less than or equal to 1 pixel.

Table 4.1: Characteristics of test images (Refer to Appendix C.3)

Image	Size (pixels)	Edges (%)	Maximum disparity (pixels)
Aloe	555 x 641	17	135
Art	555 x 695	8.5	100
Cones	375 x 450	12.8	55
Baby1	555 x 620	11	150
Books	555 x 695	9.3	100
Dolls	555 x 695	9.9	100
Midd1	555 x 698	5	98
Teddy	375 x 450	9	53

increase in the percentage of correct disparities occurs when the window is enlarged from 3x3 pixels to 5x5 pixels. Little improvement is observed for windows larger than 7x7 pixels. The percentage of correct disparities then either remains constant as the window size is increased further, or even decreases slightly. These observations can be confirmed visually by looking at the disparity map obtained for one of the test images for different window sizes (Figure 4.10).[9] In Figure 4.10(b), the disparity map is very 'noisy' and disparity values vary significantly over individual objects in the scene. When the window size is increased to 5x5 pixels (Figure 4.10(c)), the disparity map improves a lot and the disparity changes more smoothly over individual objects. A smaller improvement is obtained when increasing the window size to 7x7 pixels (Figure 4.10(d)). Changes to the disparity map become less evident when using increasingly larger windows (Figures 4.10(e)-4.10(h)).

As expected, processing time increases with image size and percentage of edge pixels, window size, and disparity search range. For any particular image, the processing time increases non-linearly as the window size is increased.

Since the biggest improvement in the accuracy of the disparity map is obtained when increasing the window size from 3x3 pixels to 5x5 pixels, a window size of 5x5 pixels would provide the ideal compromise between disparity map quality and

[9] All of the image pixels were processed in this particular example in order to make it easier to visually compare the disparity maps.

computation time. However, in order to be on the safe side, a window size of 7x7 pixels was chosen for this work. When using such a window, good improvements were obtained (for most of the test images) when increasing the window size from 5x5 pixels to 7x7 pixels (Figure 4.9).

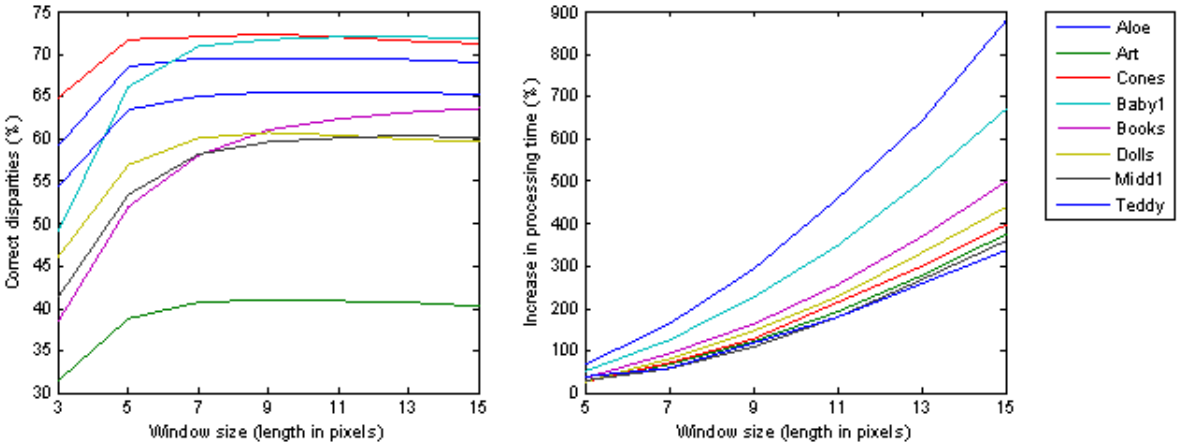


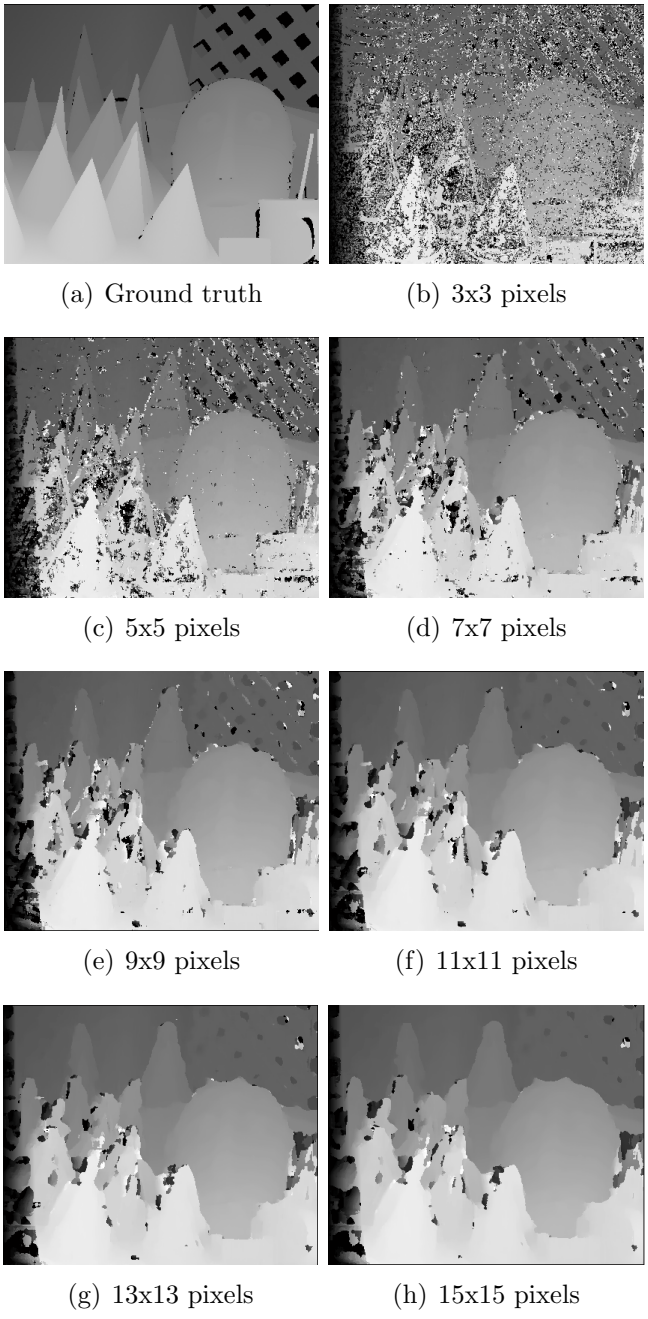
Figure 4.9: Effect of correlation window size on percentage of correct disparities (left) and percentage increase in processing time (right)

Apart from window size, another important consideration is the number of correlation windows used. If only a single window is used to match left and right image regions, a bad match is likely to occur whenever the regions contain depth discontinuities or occluded pixels, since these tend to increase the difference between the two regions. This problem can be reduced by using multiple windows, where each window is situated at a slightly different position with respect to the pixel of interest [84]. This means that even if the pixel of interest is close to an occluded region or a depth discontinuity, one or more of the windows will remain unaffected. Therefore, this increases the likelihood of finding a reliable match.

To cater for multiple windows, the correspondence algorithm described in Section 4.2.2 is modified as follows:

1. A square window of pixels is selected around a pixel $p_l(x_1, y_1)$ in the left image.

2. A similar window is selected around a candidate edge pixel, with the same row coordinate, in the right image.



(a) Ground truth (b) 3x3 pixels

(c) 5x5 pixels (d) 7x7 pixels

(e) 9x9 pixels (f) 11x11 pixels

(g) 13x13 pixels (h) 15x15 pixels

Figure 4.10: Effect of correlation window size on disparity map quality (*Cones* test image): With a window size of 3x3 pixels, the disparity map is very 'noisy' and disparity values vary significantly over individual objects in the scene (Figure 4.10(b)). With a 5x5 window, the Signal-to-Noise Ratio improves and the disparity changes more smoothly over individual objects (Figure 4.10(c)). A smaller improvement is obtained when increasing the window size to 7x7 pixels (Figure 4.10(d)), becoming even smaller with larger windows (Figures 4.10(e)-4.10(h)).

3. The matching cost between the left and right image windows is computed.

4. Steps (2) and (3) are repeated for every edge pixel that is within the disparity search region.

5. Steps (1)-(4) are repeated for each of the correlation windows.

6. The right edge pixel $p_r(x_2, y_1)$ corresponding to the minimum matching cost (over all the correlation windows) is identified.

7. The disparity d of $p_l(x_1, y_1)$ is given by Equation (4.2.1).

To demonstrate the benefit of using a multi-window scheme, the correspondence algorithm was tested on 100 noisy images generated through the simulation environment described in Section 2.2.4. Nine 7x7 correlation windows (shown in Figure 4.11) were used during the correspondence process. When all of the images were processed, the total number of times that each type of window produced the best match (minimum matching cost) was expressed as a percentage of the total number of processed edge pixels.

Another test was carried out to check the percentage increase in processing time when varying the number of correlation windows from 1 to 9.[10]

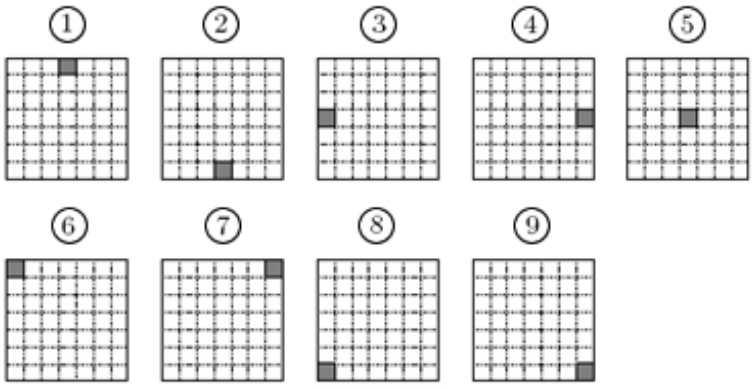
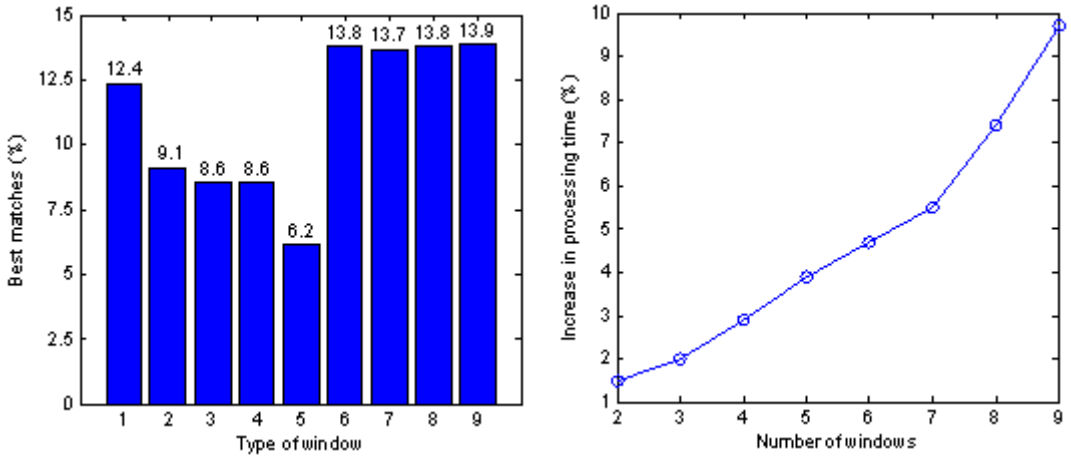


Figure 4.11: Different types of correlation windows (The grey pixel represents the pixel of interest)

[10]The increase in processing time was measured with respect to the processing time when using only 1 window.



(a) (b)

Figure 4.12: Percentage of best matches provided by each type of window (a) and percentage increase in processing time with number of windows (b)

The results are shown in Figure 4.12. In Figure 4.12(a), the numbers on the x axis represent the different types of windows shown in Figure 4.11. As expected, the window that produces the least best matches is the central window (window type 5). Since most of the edge pixels occur at object boundaries, the central window has a greater probability of containing depth discontinuities. Therefore, the disparity is not constant within the window, leading to bad matches.

The best matches are produced by window types 6-9, where the pixel of interest lies at the corner of the correlation window. These four windows account for over half of the best matches. Moreover, there is only a 2.9% increase in processing time when using four windows as opposed to a single window. Therefore, as a compromise between matching accuracy and processing time, a 4-window scheme (consisting of window types 6-9) was adopted in this work.

4.2.4 Detection of Incorrect Disparities

After estimating the disparity of a pixel, a number of tests are carried out to ensure that this disparity is reliable and accurate.

The first test exploits the uniqueness constraint mentioned in Section 4.2.1.2.

Assume that, when finding the disparity of a left image pixel p_{l1}, the best match is provided by a right image pixel p_r, with matching cost C_1. Now assume that, when determining the disparity of another left image pixel p_{l2} (with the same row coordinate as p_{l1}), the best match is again provided by the right image pixel p_r, with matching cost C_2 (Figure 4.13). This means that pixel p_r does not correspond uniquely to a single left image pixel. This violates the uniqueness constraint and, therefore, one of the matches must be incorrect. In this case, the matching costs C_1 and C_2 are compared and the better match (lower matching cost) is accepted while the other is rejected. This allows the correspondence algorithm to recover from previous matching errors.

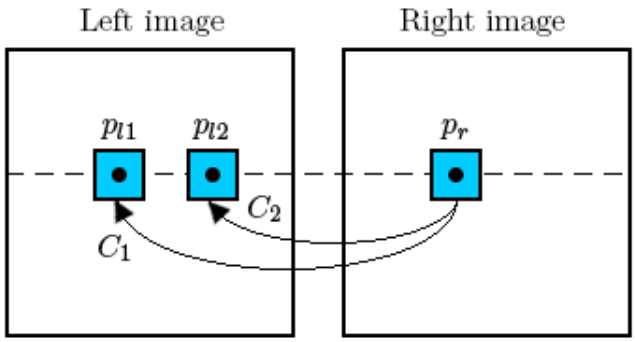


Figure 4.13: Violation of the uniqueness constraint: Every point in the scene is projected onto a single pixel in the left and right images. Therefore, each pixel should have a unique corresponding match. In this figure, the uniqueness constraint is violated because point p_r is matched with two left image pixels (p_{l1} and p_{l2}). Hence, one of the matches must be incorrect. The matching costs C_1 and C_2 are compared and the better match (lower matching cost) is accepted while the other is rejected.

If the uniqueness constraint is satisfied, a second test is carried out. This is called the *sharpness* test and it compares the disparity corresponding to the global minimum with the disparity associated with three pseudo-minima:[11]

$$\Delta d = \sum_{i=1}^{3} |d_i - d_{min}| \tag{4.2.3}$$

where:

[11]The global minimum and pseudo-minima are found by sorting out the matching costs during Step 5 of the correspondence algorithm described in Section 4.2.2.

d_{min} is the disparity corresponding to the global minimum,
d_i is the disparity corresponding to the pseudo-minima.

A large value of Δd implies that the pseudo-minima occur far from the position of the global minimum. In this case the match is considered to be ambiguous unless the matching cost of the global minimum is much smaller than that of the pseudo-minima. On the other hand, a small value of Δd implies that the pseudo-minima are close to the global minimum and the match is considered to be reliable even if the score of the global minimum is not much smaller than that of the pseudo-minima.

If Δd is larger than a certain threshold (due to an ambiguous match), a third test is carried out. This is called the *distinctiveness* test and it compares the error value of the global minimum with that of the pseudo-minima:

$$\Delta C = \sum_{i=1}^{3} |C_i - C_{min}| \tag{4.2.4}$$

where:
C_{min} is the matching cost corresponding to the global minimum,
C_i is the matching cost corresponding to the pseudo-minima.

If ΔC is greater than a certain threshold, the disparity is considered to be valid; otherwise, it is rejected.

The sharpness and distinctiveness tests were adopted from [82].

Figure 4.14 shows different correlation profiles obtained using the correspondence algorithm described. Figures 4.14(a) and 4.14(b) show two profiles where the location of the global minimum is clear and distinct. These profiles satisfy the sharpness test and, therefore, the pixel disparities associated with them are considered to be valid. On the other hand, Figures 4.14(c) and 4.14(d) show two ambiguous correlation profiles. These profiles are the result of repetitive texture in the case of Figure 4.14(c) and uniform texture in the case of Figure 4.14(d). The location of the global minimum cannot be accurately determined from these profiles and, therefore, these profiles are successfully rejected by the sharpness and distinctiveness tests.

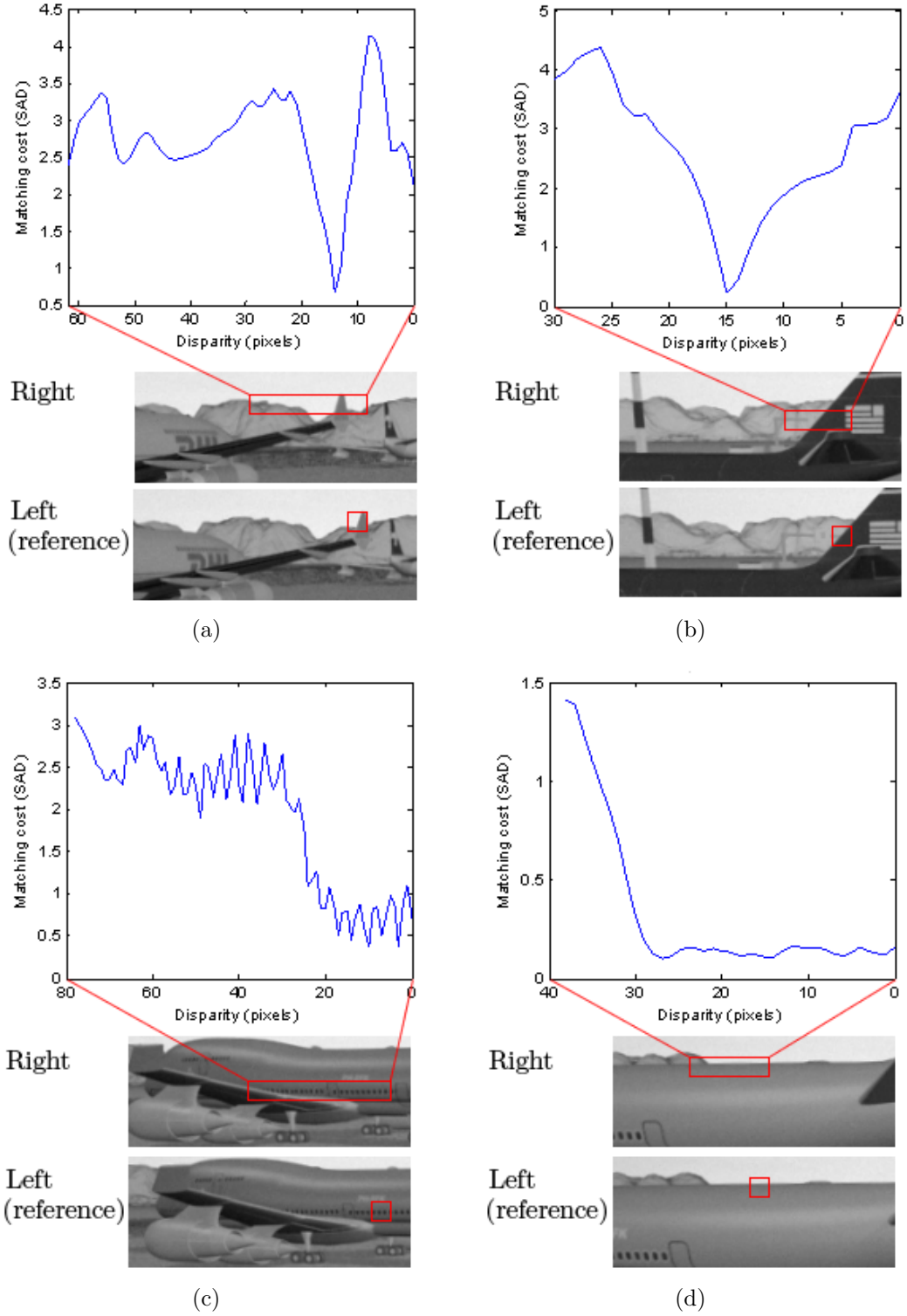


Figure 4.14: Correlation profiles: (a,b) reliable profiles and (c,d) ambiguous profiles detected by the correspondence algorithm

4.2.5 Disparity Refinement

It is assumed that disparity varies smoothly over a very small region of pixels. Therefore, for every valid disparity, a second degree polynomial is fitted to the global minimum and its two closest neighbours to calculate the disparity with sub-pixel precision using Equation (4.2.5):

$$d_{sub-pixel} = d_{min} + \frac{C_{d_{min}-1} - C_{d_{min}+1}}{2(C_{d_{min}-1} - 2C_{d_{min}} + C_{d_{min}+1})} \tag{4.2.5}$$

Figure 4.15 shows an example of sub-pixel interpolation. Through interpolation, the disparity corresponding to the global minimum is found to be about 49.3 pixels instead of 49 pixels.

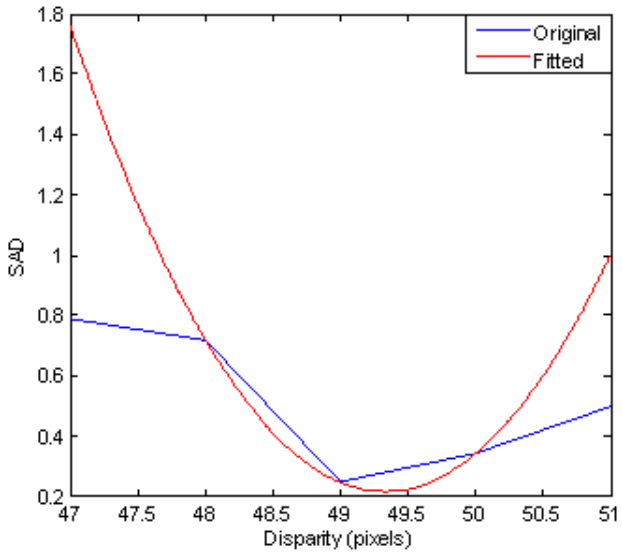


Figure 4.15: Sub-pixel interpolation

4.2.6 Reduction of Computation Time

One of the factors that affect the processing time of correspondence is the disparity search range. This depends on the stereo setup and on the size of the region over which obstacles need to be detected. In this application, the main region of interest is the protection zone. By substituting the calibration parameters into Equation 2.1.6, it is found that the variation of disparity with distance from the cameras is as shown in

Figure 4.16. It can be observed that the disparity of features varies significantly within the protection area (from 208 pixels at 4m to 17 pixels at 50m). If each edge pixel is processed by using the full disparity search range, the computation time complexity will be high. One way of handling such a large range of disparities and reducing the processing time is by using multiresolution (hierarchical) techniques [85, 86].

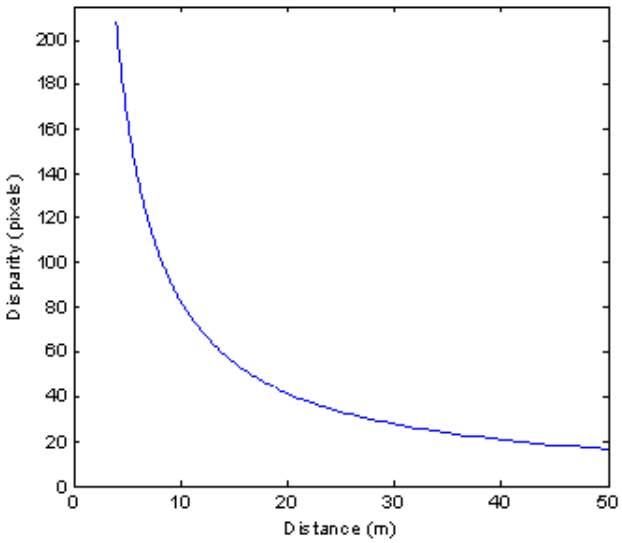


Figure 4.16: Variation of disparity with distance from the cameras

The most common approach consists of sub-sampling the original stereo images in order to create an image pyramid, with very low resolution images at the top and increasingly higher resolution images towards the bottom of the pyramid. The coarse images contain approximate information about the scene. Consequently, correspondence is first carried out on the lowest resolution images using the full disparity search range. This provides rough estimates of disparity. Then, correspondence is carried out at the next level of the pyramid. This time, however, the disparity search range for each pixel (referred to as a *child* pixel) is restricted by the disparity of its *parent* pixel in the previous level. For example, assume that the disparity of a parent pixel is d_p and that the dimensions of images at each level of the pyramid are double those of images at the previous level.[12] Then, the disparity

[12]In this case, each parent pixel would have four child pixels.

search range for a child pixel in the higher resolution image is reduced to:

$$2d_p - \tau \leq d_{search}(pixels) \leq 2d_p + \tau$$

where τ is a user-defined tolerance value. This process is repeated for the remaining levels of the pyramid, with a smaller value of τ being used as image resolution increases. Hence, the disparity search range becomes narrower down the pyramid and a more precise estimate of disparity is obtained.

This strategy speeds up the overall processing time. However, the main disadvantage of this approach is that disparity errors tend to propagate down the pyramid. If the disparity of a parent pixel is incorrect and the true disparity of its child pixels falls outside the restricted disparity search range as a result, the computed disparity of the child pixels will also be incorrect. The correspondence algorithm will not be able to recover from such an error in the remaining levels of the pyramid. Also, the higher up in the pyramid that a disparity error occurs, the larger the number of pixels that are likely to be affected at the bottom of the pyramid (in the full resolution images). One way of trying to prevent disparity errors from propagating through the pyramid would be to use a larger tolerance value and hence widen the disparity search range. However, this would increase the processing time and would defeat the whole purpose of the multiresolution approach. For this reason, a slightly different approach is proposed in this work.

First, a low resolution version of the stereo images is obtained by sub-sampling the original images using bicubic interpolation. The dimensions of the low resolution images are four times smaller than those of the original images.[13] Correspondence is carried out on the coarse images using the full disparity search range given by:

$$0 \leq d_{search}(pixels) \leq 52$$

Due to their very small size, the coarse images are processed very quickly. Then, the maximum and minimum disparities of the images, d_{max} and d_{min}, are determined.

[13]The maximum disparity of the original images is considered to be 208 pixels. This is equivalent to a disparity of $\frac{208}{4} = 52$ pixels in the coarse images.

d_{max} and d_{min} are scaled and the original images are then processed using the following disparity search range:

$$4d_{min} \leq d_{search}(pixels) \leq 4d_{max}$$

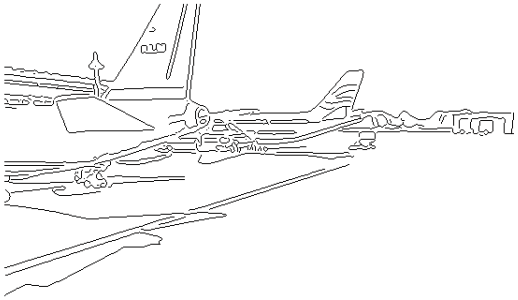
With this method, individual disparity errors that occur when processing the low resolution images are very unlikely to propagate to the original images. This is because the disparity search range for a child pixel in the original images is not directly linked to the disparity of its parent pixel. Therefore, as long as d_{max} and d_{min} are correct, the correspondence algorithm can recover from any disparity errors that occur when processing the coarse images.

The main assumption being made in this implementation is that obstacles will only occupy part of the detection area during any single frame. Hence, the range of disparities will vary between frames and will rarely reach the boundaries of the full disparity search range. For example, if no obstacles are within the protection zone (as in normal operation), the range of disparities will be small. Therefore, computation time is significantly reduced by using this modified multiresolution approach.

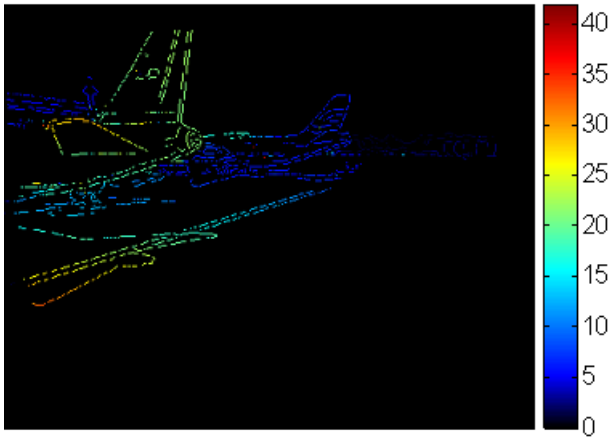
4.2.7 Correspondence Results

Figures 4.17 and 4.18 show the results obtained when carrying out correspondence on two pairs of noisy images featuring typical aerodrome scenes. As expected, objects that are closer to the cameras have a higher disparity. Also, the disparity varies smoothly over individual objects in the scene. In both pairs of images, the maximum disparity is much less than the upper limit of the full disparity search range. The peaks in the disparity histograms give an indication of the number and size of objects in the scene. By looking at the edge map and the disparity map, it can be observed that the disparity of most edge pixels has been computed successfully. Although the majority of unreliable matches are rejected, some isolated pixels with incorrect disparities can still be identified. These are pixels whose disparity differs significantly from that of neighboring pixels. Steps to detect and remove these noisy pixels are discussed in the next chapter.

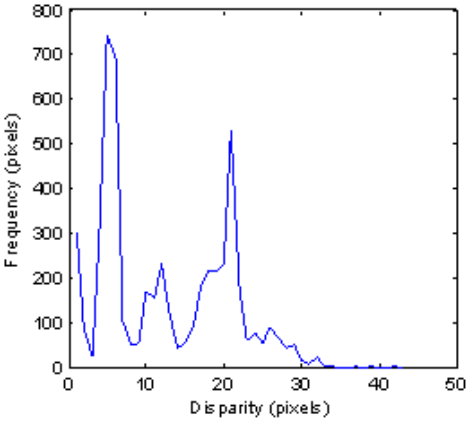
(a) Left intensity image



(b) Left edge map



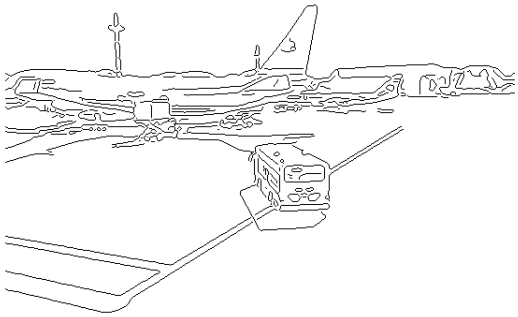
(c) Edge disparity map



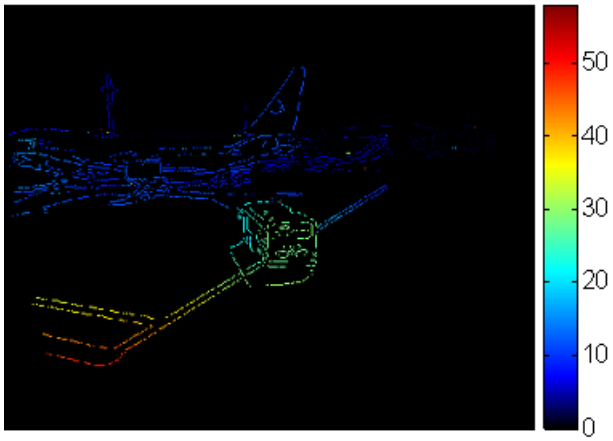
(d) Disparity histogram

Figure 4.17: Correspondence results (Example 1)

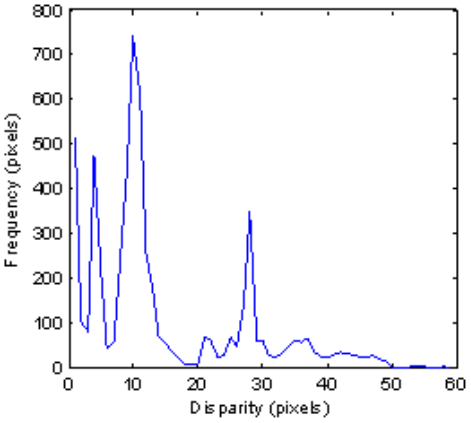
(a) Left intensity image



(b) Left edge map



(c) Edge disparity map



(d) Disparity histogram

Figure 4.18: Correspondence results (Example 2)

Chapter 5

Reconstruction and Obstacle Detection

Reconstruction is the process of recovering 3D information about points in the scene from the 2D coordinates of corresponding points in the stereo images. Section 5.1 looks at how reconstruction was carried out in this work using triangulation techniques. First, the triangulation algorithm is explained. Then, the effects of the baseline distance and focal length on triangulation uncertainty and range accuracy, are discussed. The results of experiments to select appropriate values for these parameters are then presented.

Section 5.2 addresses obstacle detection. First, the method used to detect obstacle points from the results of 3D reconstruction and the absolute extrinsic calibration parameters, is explained. Then, the effect of wing flexing on 3D reconstruction and obstacle detection is discussed. This is followed by an overview of clustering algorithms and an explanation of the clustering algorithm designed to group obstacle points and remove false obstacles. Finally, some obstacle detection results are presented.

5.1 3D Reconstruction

Given a pair of corresponding edge pixels, p_l and p_r (Figure 5.1), triangulation can be used to determine the coordinates of the corresponding 3D point P. In theory, P lies at the point of intersection of the two rays passing through O_l and p_l, and O_r

and p_r respectively. However, in practice, due to a number of errors, the rays will not intersect. Instead, they will converge to a closest point P_1 and P_2 on each ray, respectively. This allows the generation of an estimate of P at P', half way between P_1 and P_2.

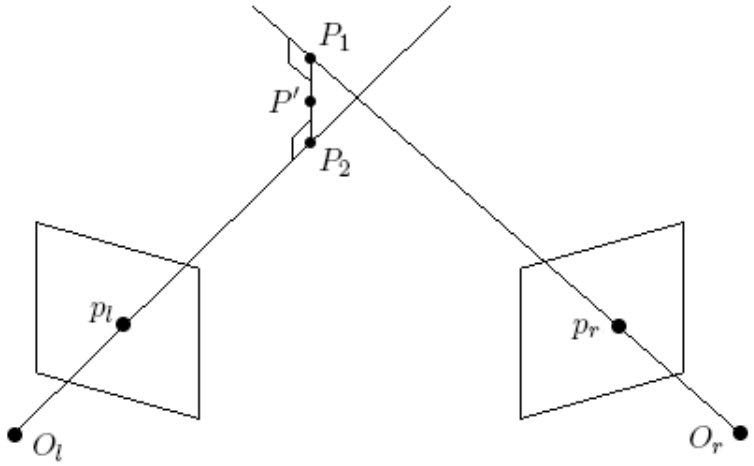


Figure 5.1: Reconstruction by triangulation

5.1.1 The Triangulation Algorithm

The triangulation algorithm used in this work was implemented as part of the calibration toolbox [66]. The following is a mathematical description of this algorithm.

As shown in Equation (3.1.3), a point $\boldsymbol{P_l} = (X_l, Y_l, Z_l)$ in the left CRF and its corresponding point $\boldsymbol{P_r} = (X_r, Y_r, Z_r)$ in the right CRF are related as follows:

$$\boldsymbol{P_r} = \boldsymbol{R_{rel}}\boldsymbol{P_l} + \boldsymbol{T_{rel}} \tag{5.1.1}$$

where $\boldsymbol{T_{rel}}$ and $\boldsymbol{R_{rel}}$ are the relative position and orientation between the stereo cameras, respectively. The normalised projections $\boldsymbol{p_{ln}}$ and $\boldsymbol{p_{rn}}$ of $\boldsymbol{P_l}$ and $\boldsymbol{P_r}$ are given by:

$$\begin{aligned} \boldsymbol{p_{ln}} &= \frac{\boldsymbol{P_l}}{Z_l} = (x_{ln}, y_{ln}, 1)^T \\ \boldsymbol{p_{rn}} &= \frac{\boldsymbol{P_r}}{Z_r} = (x_{rn}, y_{rn}, 1)^T \end{aligned} \tag{5.1.2}$$

The normalised coordinates are obtained from the coordinates of corresponding edge pixels using Equation (4.1.8). From Equations (5.1.1) and (5.1.2), the following relation is obtained:

$$\boldsymbol{p_{rn}} Z_r = \boldsymbol{R_{rel}} \boldsymbol{p_{ln}} Z_l + \boldsymbol{T_{rel}} \tag{5.1.3}$$

This can be expressed in matrix form as

$$\begin{pmatrix} \boldsymbol{p_{rn}} & -\boldsymbol{R_{rel}} \boldsymbol{p_{ln}} \end{pmatrix} \begin{pmatrix} Z_r \\ Z_l \end{pmatrix} = \boldsymbol{T_{rel}} \tag{5.1.4}$$

Let $\boldsymbol{\alpha_l} = -\boldsymbol{R_{rel}} \boldsymbol{p_{ln}}$ and $\boldsymbol{A} = \begin{pmatrix} \boldsymbol{p_{rn}} & \boldsymbol{\alpha_l} \end{pmatrix}$. Z_l and Z_r can then be obtained as follows:

$$\begin{aligned}
\begin{pmatrix} Z_r \\ Z_l \end{pmatrix} &= (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{T_{rel}} \\
&= \left(\begin{pmatrix} \boldsymbol{p_{rn}} \\ \boldsymbol{\alpha_l} \end{pmatrix} \begin{pmatrix} \boldsymbol{p_{rn}} & \boldsymbol{\alpha_l} \end{pmatrix} \right)^{-1} \begin{pmatrix} \boldsymbol{p_{rn}} \\ \boldsymbol{\alpha_l} \end{pmatrix} \boldsymbol{T_{rel}} \\
&= \begin{pmatrix} \boldsymbol{p_{rn}} \cdot \boldsymbol{p_{rn}} & \boldsymbol{p_{rn}} \cdot \boldsymbol{\alpha_l} \\ \boldsymbol{\alpha_l} \cdot \boldsymbol{p_{rn}} & \boldsymbol{\alpha_l} \cdot \boldsymbol{\alpha_l} \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{p_{rn}} \\ \boldsymbol{\alpha_l} \end{pmatrix} \boldsymbol{T_{rel}} \\
&= \frac{1}{Det} \begin{pmatrix} \boldsymbol{\alpha_l} \cdot \boldsymbol{\alpha_l} & -\boldsymbol{p_{rn}} \cdot \boldsymbol{\alpha_l} \\ -\boldsymbol{\alpha_l} \cdot \boldsymbol{p_{rn}} & \boldsymbol{p_{rn}} \cdot \boldsymbol{p_{rn}} \end{pmatrix} \begin{pmatrix} \boldsymbol{p_{rn}} \\ \boldsymbol{\alpha_l} \end{pmatrix} \boldsymbol{T_{rel}}
\end{aligned} \tag{5.1.5}$$

where $Det = (\boldsymbol{\alpha_l} \cdot \boldsymbol{\alpha_l})(\boldsymbol{p_{rn}} \cdot \boldsymbol{p_{rn}}) - (\boldsymbol{\alpha_l} \cdot \boldsymbol{p_{rn}})(\boldsymbol{p_{rn}} \cdot \boldsymbol{\alpha_l})$. Therefore, Z_l and Z_r are given by:

$$Z_r = \frac{(\boldsymbol{\alpha_l} \cdot \boldsymbol{\alpha_l})(\boldsymbol{p_{rn}} \cdot \boldsymbol{T_{rel}}) - (\boldsymbol{p_{rn}} \cdot \boldsymbol{\alpha_l})(\boldsymbol{\alpha_l} \cdot \boldsymbol{T_{rel}})}{(\boldsymbol{\alpha_l} \cdot \boldsymbol{\alpha_l})(\boldsymbol{p_{rn}} \cdot \boldsymbol{p_{rn}}) - (\boldsymbol{\alpha_l} \cdot \boldsymbol{p_{rn}})(\boldsymbol{p_{rn}} \cdot \boldsymbol{\alpha_l})} \tag{5.1.6}$$

$$Z_l = \frac{(-\boldsymbol{\alpha_l} \cdot \boldsymbol{p_{rn}})(\boldsymbol{p_{rn}} \cdot \boldsymbol{T_{rel}}) + (\boldsymbol{p_{rn}} \cdot \boldsymbol{p_{rn}})(\boldsymbol{\alpha_l} \cdot \boldsymbol{T_{rel}})}{(\boldsymbol{\alpha_l} \cdot \boldsymbol{\alpha_l})(\boldsymbol{p_{rn}} \cdot \boldsymbol{p_{rn}}) - (\boldsymbol{\alpha_l} \cdot \boldsymbol{p_{rn}})(\boldsymbol{p_{rn}} \cdot \boldsymbol{\alpha_l})} \tag{5.1.7}$$

$\boldsymbol{P_l}$ and $\boldsymbol{P_r}$ are recovered using Equation (5.1.2):

$$\begin{aligned}
\boldsymbol{P_r} &= \boldsymbol{p_{rn}} Z_r \\
\boldsymbol{P_l} &= \boldsymbol{p_{ln}} Z_l
\end{aligned} \tag{5.1.8}$$

$\boldsymbol{P_r}$ is expressed in the left CRF as follows:

$$\boldsymbol{P_1} = \boldsymbol{R_{rel}}^T(\boldsymbol{P_r} - \boldsymbol{T_{rel}}) \tag{5.1.9}$$

Let $\boldsymbol{P_2} = \boldsymbol{P_l}$. Then, the approximate point of intersection is given by the midpoint of $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ (Figure 5.1):

$$\boldsymbol{P_{left}} = \frac{(\boldsymbol{P_1} + \boldsymbol{P_2})}{2} \tag{5.1.10}$$

where $\boldsymbol{P_{left}}$ is an expression of $\boldsymbol{P'}$ in the left CRF. $\boldsymbol{P_{left}}$ is expressed in the WRF using Equation (3.1.1) as follows:

$$\boldsymbol{P_w} = \boldsymbol{R_l}^T(\boldsymbol{P_{left}} - \boldsymbol{T_l}) \tag{5.1.11}$$

where $\boldsymbol{R_l}$ and $\boldsymbol{T_l}$ are the rotation matrix and translation vector respectively.[1]

5.1.2 Selection of Baseline Distance and Focal Length

Errors in triangulation occur for three main reasons: (a) triangulation uncertainty, (b) calibration errors and (c) correspondence errors. Figure 5.2(a) illustrates triangulation uncertainty. Assuming that disparity is calculated with a precision of 1 pixel, regions of uncertainty are given by the intersection of lines going through the optical centre of each camera and the boundaries of pixels in the image plane. For instance, the region of uncertainty of point P is shaded in grey.

As can be observed, triangulation uncertainty increases with distance from the cameras. This affects both the range resolution and range accuracy of the system. Range resolution Δz is obtained by differentiating Equation (2.1.6) as follows:

$$\begin{aligned} z &= \frac{bf}{d} \\ \frac{\Delta z}{\Delta d} &= \frac{-bf}{d^2} \end{aligned} \tag{5.1.12}$$

where:

d is the disparity (pixels),

Δd is the precision of the disparity (pixels),

[1] $\boldsymbol{R_l}$ and $\boldsymbol{T_l}$ are determined during absolute extrinsic calibration.

z is the range (m),

b is the baseline distance (m),

f is the focal length (pixels).

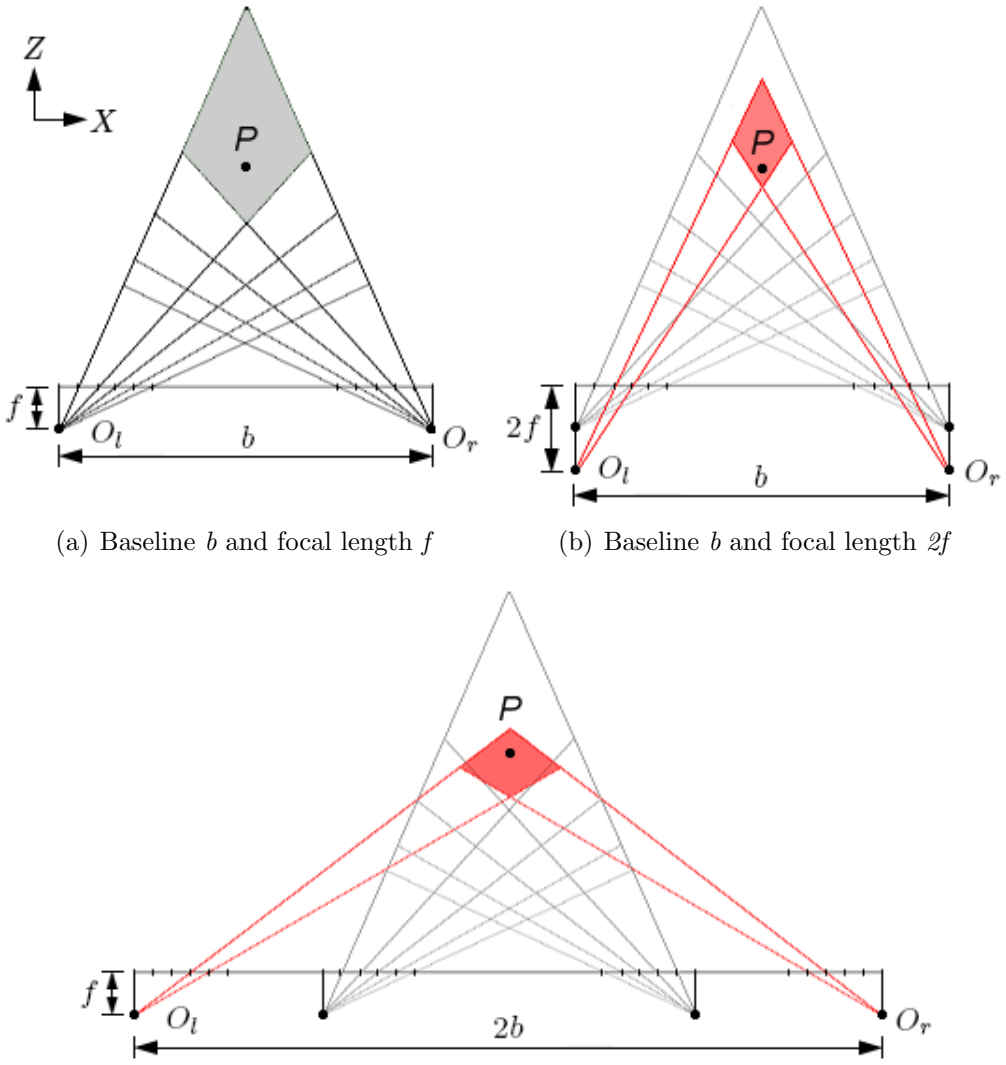
From Equation (2.1.6), $d = \frac{bf}{z}$. Substituting in Equation (5.1.12), this gives

$$\begin{aligned} \frac{\Delta z}{\Delta d} &= \frac{-z^2}{bf} \\ |\Delta z| &= \frac{z^2 \Delta d}{bf} \end{aligned} \tag{5.1.13}$$

From this equation it can be observed that range resolution (and triangulation uncertainty) can be improved by increasing the baseline distance and/or focal length or by reducing Δd. Δd is reduced by computing the disparity of a pixel with sub-pixel precision. With the correspondence method described in Chapter 4, disparity is calculated with a precision of 0.25 pixels. Increasing the focal length or baseline distance reduces the uncertainty region of point P as shown in Figures 5.2(b) and 5.2(c) respectively. Increasing the baseline distance tends to reduce the uncertainty mostly in the z axis whereas an increase in the focal length tends to reduce the uncertainty mostly in the x axis.

Apart from reducing triangulation uncertainty, an increase in the baseline distance and/or focal length results in a larger disparity value for a particular object at a certain depth. This reduces the impact of disparity errors on triangulation. Referring to Equation (2.1.6), if an object has a disparity of 10 pixels, a disparity error of 1 pixel will introduce an error of about 10% in the range measurement. However, if the same object has a disparity of 100 pixels, the same disparity error will reduce the range measurement error to only about 1%.

However, increasing the baseline distance and/or focal length has some disadvantages. The disparity of an object at a given range is increased, consequently necessitating an increase in the disparity search range. This, in turn, increases the computation time. Also, the common FOV of the stereo vision system is reduced. This reduces the ability of the system to find corresponding points, especially at close range. As mentioned in Section 4.2.1.1, an increase in baseline distance results in



(a) Baseline b and focal length f

(b) Baseline b and focal length $2f$

(c) Baseline $2b$ and focal length f

Figure 5.2: Variation of triangulation uncertainty with changes in baseline distance and focal length

greater projective distortion between the stereo images and increases the possibility of occlusions. Therefore, it is necessary to make a compromise when choosing the baseline distance and focal length.

In order to choose suitable values for the baseline distance and focal length, an experiment was carried out to test the positional accuracy of the system for 20 combinations of these two parameters. The test values for each parameter are given in Table 5.1.

Table 5.1: Values used for baseline distance and focal length tests

b (m)	f (pixels)
0.5	773 (horizontal FOV = 45°)
1	554 (horizontal FOV = 60°)
1.5	417 (horizontal FOV = 75°)
2	320 (horizontal FOV = 90°)
2.5	

For each of the 20 combinations of b and f, the simulation environment was used to place a small textured object on the ground at different positions on a grid measuring 20m by 55m along the x and z axes respectively of the WRF. Synthetic stereo images were captured for each of these positions and gaussian noise was added to them. Then, the following steps were carried out for each object position:

1. The edge disparity map was obtained from the stereo images.

2. 3D reconstruction was carried out and the edge points were projected onto the WRF.

3. Points corresponding to the object were filtered from the rest of the points by measuring their height above the ground. Points higher than a certain threshold were classified as object points.

4. A depth map was constructed by plotting the x and z coordinates of the object points.

5. The centroid of the group of points corresponding to the object was found.

6. The distance (error) between the correct (actual) and measured object positions was calculated.

In order not to influence the outcome of the experiment by any calibration errors, the camera parameters were assumed to be known for each stereo setup. The complete results of this experiment are presented in Appendix D. Figure 5.3 shows the *total* distance error obtained for each combination of baseline distance and focal length.[2] In general it can be observed that, when the focal length is increased for a particular value of baseline distance, the total distance error decreases (indicating an improvement in positional accuracy). This is because an increase in the focal length results in a decrease in the triangulation uncertainty. This is mostly evident when the baseline distance is 0.5m. For the other baseline distance values, the greatest decrease in the total distance error occurs when increasing the focal length from 320 pixels to 417 pixels.

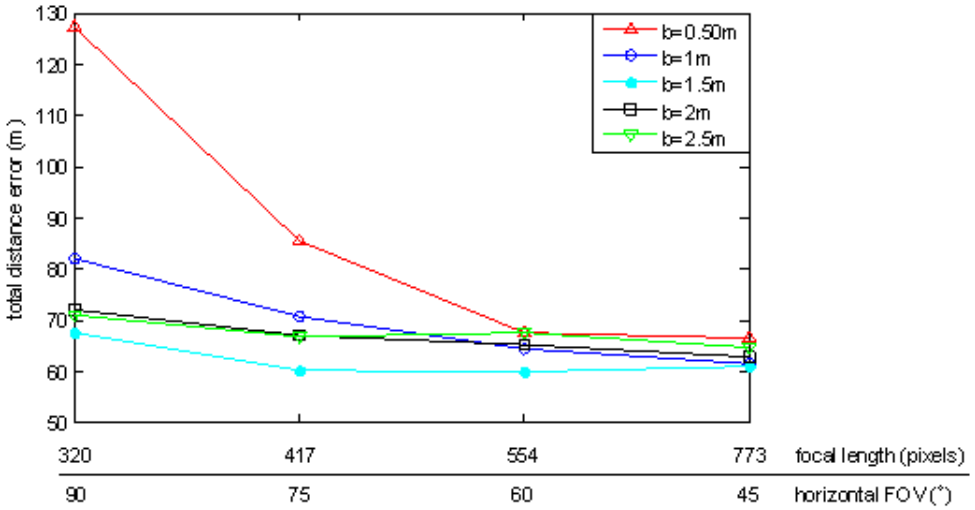


Figure 5.3: Variation of total distance error with baseline distance and focal length

The triangulation uncertainty is also reduced by increasing the baseline distance. In fact, a decrease in the total distance error is observed when increasing the baseline

[2]The total distance error is the sum of the individual distance errors obtained at each of the test positions in the WRF.

distance (for a particular value of focal length) from 0.5m to 1m and from 1m to 1.5m. However, when the baseline distance is increased further, the total distance error starts increasing. This happens because, at these longer baseline distances, the projective distortion in the stereo images is more pronounced and has a larger negative impact on correspondence. This effectively cancels the benefits resulting from a reduction in triangulation uncertainty due to an increase in baseline distance. Therefore, in this case, there is no added benefit in increasing the baseline distance beyond 1.5m.

Since the lowest total distance errors were obtained with a baseline distance of 1.5m, the baseline distance was set to 1.5m. With this baseline distance, there is very little difference in the total distance error obtained with focal length values of 417 pixels, 554 pixels and 773 pixels. Since a shorter focal length implies less computation time, it seems logical to select a focal length of 417 pixels. However, since this experiment was conducted under ideal conditions (where the camera parameters were known) and it is expected that the positional error will be larger in practice, it was decided to select a longer focal length (554 pixels) in order to have better positional accuracy.

The decrease in triangulation uncertainty as a result of an increase in baseline distance can be observed in Figure 5.4. In this example, the textured object is at position (x=0m,z=50m) and the horizontal FOV is 60°. The points corresponding to the object are plotted in the xz plane of the WRF. As the baseline distance is increased, these points become more compact along the z axis, implying that the region of triangulation uncertainty is reduced.

The positional accuracy of the system was determined using the values chosen for the baseline distance and focal length, with the difference that the system was calibrated first. The results are shown in Figure 5.5. As expected, calibration errors reduce the positional accuracy. The positional error is less than 0.8m and 2m in the x and z axes respectively.

The range resolution of the system was calculated for different range values using

Figure 5.4: Plan view of points corresponding to the textured object at position (x=0m,z=50m) when the horizontal FOV is 60° and b is (a) 0.5m, (b) 1m, (c) 1.5m, (d) 2m, (e) 2.5m

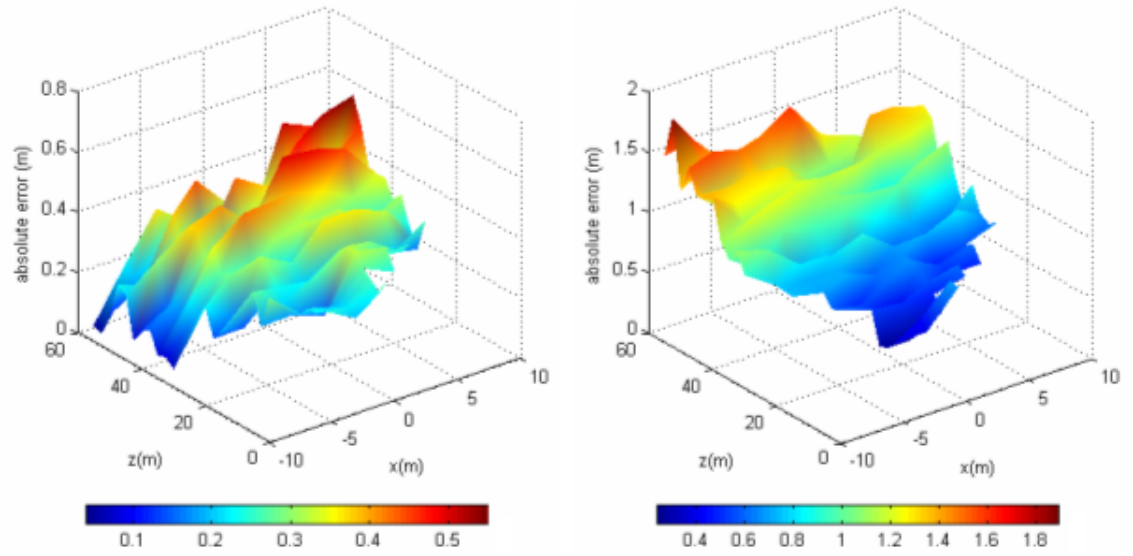


Figure 5.5: Plot of error in (a) x axis and (b) z axis with respect to position of textured object in the WRF (These are the results obtained with the calibrated system)

Equation (5.1.13) (with $b = 1.5$m, $f = 554$ pixels and $\Delta d = 0.25$ pixels) and is shown in Figure 5.6.

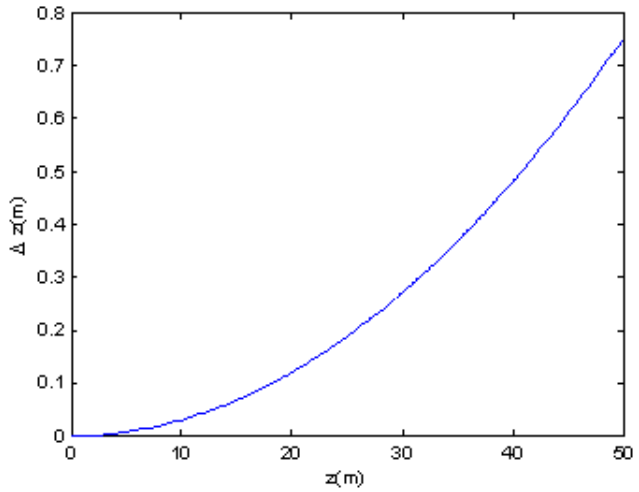


Figure 5.6: Variation of range resolution with distance from the cameras

5.2 Obstacle Detection

5.2.1 Ground Modeling

In most ground-based stereo vision obstacle detection systems, obstacles are defined as objects that do not belong to the ground. Hence, a model is required to define the ground surface.

In certain applications, the ground (road) surface is assumed to be flat within the region of interest of the system [87, 88]. This allows the ground to be modeled offline as part of the calibration process. Such modeling can be done by fitting a planar equation to the ground or by computing the disparity of ground pixels. However, when the road topography is constantly changing (e.g. due to land features such as hills or valleys) and in off-road situations, the planar ground assumption does not hold and the ground has to be modeled in each frame.

In [2,3], the longitudinal profile of the road is extracted by computing the disparity of ground pixels in each row of the stereo images. First, a pair of stereo images is captured with the assumption that the road occupies a large area of these images. Then, the disparity map is obtained as shown in Figure 5.7(a). This disparity map is represented in a different form known as the *V-disparity map* (Figure 5.7(b)). This has the same number of rows as the original disparity map. The intensity of a pixel with row coordinate r in the V-disparity map represents the number of pixels along row r in the original disparity map that have a particular disparity. Since ground features occupy most of the stereo images and ground pixels on the same row have the same disparity, the maximum intensity value in each row of the V-disparity map corresponds to the disparity of ground pixels in a particular row. The longitudinal road profile is modeled as a piecewise linear curve by identifying lines in the V-disparity map using the Hough Transform (Figure 5.7(c)).

In [79], the 3D points belonging to the ground are projected laterally as shown in Figure 5.8. Then, the longitudinal profile of the road is obtained by fitting a curve to these points.

The methods of extracting the road profile as proposed in [2, 3, 79] rely on the

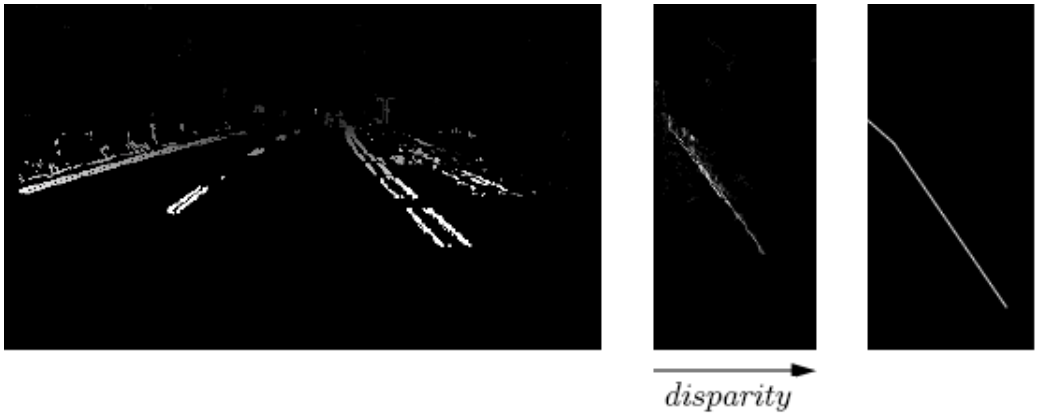


Figure 5.7: Modeling of longitudinal road profile using the method described in [2,3]: (a) disparity map, (b) V-disparity map, (c) Hough Transform image

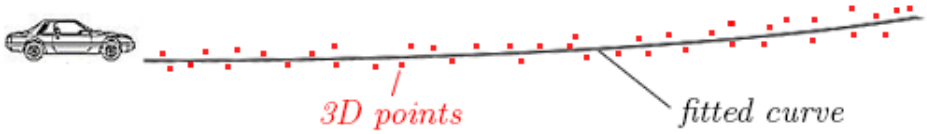
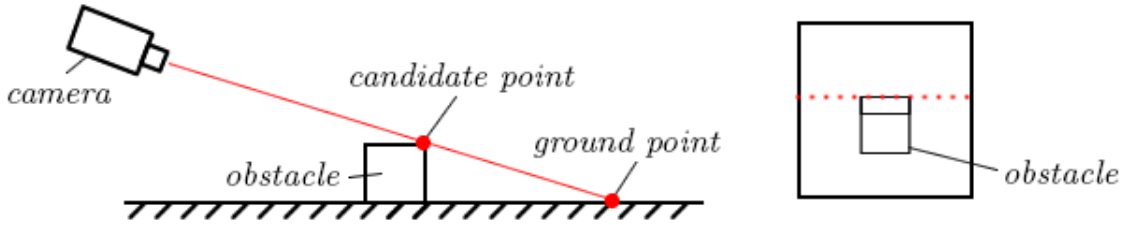


Figure 5.8: Lateral projection of ground points and extraction of road profile by curve fitting

presence of good ground features in each frame. Therefore, they will not produce reliable results in low texture conditions or when obstacles occupy a large area of the frame.

Once the ground has been modeled, it is possible to distinguish between ground features and obstacles. One method is to compare the disparity of a candidate pixel with the disparity of a ground pixel on the same scanline. In the example shown in Figure 5.9(a), the candidate point and the ground point are projected onto the same image scanline which is represented by the red dotted line in Figure 5.9(b). Since the candidate point is closer to the camera than the ground point, the disparity of the candidate pixel is larger than that of the ground pixel. Therefore, the candidate pixel is classified as an obstacle point.

Another method of identifying obstacles is to measure the height of points with



(a) Candidate point and ground point (b) Image projection

Figure 5.9: Distinguishing between ground features and obstacles

respect to the ground. Points that are above the ground, or whose height exceeds a certain threshold, are classified as obstacle points [79, 88]. Orientation information can also be used to identify potential obstacles and ground features. In [89], obstacles are identified by measuring the angle between straight 3D features and the ground. If this angle is larger than a certain value, the 3D features are classified as obstacles. Similarly, in [90], points are classified as road features if their normal vector lies within a certain range from the normal of the ground plane.

For the application considered in this research, a planar ground surface is assumed. This assumption holds for aerodrome areas since the ground is relatively flat in the vicinity of an aircraft. As mentioned previously, points are expressed in the WRF during 3D reconstruction. The xz plane of the WRF coincides with the ground plane. Therefore, potential obstacle points are easily detected by height thresholding.

As a result of obstacle detection, a binary obstacle map is produced. This contains all the edge pixels that are classified as potential obstacle points. Isolated pixels in this map are detected by checking the 3-by-3 neighborhood of each obstacle pixel. If none of the neighboring pixels is an obstacle, the obstacle pixel is assumed to be a noisy point and is removed.

5.2.2 Wing Bending Considerations

As an aircraft manoeuvres on the ground, the wings undergo a certain amount of vertical flexing as shown in Figure 5.10. The amount of flexing increases with

wingspan. As the cameras are installed at the wingtips, they will essentially oscillate vertically as the aircraft moves. This changes the position and attitude of the cameras with respect to the ground plane. The main parameter affected is camera height above the ground. Camera height also varies with other parameters such as aircraft payload and the quantity of fuel in the wings. For instance, departing aircraft have more fuel - and, hence, heavier wings - than arriving aircraft.

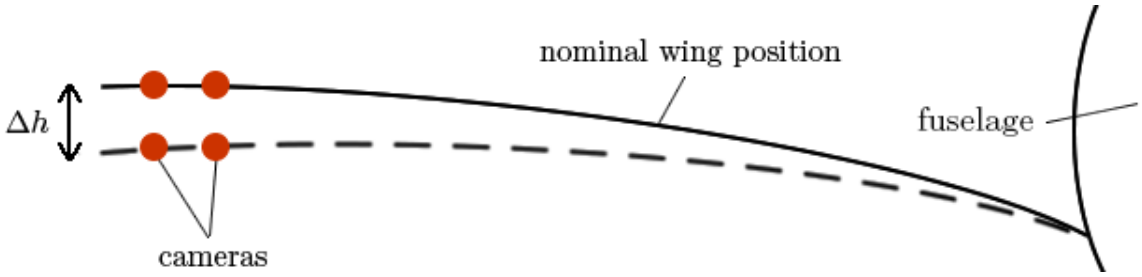


Figure 5.10: Wing flexing

The change in camera height can affect the reconstruction and obstacle detection process. For instance, when the cameras move downwards, the physical threshold that is used to separate ground features from obstacles shifts downwards as well. This implies that ground features might be wrongly classified as obstacle points. In order to compensate for this, one option is to determine the absolute position of the cameras with respect to the WRF in each frame. Different online calibration methods, used to extract some or all of the absolute extrinsic calibration parameters, are described in [59, 91, 92]. The methods suggested in [91, 92] are designed for automotive applications and rely on the presence of road markings. They will fail if these features are either not detected or are absent from the scene. The method suggested in [59] assumes that the ground occupies the largest part of the image and fails either if wide obstacles are present in the image or if there is insufficient ground texture.

Instead of measuring absolute camera position, one can monitor the change in camera height between consecutive frames. This approach does not impose constraints on the structure of the scene and consists of tracking one (or more) corresponding pairs of pixels in an image sequence. The 3D points corresponding

to the stereo pixels are determined in each frame by triangulation. Assuming that the tracked pixels correspond to static scene features, any change in the height of the 3D points between frames can be attributed to a change in ownship camera height. Thus, the absolute height of the cameras can be updated in each frame. The main drawback of this method is that errors in camera height will accumulate over time. This effect gets worse when the image does not contain sufficiently good features that are easy to track.

Online calibration is not a suitable option for this application. Although ground markings are present in the aerodrome environment, they cannot be guaranteed to be present in the scene or to lie within the camera FOV. Similarly, the ground may not always occupy the largest area of an image and, even if it does, it is very likely to have uniform texture. For these reasons, another option was considered in this work. This consists of increasing the height threshold for obstacle detection. The threshold can be increased to a value that is equal to (or greater than) the worst case change in camera height due to wing bending and aircraft loading. The aircraft considered in this research has a semi-wingspan in the range of 40m and it is assumed that the maximum change in camera height will be 1m. Therefore, the height of the cameras above the ground will be expected to vary between 8m (when the aircraft is stationary and unloaded) and 7m. Since the distance between the stereo cameras is very short in relation to the wingspan, it is assumed that both cameras will undergo the same translation in each frame. It is also assumed that the cameras will not undergo any rotation. Therefore, in the worst case scenario, ground features will still be classified correctly if the height threshold is set to 1m.

The drawback of increasing the height threshold is that, when the wing is at its nominal position, the system will not detect obstacles lower than 1m above the ground. This is however not a problem in this application because the obstacles being targeted, particularly aircraft extremities, are higher than this threshold. Therefore, this option was selected and the height threshold was set to 1m.

5.2.3 Clustering

5.2.3.1 Background

Due to errors in the previous stages of stereo vision, certain edge points are mapped onto incorrect positions in the WRF. As a result, height thresholding is not sufficient to filter out ground features and incorrect 3D points. Therefore, after detecting potential obstacle points, the next step is to group these points into individual obstacles. By the end of this process, the remaining 'noisy' points are removed while the true obstacle points are retained.

Individual obstacles are obtained from the set of potential obstacle points by means of clustering techniques. Clustering is the process of organising data sets (such as a set of 3D points) into groups (called *clusters*) based on a number of neighborhood and similarity criteria. Ideally, clustering should maximise the distance[3] between clusters (also known as the *inter-cluster distance*) and minimise the distance between points in the same cluster (also known as the *intra-cluster distance*). The clustering criteria used and the definitions of the distances vary according to the data set being considered. There are two main classifications of clustering techniques: hierarchical clustering and non-hierarchical (partitional) clustering.

Hierarchical clustering is further divided into agglomerative and divisive clustering. Agglomerative clustering follows a bottom-up approach. Each point is initially treated as a cluster. Then, the two closest clusters are merged into a single cluster. This merging process is repeated by measuring the distance between clusters and merging the two closest clusters. As the clusters become larger, more distant clusters are linked together and the dissimilarity between elements of the same cluster increases. Eventually, all the points are grouped into a single cluster. This creates a hierarchy (or tree) of clusters where each cluster is made up of smaller clusters. In practice, the merging process can be stopped when a suitable number of clusters have been formed or when the distance between clusters crosses a certain threshold.

Divisive clustering follows a top-down approach. The points are initially treated

[3]In the context of clustering, the word 'distance' does not necessarily mean 'physical distance'.

as one cluster. This is repeatedly divided into smaller clusters until some stopping condition is met. In the limit, the number of clusters is equal to the total number of points. Divisive clustering assumes that the points are initially connected in some form of tree or mesh.

One definition of inter-cluster distance is that of the *nearest-neighbour* which states that the distance between two clusters is the distance between the two closest elements (points) of the clusters. With this distance measure, points that are far apart are connected by a chain of close objects. This measure is suitable when detecting elongated clusters and when the clusters are not spherical or compact. If agglomerative clustering is used and the merging process is allowed to continue until a single cluster is formed, the result is a Minimum Spanning Tree (MST). The tree is 'minimum' in the sense that the sum of the distances between points in the tree is minimised. The tree consists of nodes (points) connected by edges. The MST can be used as a starting point for divisive clustering. Initially, the tree is treated as a single cluster. Separate clusters are obtained by removing edges that are longer than a certain threshold or by recursively removing the longest edge until a particular number of clusters are obtained.

Other definitions of inter-cluster distances are used to detect different types of clusters. These include *farthest-neighbour* (which finds the distance between the farthest elements of the clusters), *average linkage* (which finds the average distance between clusters) and *centroid linkage* (which finds the distance between cluster centroids).

The second category of clustering techniques is partitional clustering. These techniques divide the data set into clusters, typically by trying to minimise some criterion or error function. For instance, the error function could be one that minimises intra-cluster distance while maximising inter-cluster distance. One common example of partitional clustering is K-means clustering. The number of clusters has to be known a priori and *seed* points are selected (usually randomly) as cluster centers at the beginning of the algorithm. During the algorithm, each point

is assigned to a particular cluster and the cluster centers are updated. This process is repeated until the cluster centers do not move any more (i.e. each point remains associated with the same cluster) or some other condition is met. This algorithm can be repeated for different numbers of clusters and/or different initial seed points and the combination that minimises the error function is selected. A derivative of K-means clustering is fuzzy clustering. In this case, a point can be a member of multiple clusters. The degree of membership in a cluster is continuous and depends, for instance, on the distance between a point and the cluster centroid. K-means and fuzzy clustering tend to favor compact, spherical clusters and rely more on the statistical attributes of clusters rather than on their geometrical properties.

Another partitional algorithm is the Self-Organising Map (SOM). This is a type of Artificial Neural Network (ANN) which can determine an appropriate number of clusters. However, it needs to be trained.

5.2.3.2 Outline of the clustering algorithm

As mentioned in previous chapters, the system needs to be able to detect a wide range of obstacles, particularly aircraft extremities and vehicles. The number and size of obstacles in each frame is therefore unknown. The shape of the obstacles and the fact that only the edge pixels are processed suggests that the clusters will tend to be elongated. Taking these points into consideration, it was decided to implement a new agglomerative, hierarchical clustering technique. As will be shown, this algorithm uses both spatial and non-spatial attributes to cluster obstacle points.

Initially, the obstacle points are treated as individual clusters. The clustering algorithm then proceeds as follows:

1. The first point is selected from the 3D point cloud and is labelled as *Cluster n*. This point is defined as the *root*. n is initialised to 1.

2. A search is carried out for points that are within a specific distance from the *root*. These points are defined as *children* of the *root*.

3. A search is carried out for points that are within a specific distance from each *child*. This step is repeated for any subsequent *children* until no more *children* are found. All *children* are assigned the same label (*Cluster n*) as the *root*.

4. All points labelled as *Cluster n* are removed from the 3D point cloud and n is incremented by 1.

5. Steps (1)-(4) are repeated until the 3D point cloud is empty and all the points are labelled.

6. Any clusters that do not satisfy certain criteria are removed.

The clustering process is illustrated through a simple 2D example in Figure 5.11. It can be observed that *Cluster 1* is formed after 4 iterations of Step (3) whereas *Cluster 2* is formed after 2 iterations.

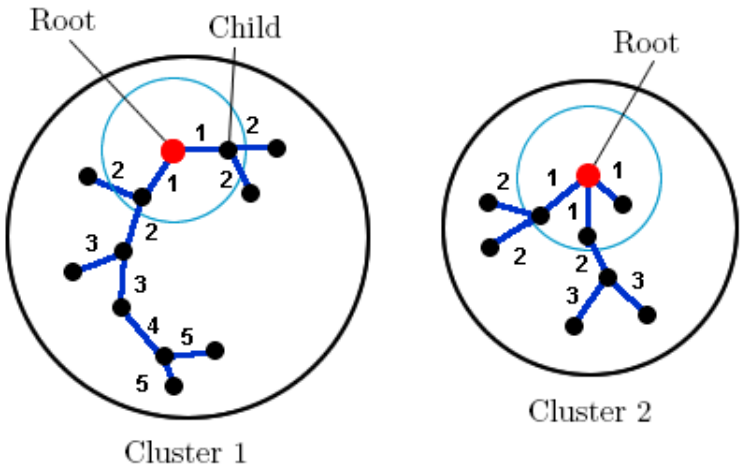


Figure 5.11: Clustering

After clustering, the remaining obstacle points are transformed from the WRF to the ARF using Equation (3.1.5).

5.2.3.3 Grouping and filtering criteria

During Steps (2) and (3) of the clustering algorithm, the distance between a point and the root/child is defined in terms of three grouping criteria (which are weighted such that the individual weights add up to 1):

- C_{3d} - The 3D Euclidean distance between the two points. This is the most important measure and is given the greatest weighting ($w_1 = 0.5$).

- C_{2d} - The 2D Euclidean distance between the pixels corresponding to the two points. This is an important measure which is made under the assumption that points that are close in 3D space are also close in the image plane. It is not as important as the first criterion because it is possible that neighboring points in the image plane are far apart in 3D space. This can happen, for instance, at object boundaries. This criterion is given a weighting of $w_2 = 0.4$.

- C_{int} - The absolute difference in intensity between the pixels corresponding to the two points.[4] It is expected that, if the pixels belong to the same object and are situated close to each other, the intensity difference between them will be very small. However, it is also possible that completely unrelated pixels have the same intensity. For this reason, this measure is given the smallest weighting ($w_3 = 0.1$).

Rather than imposing 'hard' thresholds for each criterion (i.e. a criterion is either met or not), it was decided to use 'soft' thresholding where C_{3d}, C_{2d} and C_{int} are mapped onto score values between 0 and 1 as shown in Figure 5.12.[5] Since the three criteria have different units, the mapping also serves to normalise the data. Then, each score is multiplied by its corresponding weighting value and the overall distance D between the two points is calculated as follows:

$$D = w_1 S_{3d} + w_2 S_{2d} + w_3 S_{int} \tag{5.2.1}$$

where:

S_{3d}, S_{2d} and S_{int} are the score values associated with each criterion,

$w_1 = 0.5$, $w_2 = 0.4$ and $w_3 = 0.1$ are the weighting values associated with each criterion,

$w_1 + w_2 + w_3 = 1$.

[4]The maximum possible intensity difference between the pixels is 255 grey levels.

[5]The values chosen for thresholds $t_1..t_6$ are presented at the end of this section.

As can be observed, D can vary between 0 and 1. If D is greater than a certain threshold $thres_1 = 0.7$, the two points are considered to belong to the same cluster.

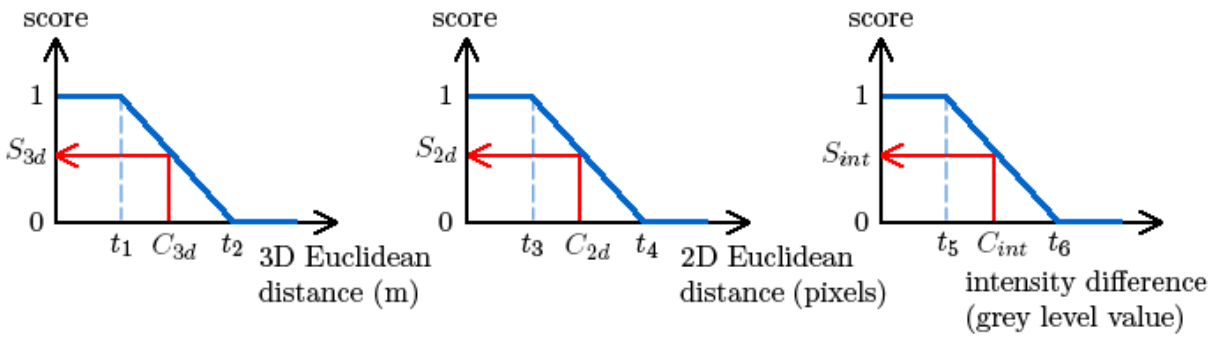


Figure 5.12: Mapping of grouping criteria onto score values

Due to the use of soft thresholding, the clustering algorithm is more flexible in the way that it retains obstacle points. This is because the score values corresponding to the individual grouping criteria can compensate for each other. For example, if an obstacle point (whose distance from the root/child is being measured) has high, average and low scores corresponding to C_{3d}, C_{2d} and C_{int} respectively, it is as likely to be retained by the algorithm as another obstacle point that scores average values for each of the three criteria.

In Step (6) of the clustering algorithm, the following weighted criteria are used to filter the clusters:

- $C_{\overline{3d}}$ - The average 3D distance between neighboring points in the cluster. This gives an indication of point density. The greater the density, the more likely it is that the cluster is an obstacle. This criterion is given a weighting of $w_4 = 0.4$.

- $C_{\overline{2d}}$ - The average 2D distance between pixels corresponding to neighboring points in the cluster. Like the first criterion, this gives an indication of point density. However, because of the possibility that neighboring pixels might be far apart in 3D (as mentioned when describing C_{2d}), this criterion is given a smaller weighting ($w_5 = 0.3$).

- C_{pts} - The number of points in the cluster. This gives an idea of obstacle size.

Given the different sizes of obstacles that are likely to be present in the scene, the number of points can vary considerably. Furthermore, the number of points does not only depend on obstacle size but also on the distance from the camera. As an object moves away from the camera, its apparent size decreases and it is represented by fewer pixels in the image plane. On the other hand, as an object approaches the camera, it is less likely to fit completely within the camera FOV, especially in the case of larger objects such as aircraft. Nonetheless, it can still be assumed that very small clusters are due to noise. This criterion is assigned a weighting of $w_6 = 0.3$.

As in the case of the grouping criteria, soft thresholding is used and $C_{\overline{3d}}$, $C_{\overline{2d}}$ and C_{pts} are mapped onto score values between 0 and 1 as shown in Figure 5.13.[6] Then, each score value is multiplied by its corresponding weighting value and the overall score S is calculated as follows:

$$S = w_4 S_{\overline{3d}} + w_5 S_{\overline{2d}} + w_6 S_{pts} \tag{5.2.2}$$

where:

$S_{\overline{3d}}$, $S_{\overline{2d}}$ and S_{pts} are the score values associated with each criterion,

$w_4 = 0.4$, $w_5 = 0.3$ and $w_6 = 0.3$ are the weighting values associated with each criterion,

$w_4 + w_5 + w_6 = 1$.

S can vary between 0 and 1. If S is greater than a certain threshold $thres_2 = 0.7$, the cluster is assumed to be valid; otherwise, it is removed.

As shown in Section 5.1.2, the density of points in 3D space decreases with increasing distance from the cameras. This is mainly due to triangulation uncertainty which affects range resolution and positional accuracy. Hence, this factor needs to be taken into account when normalising C_{3d} and $C_{\overline{3d}}$. For this purpose, thresholds t_1,

[6]The values chosen for thresholds $t_7..t_{12}$ are presented at the end of this section.

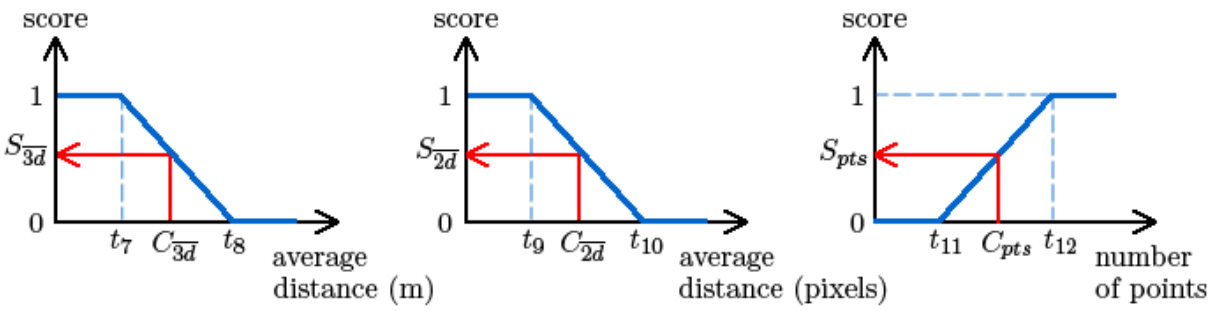


Figure 5.13: Mapping of filtering criteria onto score values

t_2, t_7 and t_8 increase linearly with distance as follows:

$$\begin{aligned} t_1 &= k_1 z \\ t_2 &= k_2 z \\ t_7 &= k_1 \bar{z} \\ t_8 &= k_2 \bar{z} \end{aligned} \tag{5.2.3}$$

where:

z is the z coordinate of the point (such as the *root*) whose *children* need to be determined,

$\bar{z}$ is the average of the z coordinates of all the points within a cluster,

k_1 and k_2 are positive constants.

5.2.3.4 Selection of thresholds used in the clustering algorithm

Suitable values for thresholds $t_1..t_{12}$ used in the clustering algorithm were determined experimentally by varying the threshold values and running the algorithm on different images. Figure 5.14 shows one of the test images used.

Some of the experimental results are shown in Figures 5.15 and 5.16. Figure 5.15 shows the results obtained for different values of k_1 and k_2 (which affect thresholds t_1, t_2, t_7 and t_8) whereas Figure 5.16 shows the results obtained for different values of t_3 and t_4. The clusters detected by the algorithm in each case are represented by different colours, both in the left intensity image as well as in the projection of the

(a) Left intensity image (b) Plan view

Figure 5.14: One of the test images used in experiments to determine suitable values for different clustering parameters

obstacle points in the ARF.[7] It is observed that, for small values of k_1 and k_2 (or t_3 and t_4), the clusters detected are both small and compact. Also, while the noise is kept to a minimum, certain sections of the aircraft (such as the wingtips) are not detected at all (as can be observed from Figures 5.15(a) and 5.16(a)). As a result, the shape of the aircraft is not well-defined in the ARF. On the other hand, as k_1 and k_2 (or t_3 and t_4) are increased, previously disconnected clusters are grouped together and, therefore, clusters become larger and less compact. The wingtips of the aircraft are detected and the aircraft's shape becomes more visible in the ARF (as can be observed, for instance, in Figures 5.15(b) and 5.16(c)). At the same time, however, fewer noisy obstacle points are removed. For instance, in Figure 5.15(d), the left wingtip of the aircraft and the terminal building in the background are connected together by a group of noisy points, thus forming a single cluster.

The thresholds' values were chosen as a balance between false detections and missed detections. A compromise was reached by selecting the following values: $k_1 = 0.05$, $k_2 = 0.06$, $t_3 = 20$ pixels and $t_4 = 30$ pixels.

Figure 5.17 shows how thresholds t_1 and t_2 (which are associated with grouping

[7]The protection zone is represented by a red rectangle around the left wingtip of the ownship in the ARF whereas the boundaries of the common FOV of the stereo vision system are represented by black dashed lines.

(a) $k_1 = 0.02$ and $k_2 = 0.03$

(b) $k_1 = 0.05$ and $k_2 = 0.06$

(c) $k_1 = 0.08$ and $k_2 = 0.09$

(d) $k_1 = 0.11$ and $k_2 = 0.12$

Figure 5.15: Clustering results for different values of k_1 and k_2

(a) $t_3 = 1$ pixel and $t_4 = 11$ pixels

(b) $t_3 = 10$ pixels and $t_4 = 20$ pixels

(c) $t_3 = 20$ pixels and $t_4 = 30$ pixels

(d) $t_3 = 30$ pixels and $t_4 = 40$ pixels

Figure 5.16: Clustering results for different values of t_3 and t_4

criterion C_{3d}) increase with distance from the cameras.[8] The variation of these thresholds with distance enables the clustering algorithm to reliably detect obstacles at different range values. The closer an obstacle is to the cameras, the more compact the corresponding cluster is expected to be. Therefore, by setting low values for the thresholds and keeping a narrow gap between them at close range, the clustering algorithm ensures that only the most compact clusters are retained. On the other hand, the larger the obstacle distance, the greater the expected separation between 3D points corresponding to the obstacle. Hence, by using higher values for the thresholds and widening the gap between them at long distances, the algorithm is able to detect distant obstacles.

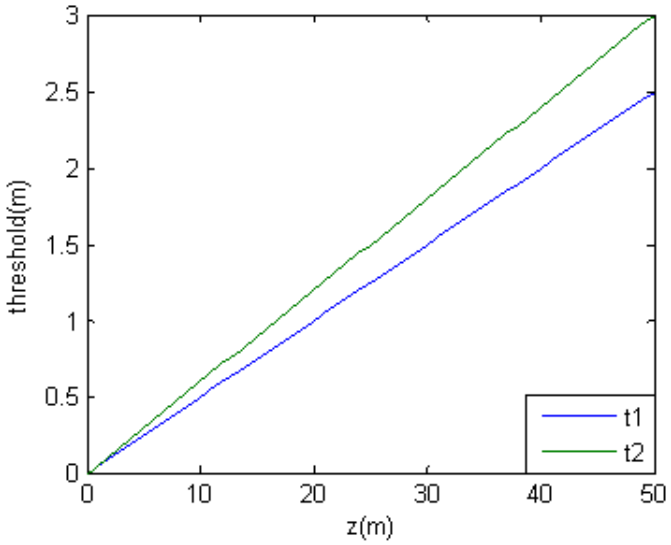


Figure 5.17: Variation of t_1 and t_2 with range

The values of all the weights and thresholds used in the clustering algorithm are given in Table 5.2.

[8]Thresholds t_7 and t_8 (which are associated with filtering criterion $C_{\overline{3d}}$) increase in a similar manner.

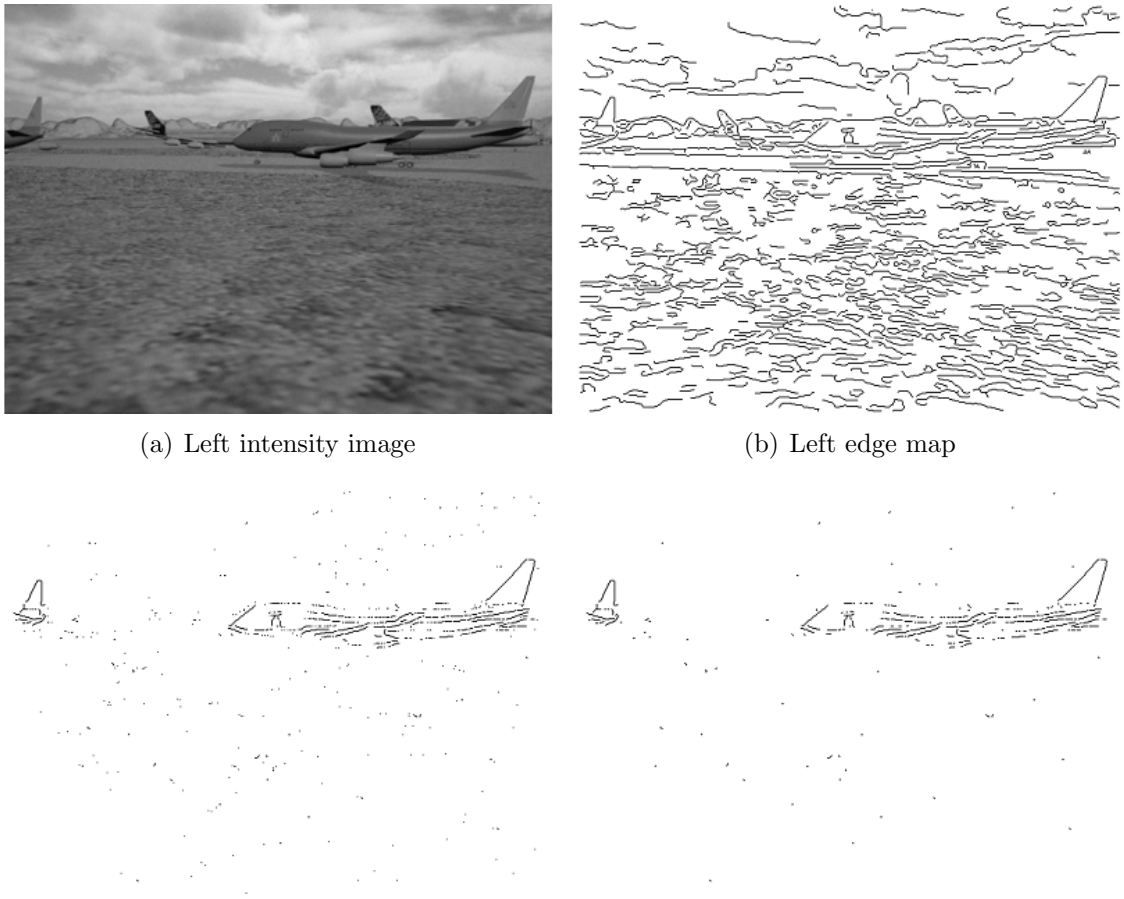
Table 5.2: Values of parameters used in the clustering algorithm

Grouping criteria									
t_1 (m)	t_2 (m)	t_3 (pixels)	t_4 (pixels)	t_5 (grey value)	t_6 (grey value)	w_1	w_2	w_3	$thres_1$
$0.05z$	$0.06z$	20	30	5	15	0.5	0.4	0.1	0.7
Filtering criteria									
t_7 (m)	t_8 (m)	t_9 (pixels)	t_{10} (pixels)	t_{11} (points)	t_{12} (points)	w_4	w_5	w_6	$thres_2$
$0.05\bar{z}$	$0.06\bar{z}$	$= t_3$	$= t_4$	20	50	0.4	0.3	0.3	0.7

5.2.4 Obstacle Detection and Clustering Results

Figures 5.18 and 5.19 show two examples of the results obtained for obstacle detection. As observed in Figures 5.18(b) and 5.19(b), edge detection extracts obstacle features (such as aircraft extremities and buildings) as well as ground features (such as grass, shadows and ground markings). Most of the ground features are successfully removed by height thresholding, resulting in the obstacle maps shown in Figures 5.18(c) and 5.19(c). Then, isolated obstacle pixels (most of which correspond to either ground or sky features) are removed from the obstacle maps as shown in Figures 5.18(d) and 5.19(d).

Figures 5.20 and 5.21 show two examples of the results of clustering. To be able to assess the effectiveness of this process, the obstacle points are superimposed on the intensity image and plotted in the ARF before and after clustering is carried out. In the example shown in Figure 5.20, the ownship is approaching an aircraft which is of a similar size to it. From Figure 5.20(c), it is apparent that obstacle detection based on height thresholding and the removal of isolated obstacle points is not sufficient to eliminate all of the 'noisy' points, that is, ground features and 3D points that are incorrectly mapped onto the WRF. A lot of noisy points are still present and some of them are actually inside the protection zone. Figure 5.20(d) shows the significant improvement obtained after clustering (The clusters are represented by different colours in Figures 5.20(b), 5.20(d) and 5.20(f)). Most of the noisy points, including those that were previously detected in the protection zone, are removed. At



(a) Left intensity image

(b) Left edge map

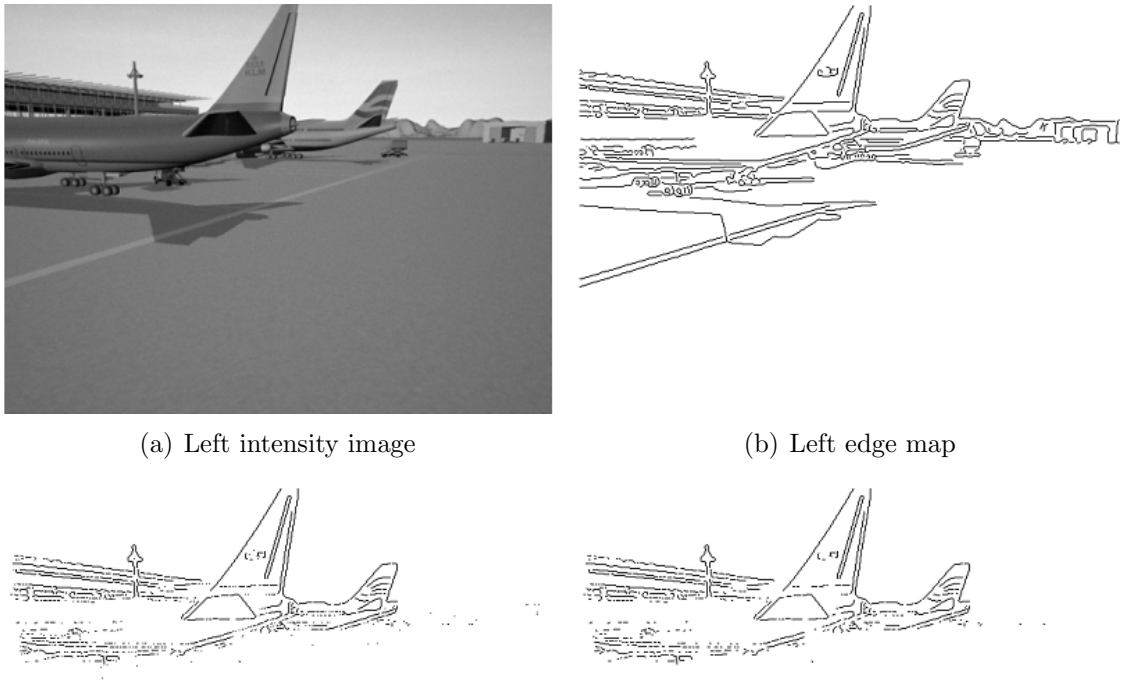
(c) Noisy obstacle map

(d) Filtered obstacle map

Figure 5.18: Obstacle detection without clustering (Example 1)

the same time, however, the points corresponding to the aircraft and to the hangar behind it, are preserved. The shape and orientation of the aircraft is clearly visible in Figure 5.20(d) and, by comparing Figures 5.20(e) and 5.20(f), it is observed that the obstacle points corresponding to the aircraft (represented by the light blue, dark blue and pink clusters) match very well with the ground truth data. It is also observed that, since the hangar is further away from the ownship, the points corresponding to it (represented by the green and red clusters) are more dispersed than the points corresponding to the aircraft. As explained previously, the clustering algorithm is able to detect such distant obstacles by adjusting its thresholds dynamically.

Figure 5.21 shows the ownship taxiing on the ramp. Different kinds of obstacles



(a) Left intensity image

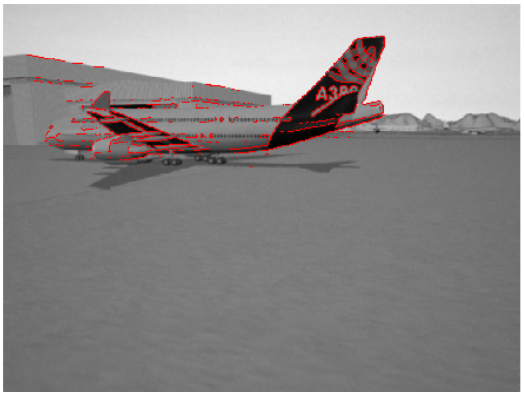
(b) Left edge map

(c) Noisy obstacle map

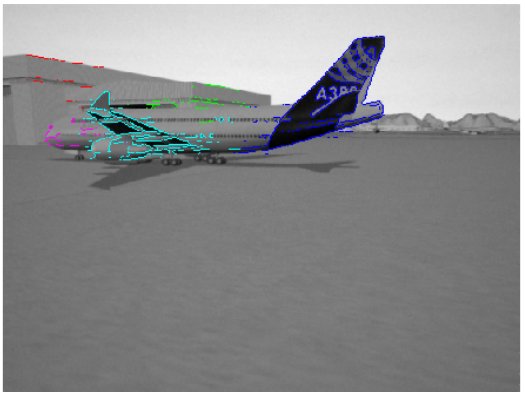
(d) Filtered obstacle map

Figure 5.19: Obstacle detection without clustering (Example 2)

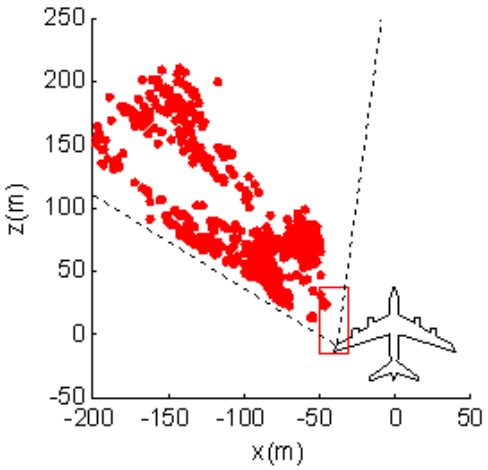
are visible in the scene, including aircraft, light poles, catering trucks and an airport coach. The coach penetrates the ownship protection zone as can be observed in Figure 5.21(e). As in the first example, Figure 5.21(d) shows the effectiveness of clustering in removing noise while retaining true obstacles. In fact, the coach is correctly detected inside the protection zone and the shape, position and orientation of the pink cluster representing it in Figure 5.21(f) accurately matches the ground truth data provided in Figure 5.21(e).



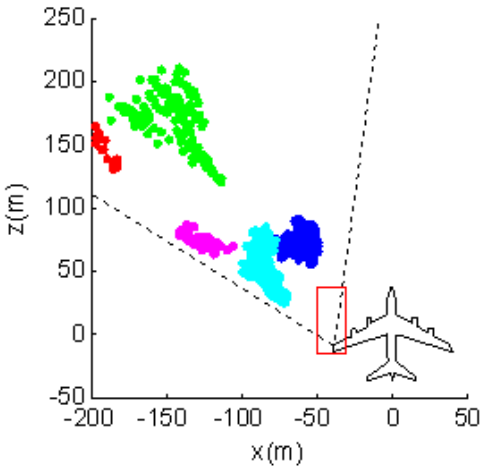
(a) Obstacle points superimposed on intensity image before clustering



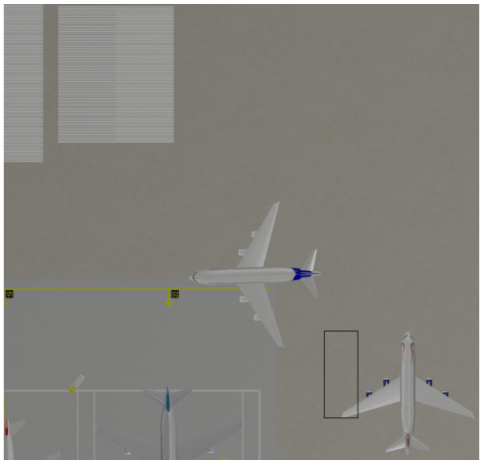
(b) Obstacle points superimposed on intensity image after clustering



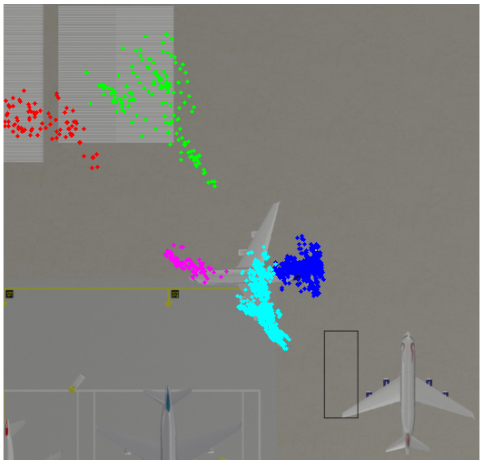
(c) Obstacle points in ARF before clustering



(d) Obstacle points in ARF after clustering



(e) Plan view

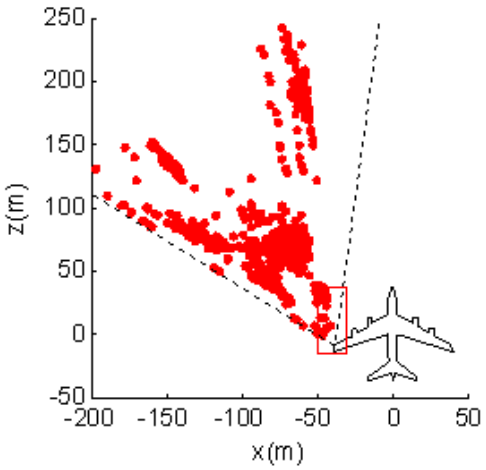


(f) Obstacle points superimposed on plan view after clustering

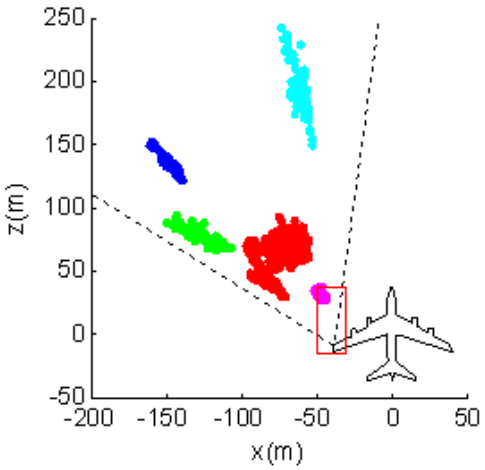
Figure 5.20: Clustering (Example 1)

(a) Obstacle points superimposed on intensity image before clustering

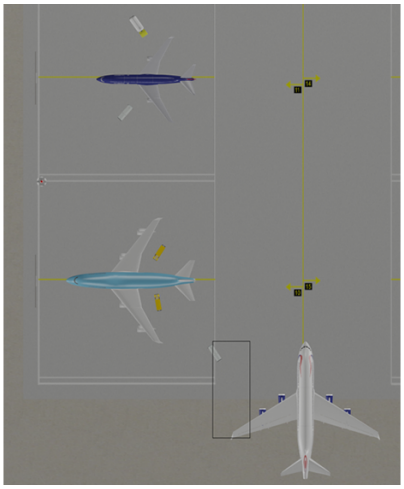
(b) Obstacle points superimposed on intensity image after clustering



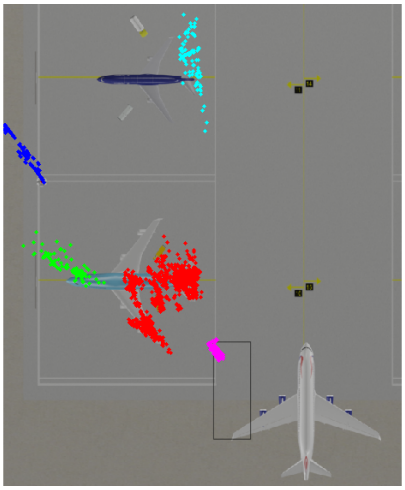
(c) Obstacle points in ARF before clustering



(d) obstacle points in ARF after clustering



(e) Plan view



(f) Obstacle points superimposed on plan view after clustering

Figure 5.21: Clustering (Example 2)

Chapter 6

Obstacle Tracking

In the chapters presented so far, the focus has been on the spatial content of images. However, a lot of information can also be obtained from the temporal content of an image sequence, and this gives rise to the concept of obstacle tracking. Section 6.1 gives an overview of visual tracking. It describes the challenges associated with this process, discusses some of the techniques available for state estimation, and outlines the benefits of tracking for this application. Section 6.2 discusses the design of the Kalman filter that was implemented in this research and explains how it is tuned. The selection of obstacles for tracking is addressed in Section 6.3 together with a discussion on how the system copes with false detections and missed detections. Some tracking results are presented and discussed in Section 6.4.

6.1 Overview of Visual Tracking

6.1.1 Categories of Visual Tracking

Visual tracking techniques can be broadly classified into four main categories. These differ in the type of image primitives and object characteristics used for tracking:

- **Region-based:** This approach tracks regions of connected pixels (or 'blobs') corresponding to individual objects. These image regions are detected using segmentation (as discussed in Section 2.1.2) or clustering techniques (as discussed in Section 5.2.3). If objects are situated too close to each other, they

can be detected as a single region in the image. Properties of each region, such as shape, colour, position and motion, are tracked along the image sequence.

A system that uses region-based tracking is proposed in [93]. This is a traffic monitoring system that uses a single static camera to track multiple vehicles. Image regions corresponding to vehicles are detected by motion segmentation. Regions detected in the current frame are matched to those detected in the previous frame on the basis of position, motion and intensity information.

- **Model-based:** With this approach, 2D or 3D geometric object models are tracked. For example, in the monocular traffic surveillance application discussed in [94], a range of 3D wire frame models are used to detect, classify and track multiple vehicles, such as buses, cars and motorcycles. First, using background estimation, motion silhouettes (corresponding to vehicles) are extracted. Then, using knowledge of the camera calibration parameters, each 3D model is placed on a grid of positions on the ground plane (corresponding to the road surface) and projected onto the image plane. Classification is carried out by determining which of the 3D models results in the maximum area of overlap between the motion silhouettes and the image projection of the 3D models. Once a silhouette is classified into one of the categories that are catered for by the system, its 3D model is tracked in subsequent frames.

  In another automotive application described in [31], a single vehicle-mounted camera is used to detect and track vehicles. Each vehicle region in an image is enclosed by a rectangular bounding box which is detected using edge information. Then, each of these 2D bounding boxes is tracked. Several vehicle parameters, such as 3D position and attitude, are estimated by projecting a simplified 3D vehicle model (a 3D bounding box) onto the image plane and then measuring the difference between the bounding boxes corresponding to the detected vehicle and the projected vehicle model.

- **Active-contour-based:** With this approach, a model is obtained for the

contour of an object and this model is updated dynamically. This method can be applied both to rigid and non-rigid objects. For instance, in [95], a stationary camera is mounted above the road. First, image regions corresponding to moving vehicles are detected using background subtraction. Then, edge detection is carried out on each of these regions and a contour is extracted which encloses the edge pixels. The contour is approximated by closed cubic splines. The motion and shape characteristics of each contour are estimated in each frame and are used to define a search area (mask) for the corresponding contour in subsequent frames.

- **Feature-based:** The previous three tracking approaches attempt to track whole objects and assume that these objects are completely within the camera FOV. On the other hand, with feature-based tracking, specific image features are tracked individually. For example, in the UAV application described in [96], corner features are detected and tracked in a monocular image sequence. Pairs of corresponding corners are found in consecutive frames using correlation techniques. Each of these corner pairs is mapped onto a 3D point. The 3D points are then grouped into clusters, each of which is surrounded by a bounding cylinder in order to be avoided by the UAV.

  In a completely different application using a single camera [97], corner features are detected and tracked for the purpose of image stabilisation. The motion information of each corner is used to estimate the image warping that is required to have a stabilised image.

For the application discussed in this work, the first three tracking approaches are not very suitable. The use of model-based tracking would require a large number of accurate geometric models to be able to detect and classify each of the several types of obstacles that are likely to be present in aerodrome areas. Moreover, as mentioned already, the first three approaches assume that the obstacles are entirely within the camera FOV. This assumption does not hold for this application due to the relatively large size of obstacles in aerodrome areas (especially aircraft). Therefore,

the approach adopted in this work is feature-based tracking, where the system tracks the closest edge feature (corresponding to an obstacle) in each frame. This approach is discussed in more detail in Section 6.3.

6.1.2 Data Association

One of the challenges of visual tracking, particularly when tracking multiple objects, is *data association*. This is the problem of deciding which of the objects detected in the previous frame corresponds to each of the objects detected in the current frame. In most applications, corresponding objects are found by applying some neighborhood criterion. This approach works well when the objects are isolated from each other. However, it can fail when objects get too close to each other and, particularly, when their trajectories intersect. In this case, objects can occlude each other and can be interpreted as a single object. As a result, it will not be possible to find a one-to-one correspondence between objects in consecutive frames.[1] Tracking algorithms must therefore be designed to handle partial or total occlusions.

In [95], vehicle motion is assumed to be constrained to the ground plane. This means that the farther away a vehicle is from the camera, the higher its position in the image. In order to detect occlusions, the proposed system sorts the tracked vehicle contours by their predicted image position in the next frame. Occlusions occur wherever two contours overlap. The contour which appears lower in the image corresponds to a vehicle that is closer to the camera. Therefore, it is considered to be occluding the other contour. The overlap between contours is taken into consideration when defining the search area used to estimate the contour of an object in the next frame.

A different method of handling occlusions is proposed in [98], where a static camera tracks multiple vehicles. As in [95], vehicles are assumed to move on a known ground plane. A single blob containing multiple merged vehicles is segmented into individual vehicles by determining the probability of different multi-vehicle configurations. For

[1] Due to their attempt to track whole objects, the first three tracking approaches discussed in Section 6.1.1 are more likely to be affected by occlusions than feature-based tracking.

each configuration, 3D bounding boxes (corresponding to a certain arrangement of vehicles on the road) are projected onto the image plane. The likelihood of this configuration is then found by checking the overlap between its synthesised image mask and the detected blob. The segmentation process is based on Markov chain Monte Carlo (MCMC) techniques.

6.1.3 State Estimation Techniques

The main objective of tracking is to monitor the state of a system. Normally, it is not possible to directly measure all of the state variables. Furthermore, the measurements taken during the tracking process are often corrupted by noise. Several recursive Bayesian techniques are available to obtain the best estimate of the state of a system based on noisy observations. These mathematical tools determine the posterior probability density function of the state by using knowledge of system and measurement dynamics, statistics of system noise and measurement errors, and initial condition information.

One of the Bayesian estimators that is most commonly used by the computer vision community is the Kalman filter [37, 54, 94, 95]. This is the optimal estimator for a linear system with Gaussian (unimodal) random variables. A variation of the Kalman filter which can be applied to nonlinear systems is the Extended Kalman Filter (EKF). This works by linearising the system around the current state, through a first order Taylor expansion of the functions. An example of an application using the EKF can be found in [31]. When the system is highly nonlinear, the Unscented Kalman Filter (UKF) is used.

The EKF and UKF are still limited to systems with unimodal random variables. The Kalman filter can be adapted to handle multimodal distributions by using Multi-Hypothesis Tracking (MHT). With this approach, the probability density function is represented by a mixture of Gaussians, where each hypothesis is tracked by a separate Kalman filter. MHT is particularly suitable when tracking multiple targets in a cluttered environment. The processing time complexity of Kalman filters

increases polynomially with state size.

When the state space is discrete and consists of a finite number of states, the grid-based filter is used. The state space is represented by a grid and the probability of each state is computed by processing the individual 'cells'. This filter can represent arbitrary distributions and can be applied to nonlinear systems. When the state space is continuous, an approximate grid-based method can be applied by discretising the state space. In this case, filter performance depends on grid density. The processing cost of the grid-based filter increase exponentially with state size. It is therefore limited to low-dimensional estimation problems.

Another Bayesian estimator is the particle filter. This can deal with multimodal probability distributions and can be applied to nonlinear systems. The particle filter is a sequential Monte Carlo method that represents probability density functions by a set of weighted state samples called *particles*. One of the ways of initialising the particles is by distributing them uniformly over the state space. The distribution and weights of the particles are then updated at each time step. One of the most common particle filter algorithms is Sequential Importance Resampling (SIR) [99]. In this algorithm, a new set of particles is created at each time step by sampling the existing particles and removing particles with small weights while multiplying particles with large weights. One of the particle filter's advantages over the grid-based approach is that it concentrates the particles into regions of the state space with a high probability density. Filter complexity increases exponentially with state size and the performance depends on several factors, including: number of particles, sampling method, and initialisation procedure.

A comprehensive review of Bayesian estimators can be found at [100].

As shown in Section 6.2.1, the process and measurement dynamics of the system designed in this work are linear. Also, it is assumed that the process and measurement variables are corrupted by Gaussian noise. For these reasons, the Kalman filter was selected as the state estimator for obstacle tracking.

6.1.4 Benefits of Obstacle Tracking for this Application

In the context of this application, the main benefits of obstacle tracking are the following:

- Better position estimates - The Kalman filter uses a weighted combination of predictions and noisy measurements in order to obtain better estimates of obstacle positions.
- Closure rate estimates - The relative closure rate between the ownship and an obstacle is not measured directly but is estimated from distance measurements.
- Tracking of missing obstacles - If an obstacle goes missing in a certain number of frames, it can still be tracked by relying on Kalman filter predictions.
- Outlier detection - 'True' obstacles normally appear in multiple consecutive frames (this is known as *temporal consistency*) whereas 'noisy' obstacles are likely to appear intermittently.
- Monitoring of obstacle trajectories - This is useful to detect potential conflicts and collisions.

6.2 Kalman Filter Design

6.2.1 System and Measurement Models

When using the Kalman filter, the system and measurement dynamics are modeled using state-space equations [74]. The Kalman filter is updated at equally spaced time instants as follows:

$$t_k = t_0 + k\Delta T \tag{6.2.1}$$

where:

t is the time,

k is a positive integer,

ΔT is the sampling time interval.

In the simulations used for this research, an update rate of 15Hz is used;[2] therefore, $\Delta T = 1/15 \simeq 0.067s$.

The system model is expressed by the following equation:

$$\mathbf{x}_k = \phi_{k-1}\mathbf{x}_{k-1} + \xi_{k-1} \tag{6.2.2}$$

where:

$\mathbf{x}$ is the state vector of an obstacle point,

ϕ is the state transition matrix (which is assumed to remain constant for this application),

ξ is a vector modeling system noise.

The state vector is defined as $\mathbf{x} = (p_1, p_2, v_1, v_2)$ where (p_1, p_2) are the coordinates of the obstacle point in the xz plane of the ARF while (v_1, v_2) are the velocity components in the xz plane. Assuming that the speed of the obstacle is constant over the time interval ΔT, Equation (6.2.2) can be expressed in matrix form as follows:

$$\begin{aligned} \mathbf{x}_k &= \phi_{k-1}\mathbf{x}_{k-1} + \xi_{k-1} \\ \begin{pmatrix} p_{1,k} \\ p_{2,k} \\ v_{1,k} \\ v_{2,k} \end{pmatrix} &= \begin{pmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p_{1,k-1} \\ p_{2,k-1} \\ v_{1,k-1} \\ v_{2,k-1} \end{pmatrix} + \begin{pmatrix} \xi_{p_1,k-1} \\ \xi_{p_2,k-1} \\ \xi_{v_1,k-1} \\ \xi_{v_2,k-1} \end{pmatrix} \end{aligned} \tag{6.2.3}$$

The measurement model is expressed by the following equation:

$$\mathbf{z}_k = H_k\mathbf{x}_k + \mu_k \tag{6.2.4}$$

where:

$\mathbf{z}$ is the measurement vector,

H is the measurement matrix,

μ is a vector modeling measurement noise.

The stereo vision system can only measure the position of an obstacle. Hence, the measurement vector is defined as $\mathbf{z} = (z_1, z_2)$ where z_1 and z_2 are the measured

[2]Most stereo vision-based obstacle detection systems in the literature have update rates in the range of 10 to 25Hz [3, 59, 79, 101].

coordinates of an obstacle point in the xz plane of the ARF. Equation (6.2.4) can be expressed in matrix form as follows:

$$\begin{aligned} \mathbf{z}_k &= H_k \mathbf{x}_k + \mu_k \\ \begin{pmatrix} z_{1,k} \\ z_{2,k} \end{pmatrix} &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} p_{1,k} \\ p_{2,k} \\ v_{1,k} \\ v_{2,k} \end{pmatrix} + \begin{pmatrix} \mu_{z_1,k} \\ \mu_{z_2,k} \end{pmatrix} \end{aligned} \tag{6.2.5}$$

The noise terms ξ and μ are considered to be uncorrelated and are assumed to be AWGN processes, with covariance matrices Q and R respectively:

$$Q = \begin{pmatrix} {\sigma_{p_1}}^2 & \sigma_{p_1 p_2} & \sigma_{p_1 v_1} & \sigma_{p_1 v_2} \\ \sigma_{p_2 p_1} & {\sigma_{p_2}}^2 & \sigma_{p_2 v_1} & \sigma_{p_2 v_2} \\ \sigma_{v_1 p_1} & \sigma_{v_1 p_2} & {\sigma_{v_1}}^2 & \sigma_{v_1 v_2} \\ \sigma_{v_2 p_1} & \sigma_{v_2 p_2} & \sigma_{v_2 v_1} & {\sigma_{v_2}}^2 \end{pmatrix}$$

where the diagonal entries of Q are the variances of the elements of state vector $\mathbf{x}$.

$$R = \begin{pmatrix} {\sigma_{z_1}}^2 & \sigma_{z_1 z_2} \\ \sigma_{z_2 z_1} & {\sigma_{z_2}}^2 \end{pmatrix}$$

where the diagonal entries of R are the variances of the elements of measurement vector $\mathbf{z}$.

6.2.2 Kalman Filter Equations

Like all recursive Bayesian estimators, the Kalman filter consists of two stages: a prediction step and an update (innovation) step. The prediction stage consists of the following two standard equations:

$$\hat{\mathbf{x}}'_k = \phi_{k-1} \hat{\mathbf{x}}_{k-1} \tag{6.2.6}$$

$$P'_k = \phi_{k-1} P_{k-1} \phi_{k-1}^T + Q_{k-1} \tag{6.2.7}$$

where:

$\hat{\mathbf{x}}'_k$ is the prediction of the k-th state estimate,

$\hat{\mathbf{x}}_{k-1}$ is the state estimate at time t_{k-1},
P'_k is the predicted covariance matrix of the k-th state estimate,
P_{k-1} is the covariance matrix of the state at time t_{k-1}.

As observed from Equations (6.2.6) and (6.2.7), the prediction stage forward-projects the state and its covariance at time t_{k-1} in order to obtain a priori estimates (predictions) of the state and its covariance at the current time step t_k.

The update stage provides feedback to the system. It incorporates the measurements into the a priori estimates in order to obtain better a posteriori estimates. The update stage consists of three standard equations:

$$K_k = P'_k H_k^T (H_k P'_k H_k^T + R_k)^{-1} \tag{6.2.8}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}'_k + K_k(\mathbf{z}_k - H_k \hat{\mathbf{x}}'_k) \tag{6.2.9}$$

$$P_k = (I - K_k)P'_k(I - K_k)^T + K_k R_k K_k^T \tag{6.2.10}$$

where:
$\hat{\mathbf{x}}_k$ is the k-th state estimate,
K_k is the k-th gain matrix,
P_k is the covariance matrix of the k-th state estimate (computed after the measurement is obtained).

The gain matrix K determines the relative importance of the measurements and the predictions. A large value of K gives more importance to the measurements whereas a small value of K gives more weighting to the predictions.

The uncertainty of state estimate $\hat{\mathbf{x}}_k$ is determined by covariance matrix P_k. When a new obstacle is tracked, $\hat{\mathbf{x}}_k$ is initialised to $\hat{\mathbf{x}}_k = (z_{1,k}, z_{2,k}, 0, 0)$ and P_k is initialised to the following diagonal matrix:

$$P_k = \begin{pmatrix} 1 \times 10^4 m^2 & 0 & 0 & 0 \\ 0 & 1 \times 10^4 m^2 & 0 & 0 \\ 0 & 0 & 1 \times 10^4 m^2/s^2 & 0 \\ 0 & 0 & 0 & 1 \times 10^4 m^2/s^2 \end{pmatrix}$$

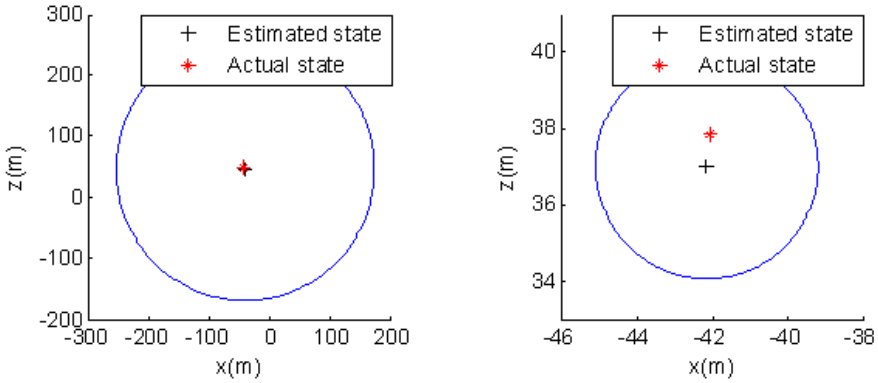
The diagonal entries of P_k are set to arbitrarily large values due to the initial state uncertainty. For a 2D state vector (such as the position of an obstacle in this

application), this uncertainty can be represented by an ellipse centered around $\hat{\mathbf{x}}_k$. The axes of this ellipse are given by $\pm c\sqrt{\lambda_i}e_i$ $(i = 1, 2)$, where λ_i and e_i are the eigenvalues and eigenvectors, respectively, of P_k. For a 90% probability that the state is inside the ellipse, $c = 2.146$. Initially, the uncertainty ellipse will be large; however, as more measurements are obtained, the Kalman filter should converge and the uncertainty should decrease.

Figure 6.1 shows the uncertainty ellipse associated with obstacle position at different stages of a particular tracking sequence. In this example, the system tracks the horizontal stabiliser of an aircraft situated in front of it. The centre of each ellipse represents the estimated position of the tracked obstacle and each ellipse encloses an area which has a 90% probability of containing the actual position of the obstacle. As expected, the magnitude of the state uncertainty (represented by the size of the ellipses) decreases during the tracking sequence. It can also be observed that the ellipses are very similar to circles, implying that there is very little difference between the uncertainties associated with the x and z coordinates of the position of the tracked obstacle.

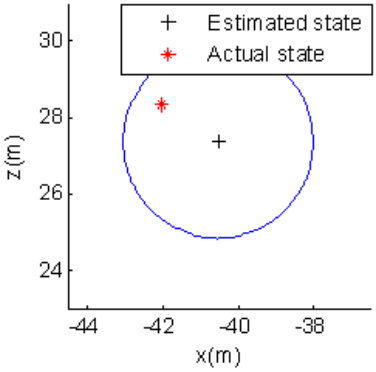
6.2.3 Kalman Filter Tuning

One of the factors that affect the performance of the Kalman filter is the selection of the noise covariance matrices Q and R. If the process noise covariance matrix Q is set to a value that is larger than the actual process noise, the Kalman filter will tend to rely more on the measurements. It will have a higher bandwidth but the estimation error will be larger when noisy data is encountered. On the other hand, if the value of Q is lower than the actual process noise, the Kalman filter will rely more on the predictions. It will have a lower bandwidth but the estimation error will be smaller. The opposite effects are observed when setting the measurement noise covariance matrix R. If the value of R is larger than the actual measurement noise, the Kalman filter will rely more on the predictions. Conversely, if the value of R is smaller than the actual measurement noise, the Kalman filter will rely more on the

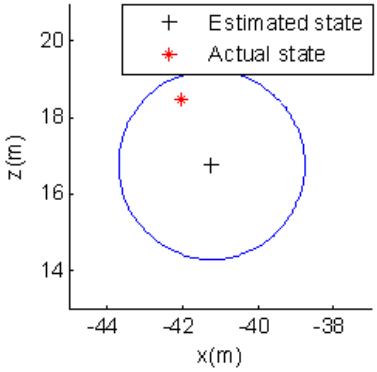


(a) Frame 1

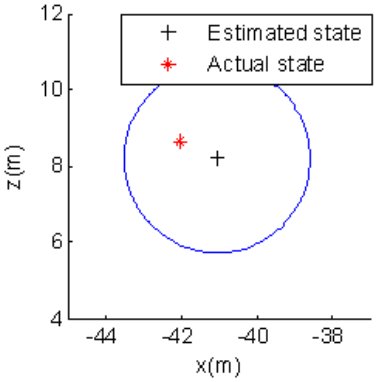
(b) Frame 20



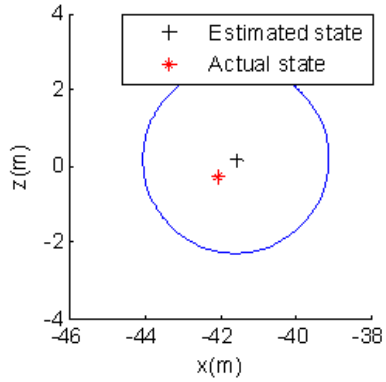
(c) Frame 40



(d) Frame 60



(e) Frame 80



(f) Frame 100

Figure 6.1: The effect of Kalman filter convergence on state uncertainty: The figure shows the uncertainty ellipse (90% confidence level), with the estimated position at the centre and the actual position of the obstacle. The uncertainty drops significantly as the number of frames increases.

measurements. Therefore, if incorrect values are selected for R and Q, the Kalman filter will be sub-optimal.

6.2.3.1 Tuning of the process noise covariance matrix

In this application, the process noise can be assumed to be constant. A suitable value for Q was found by trial and error, by analysing the performance of the Kalman filter for different values of the diagonal entries of Q (${\sigma_{p_1}}^2$, ${\sigma_{p_2}}^2$, ${\sigma_{v_1}}^2$ and ${\sigma_{v_2}}^2$). The noise characteristics of the individual elements of the state vector were assumed to be uncorrelated and, therefore, the off-diagonal entries of Q were set to 0 in each case. For each value of Q, the Kalman filter was tested by means of the scenario presented in Figure 6.2. The ownship is located on a taxiway and a stationary aircraft is located on a parallel taxiway, ahead of the ownship. Initially, the ownship is stationary. Then, after a number of frames, it starts moving at a speed of 15kts and continues doing so for some time, after which it stops once again and remains stationary until the end of the scenario. Throughout the scenario, the system tracks the right wingtip of the stationary aircraft.

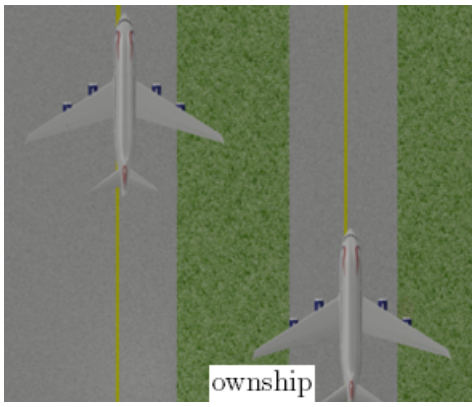


Figure 6.2: Plan view of scenario used to test the Kalman filter with different values of Q

Figure 6.3 shows the tracking results obtained for different values of Q. From Figure 6.3(a) it can be observed that, as expected, when the magnitude of Q is small, the Kalman filter is slow to react to changes in the distance and closure rate profiles because it relies more on its predictions than on the measurements. For example, when the ownship stops moving (in Frame 218), the estimated distance between the

ownship and the tracked obstacle continues to decrease. On the other hand, when the magnitude of Q is large (Figure 6.3(b)), the filter puts less confidence in the model and relies more on the measurements. As a result, there is almost no lag between the estimated and actual distance and closure rate profiles. However, the estimates obtained are very noisy.

As a compromise between filter bandwidth and estimation accuracy, the following values were chosen for the standard deviation of the process noise associated with each of the elements of the state vector: $\sigma_{p_1} = \sigma_{p_2} = 0.3m$ and $\sigma_{v_1} = \sigma_{v_2} = 0.3m/s$. The distance and closure rate estimates obtained with these values are shown in Figure 6.3(c).

6.2.3.2 Tuning of the measurement noise covariance matrix

Contrary to process noise, measurement noise cannot be assumed to be constant. The measurement error changes with lighting conditions and with distance of obstacles from the cameras. Therefore, it is necessary to have a method of tuning R online. The method chosen for this research is the one proposed in [102,103] and is described in the rest of this section.

The true state $\mathbf{x}_k$ is unknown; therefore, μ_k cannot be determined. However, an approximation for μ_k can be obtained from Equation (6.2.4):

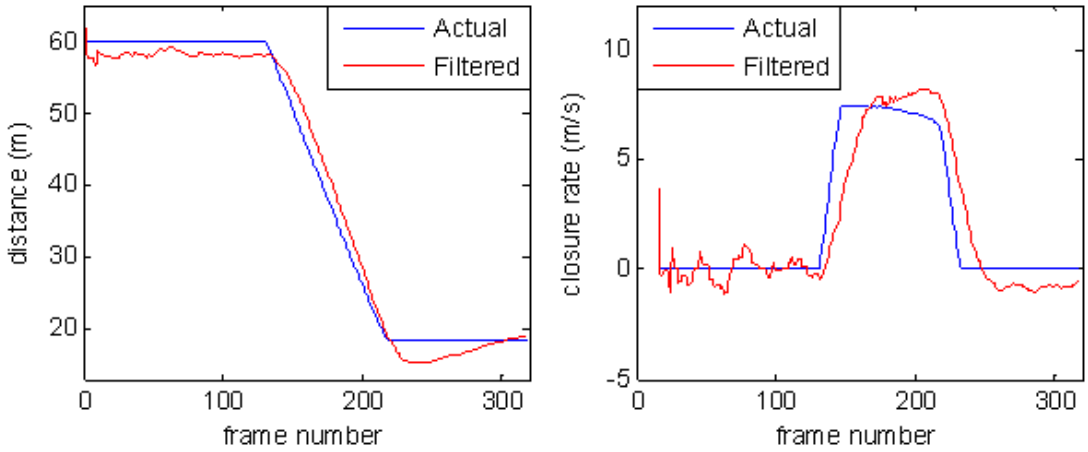
$$\mathbf{r}_j = \mathbf{z}_j - H_j\hat{\mathbf{x}}'_j \tag{6.2.11}$$

where:
$H_j\hat{\mathbf{x}}'_j$ is the 'predicted' measurement,
$\mathbf{r}_j$ is the observation noise sample,
j is a positive integer.

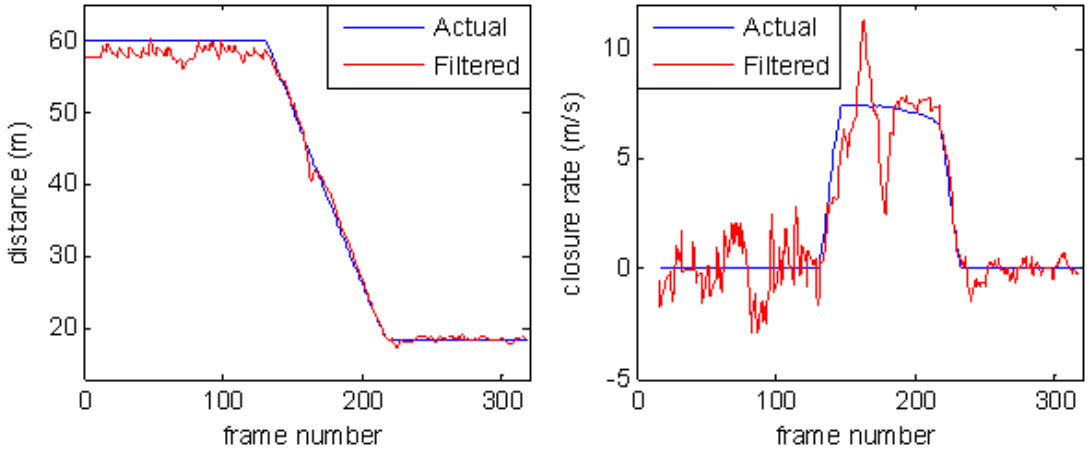
If N consecutive observation noise samples are obtained, an unbiased estimate of R is given by:

$$\hat{R} = \frac{1}{N-1}\sum_{j=1}^{N}(\mathbf{r}_j - \hat{r})(\mathbf{r}_j - \hat{r})^T - \left(\frac{N-1}{N}\right)H_j P'_j H_j^T \tag{6.2.12}$$

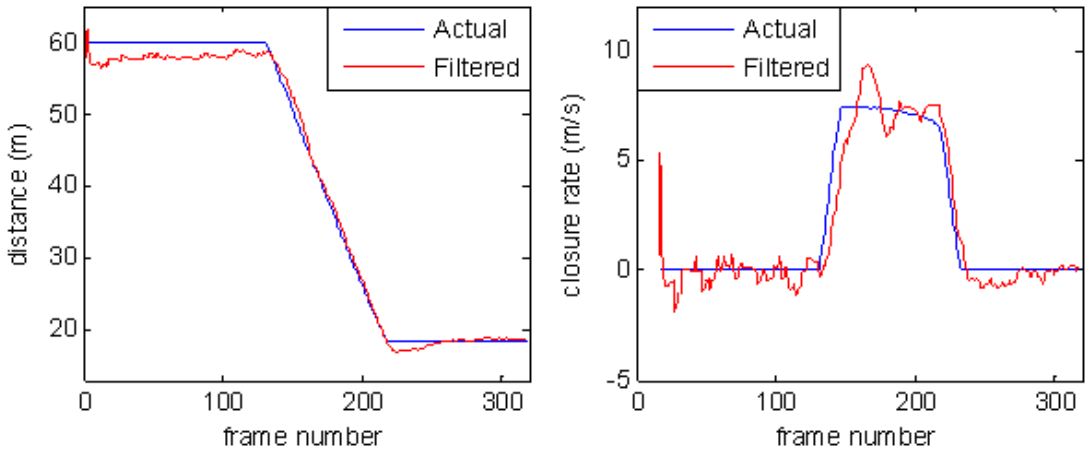
where $\hat{r}$ is the average of the N observation noise samples.



(a) Distance and closure rate estimates when $\sigma_{p_1} = \sigma_{p_2} = 0.1m, \sigma_{v_1} = \sigma_{v_2} = 0.1m/s$



(b) Distance and closure rate estimates when $\sigma_{p_1} = \sigma_{p_2} = 1m, \sigma_{v_1} = \sigma_{v_2} = 1m/s$

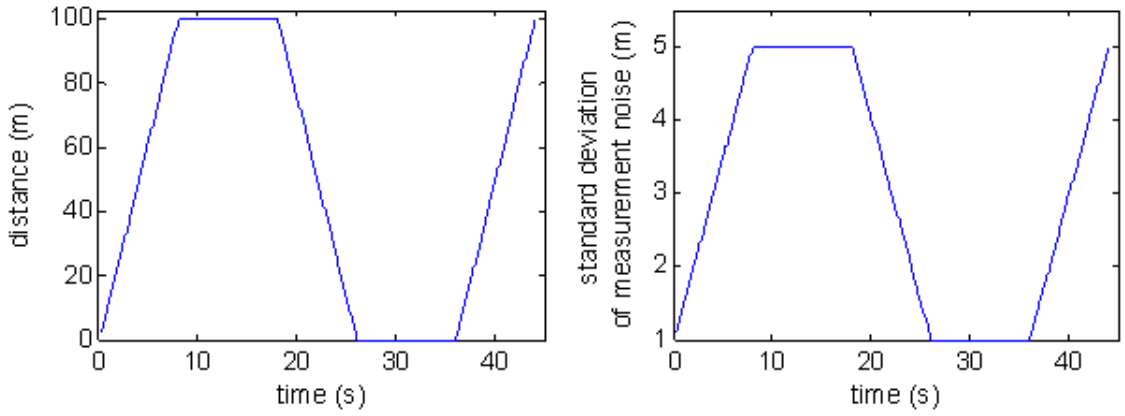


(c) Distance and closure rate estimates when $\sigma_{p_1} = \sigma_{p_2} = 0.3m, \sigma_{v_1} = \sigma_{v_2} = 0.3m/s$

Figure 6.3: Distance and closure rate estimates obtained for different values of Q

R is estimated using a sliding window. The first estimate is obtained after N frames. Subsequently, R is estimated in every frame by considering the N most recent observation noise samples. It is assumed that the N noise samples are statistically independent and that the noise statistical parameters remain constant over N sample times.

To test the effectiveness of the method of online measurement noise estimation and to choose a suitable value for N, a tracking scenario was simulated where the distance between the ownship's left wingtip and an obstacle varied between 0 and 100m according to the profile shown in Figure 6.4(a). The maximum closure rate between the ownship and the obstacle was about 25kts. In order to have ground truth data of the measurement noise, the measurements (z_1 and z_2) were not obtained by using the stereo vision system but by adding Gaussian noise to the actual position of the obstacle in each frame of the scenario. The standard deviation of the measurement noise (σ_{z_1} and σ_{z_2}) varied linearly between 1m and 5m with distance from the ownship, as shown in Figure 6.4(b).



(a) Distance profile (b) Measurement noise profile

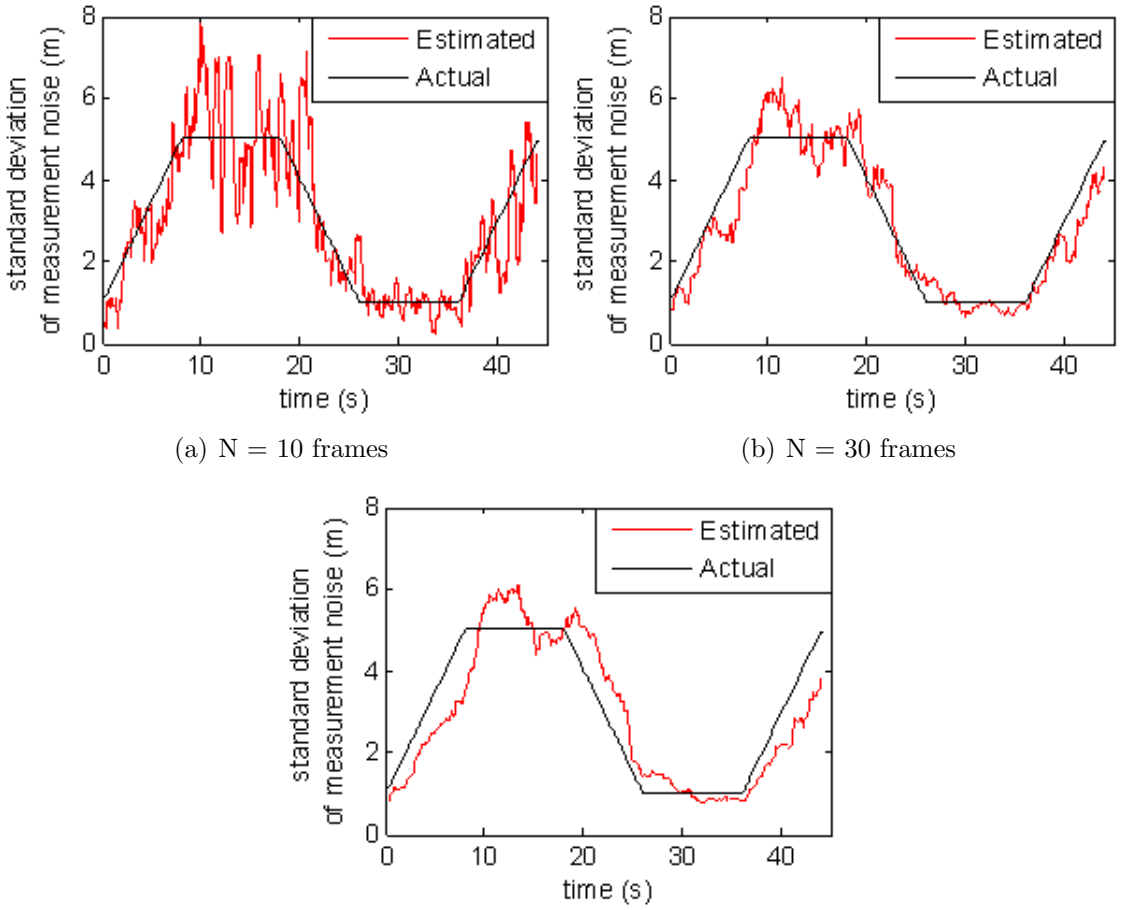
Figure 6.4: Distance and measurement noise profiles used to test the online measurement noise estimation algorithm and to choose a suitable value for window size N

The tracking scenario was repeated for different values of window size N. The estimations obtained for σ_{z_2} (the standard deviation of the measurement noise

associated with the z coordinate of the position of the obstacle) are presented in Figure 6.5 for three particular values of N. It can be observed that, as N is increased, the error in the estimation of the measurement noise decreases. This is because the estimation process improves with a larger number of samples. However, the larger the value of N, the greater the delay in the noise estimation. This can be clearly observed in Figure 6.5(c) where the estimated noise profile lags the actual noise profile. The reason this happens is that, the wider the sliding window, the greater the possibility that the noise characteristics are not uniform over the whole window. Therefore, whenever there is a change in the noise characteristics, the Kalman filter is slower to react. As a trade-off between good noise estimation and adaptability to changes in noise characteristics, it was decided to choose a window size of $N = 30$ frames (Figure 6.5(b)).

The online measurement noise estimation algorithm was tested on another tracking scenario, with the difference that the measurements of obstacle position were obtained using the stereo vision system. The tracking scenario was the same as the one used for the selection of the process noise covariance matrix (Refer to Figure 6.2). In order to demonstrate the benefit of online measurement noise estimation, this tracking sequence was repeated for three test cases. In the first two cases, the diagonal entries of matrix R, ${\sigma_{z_1}}^2$ and ${\sigma_{z_2}}^2$, were set to static values. It was assumed that the measurement noise associated with z_1 was independent from the measurement noise associated with z_2. Therefore, the off-diagonal entries of R were set to 0. In the first test case, σ_{z_1} and σ_{z_2} were set to a value of 0.1m (which was smaller than the actual value) whereas, in the second case, they were set to a value of 10m (which was larger than the actual value). In the third test case, the standard deviation of the measurement noise was estimated online.

Figure 6.6 shows the tracking results obtained for the different test cases. From Figure 6.6(a) it can be observed that, when R is set to a value that is lower than the actual measurement noise, the Kalman filter gives the measurements greater importance. As a result, the distance and closure rate estimates are very noisy



(a) N = 10 frames

(b) N = 30 frames

(c) N = 60 frames

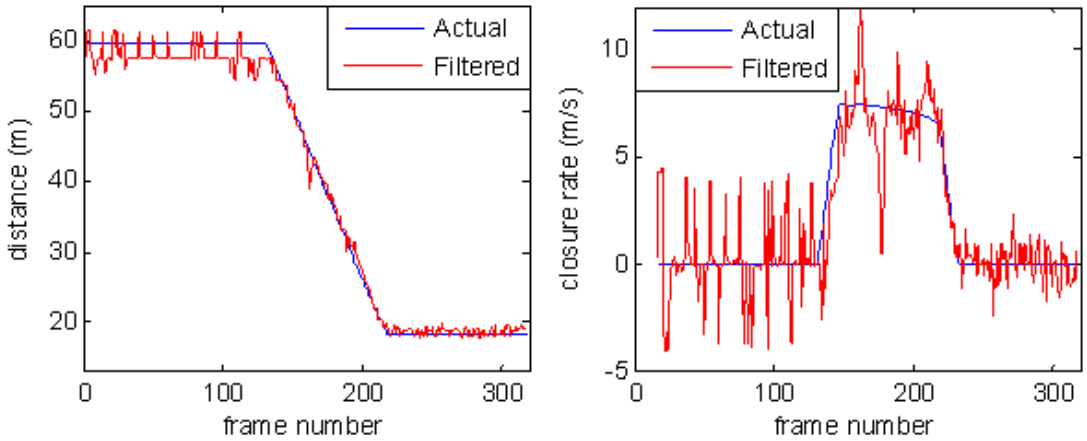
Figure 6.5: Online measurement noise estimation with different sizes of sliding window

(especially at long distances from the cameras where the measurement noise is largest). At the same time, however, due to its greater reliance on the measurements, the Kalman filter is very responsive. Therefore, there is very little lag between the filtered and the actual distance and closure rate profiles.

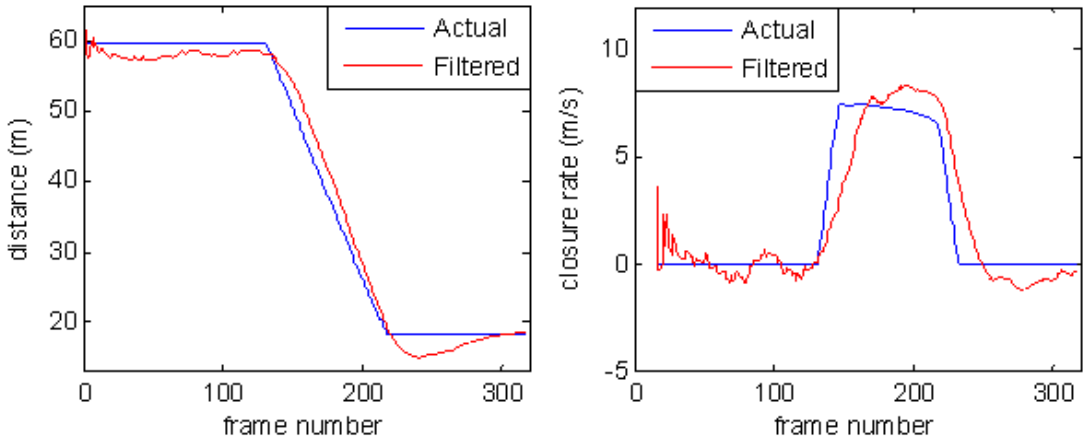
When R is set to a value that is larger than the actual measurement noise, the Kalman filter relies more on the predictions. As observed in Figure 6.6(b), the output of the filter is very smooth but its bandwidth is less than that of the filter in the first test case. This can be observed from the fact that the Kalman filter is slow to react to changes in the distance and closure rate profiles, particularly when the ownship

starts moving (in Frame 131) and when it stops later on in the sequence (in Frame 218). This makes it more difficult for the Kalman filter to track obstacles in dynamic situations.

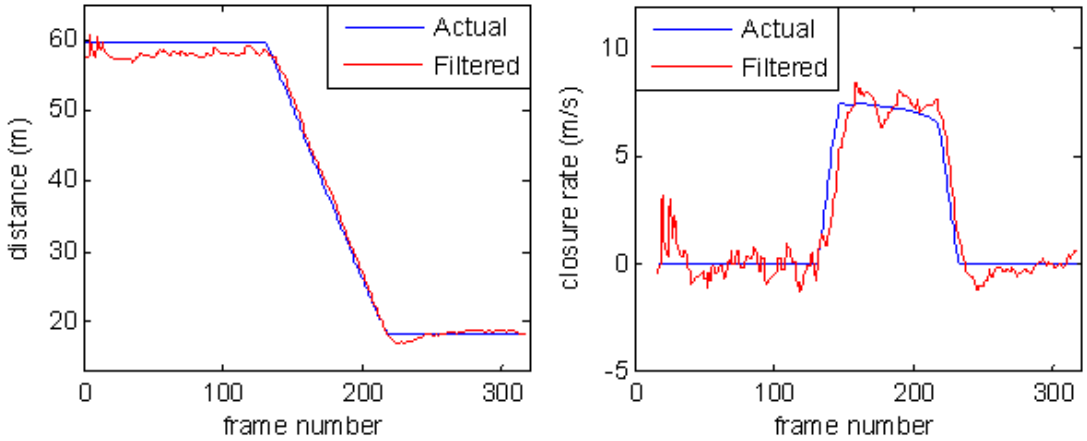
With online measurement noise estimation, the resultant filter is adaptive and changes the relative importance of the measurements and the predictions depending on the distance of the obstacle from the cameras. The closer the obstacle, the lower the measurement noise and the greater the reliance of the Kalman filter on the measurements. On the other hand, the greater the distance, the larger the measurement noise and the greater the importance of the predictions. As a result, in Figure 6.6(c) it can be observed that the output of the Kalman filter is much smoother than in the first test case (when the magnitude of R is underestimated) and its bandwidth is greater than in the second test case (when the magnitude of R is overestimated).



(a) Distance and closure rate estimates when $\sigma_{z_1} = \sigma_{z_2} = 0.1m$



(b) Distance and closure rate estimates when $\sigma_{z_1} = \sigma_{z_2} = 10m$



(c) Distance and closure rate estimates when R is estimated online

Figure 6.6: Distance and closure rate estimates obtained for different values of R

6.3 Obstacle Tracking and Outlier Rejection

Although the ownship may be surrounded by several obstacles, only the closest obstacle is tracked. If there are obstacles inside the protection zone, the system tracks the obstacle point that is closest to the wingtip. Otherwise, if there are obstacles only outside the protection zone, the system tracks the obstacle point that is closest to the protection zone boundary. Figure 6.7 shows which obstacle point is tracked in typical conflict scenarios.

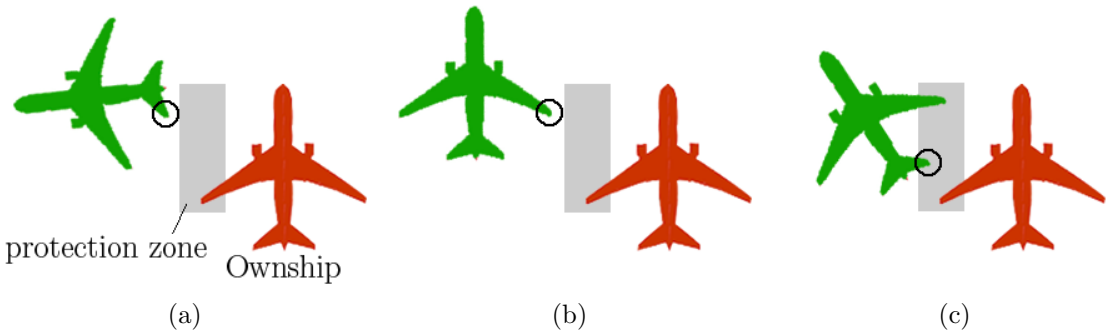


Figure 6.7: Obstacle point selection in obstacle tracking

If no obstacle is being tracked and an obstacle is detected in the current frame, the Kalman filter is initialised and starts tracking that obstacle. If an obstacle is already being tracked, its current state $\hat{\mathbf{x}}'_k$ is predicted using Equation (6.2.6). Then, if an obstacle is detected in the current frame, the system checks whether $\hat{\mathbf{x}}'_k$ and the measurement $\mathbf{z}_k$ correspond to the same obstacle. This is done by calculating the distance between the predicted and measured obstacle positions. If the distance is within a certain threshold, $\hat{\mathbf{x}}'_k$ and $\mathbf{z}_k$ are assumed to correspond to the same obstacle. Then, the update equations of the Kalman filter are applied to obtain the state estimate $\hat{\mathbf{x}}_k$.

If an obstacle is being tracked but goes missing in a frame (either because no obstacle is detected or because the distance between the measured and the predicted obstacle position exceeds the threshold), it can still be tracked for a certain number of frames by relying solely on the prediction stage of the Kalman filter. The

number of consecutive frames that an obstacle can go missing and still be tracked ($mFramesAllowance$) depends on the number of frames in which it has been tracked ($framesTracked$):

$$mFramesAllowance = \begin{cases} \lfloor framesTracked/a \rfloor & \text{if } framesTracked < b \\ \lfloor b/a \rfloor & \text{if } framesTracked \geq b \end{cases} \quad (6.3.1)$$

where $a = 5$ frames and $b = 25$ frames.

This technique makes the tracking process more robust because obstacles that are detected as a result of noise are unlikely to be tracked for more than a few frames. In fact, when testing the tracking algorithm, it was observed that such obstacles are rarely detected for more than 4 consecutive frames. Therefore, by setting a to 5 frames, these obstacles are rejected by the system as soon as they go missing. On the other hand, a true obstacle is likely to be detected consistently in an image sequence. The longer it is tracked, the more reliable it is considered to be and the greater the number of frames that it can go missing. Nevertheless, an upper limit is defined for $mFramesAllowance$. This is done so that, even when an obstacle that has been tracked for a large number of frames goes missing because a new obstacle threat is detected, the algorithm can start tracking the new obstacle without incurring a big delay which might otherwise affect the system's ability to detect conflicts. b was chosen such that the maximum possible delay incurred in tracking a new obstacle is 5 frames (or $\frac{1}{3}$s). The flowchart in Figure 6.8 shows how the system detects new and existing obstacles during the tracking process.

After $\hat{\mathbf{x}}_k$ (or $\hat{\mathbf{x}}'_k$ in the case of a missing obstacle) is obtained, the current distance D_k between the ownship's wingtip and the obstacle is determined. Then, the closure rate V_k between the obstacle and the wingtip is estimated using Equation (6.3.2):

$$V_k = D_{k-15} - D_k \quad (6.3.2)$$

where D_{k-15} is the distance between the wingtip and the obstacle 15 frames (i.e. 1s) previously. The closure rate therefore starts being estimated after 15 frames (or 1s) of tracking. If V_k is positive, the obstacle is approaching the wingtip and the Time

to Collision TTC is given by:

$$TTC = \frac{D_k}{V_k} \tag{6.3.3}$$

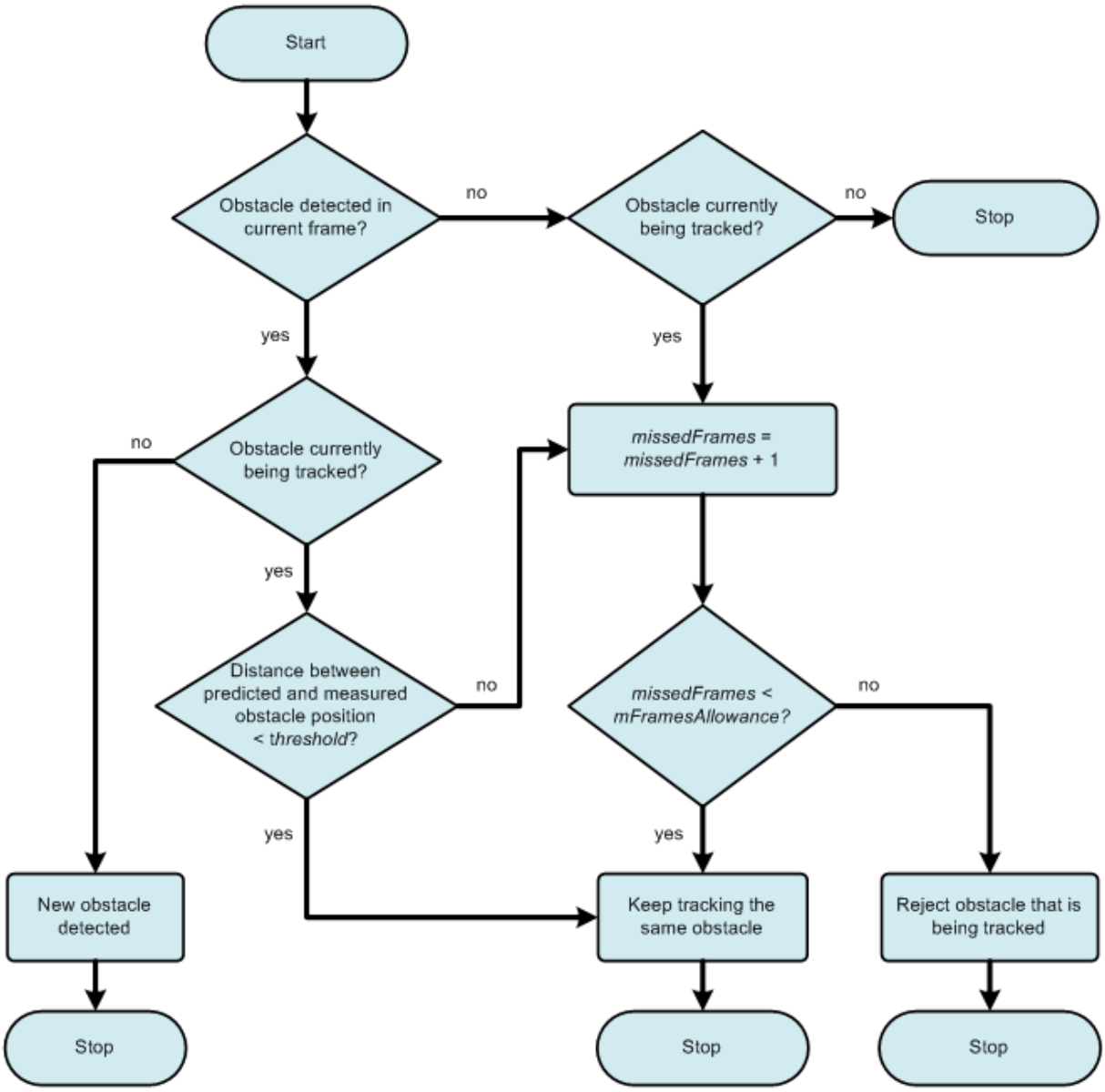


Figure 6.8: Flowchart of logic used to detect new obstacles, track existing obstacles, and reject outliers

6.4 Tracking Results

Figures 6.9-6.12 show the results obtained for two tracking scenarios where the tracked obstacle is part of a second aircraft (the *target*).

In the first scenario, the ownship is executing a right turn at the intersection between two taxiways. The target is a stationary aircraft of similar size to the ownship and is also located at the intersection, to the left of the ownship (Refer to the plan view of the scenario in Figure 6.9(b)). Initially, there are no obstacles inside the protection zone and the right wingtip of the target is the closest obstacle to the protection zone boundary. Therefore, it is selected for tracking (The tracked obstacle is marked with a white spot in Figure 6.9(a)). The wingtip starts being tracked from a distance of around 85m (when it first enters the common FOV of the stereo vision system) and is tracked up to a distance of around 16m from the wingtip of the ownship. Beyond that point, the target's right wingtip moves outside the common FOV of the cameras.

The measured, filtered and true obstacle positions for each frame of the first tracking sequence are shown in Figures 6.9(c) and 6.9(d). During this tracking sequence, there are five frames (Frames 40, 61, 63, 64 and 65) in which the distance between the predicted and measured obstacle position exceeds a certain threshold. Therefore, in each of these frames, the target's right wingtip is considered to be missing. These frames can be identified by 'spikes' in the measurement profiles. The spikes are due to the fact that the tracking algorithm momentarily selects a different obstacle for tracking. In these frames, the Kalman filter still manages to track the target's right wingtip and obtains a good estimate of the obstacle's position by ignoring the measurements completely and relying only on its predictions.

Figures 6.9(e) and 6.9(f) show how the distance and relative position and closure rate of the obstacle with respect to the ownship's left wingtip vary during the tracking sequence. It can be observed that, since the closure rate is obtained indirectly from the distance estimates (which are based on the Kalman filter state vector estimates), the closure rate profile is noisier than the distance profile. From the distance and closure rate estimates, the Time to Collision TTC is predicted. A plot of the actual and

estimated TTC is given in Figure 6.10. From Frame 19 onwards, the error between the actual and estimated TTC settles within $\pm 1s$ (At Frame 19, the actual distance between the ownship's left wingtip and the obstacle is 68.6m).

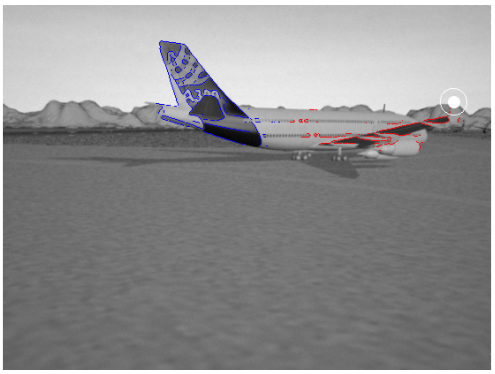
In the second tracking scenario, the ownship is to the left of the taxiway centreline and is taxiing at an average speed of 15kts. A stationary target is located to the north-west of the ownship, at the intersection between two taxiways (Refer to Figure 6.11(b)). Initially, the system tracks the left wingtip of the target because this is the closest obstacle to the protection zone boundary. However, as the ownship continues approaching the target, the horizontal stabiliser of the target becomes the closest obstacle to the protection zone and is selected for tracking (The tracked obstacle is marked with a white spot in Figure 6.11(a)). The stabiliser starts being tracked when the distance between it and the ownship's wingtip is less than 65m.

In the second scenario, the tracked obstacle goes missing in four frames (Frames 10, 88, 96 and 97). Some of these frames can be identified by spikes in the measured x and z coordinates of the obstacle in Figures 6.11(c) and 6.11(d). As in the first scenario, the Kalman filter tracks the obstacle successfully during these frames by relying on the prediction stage of the filter.

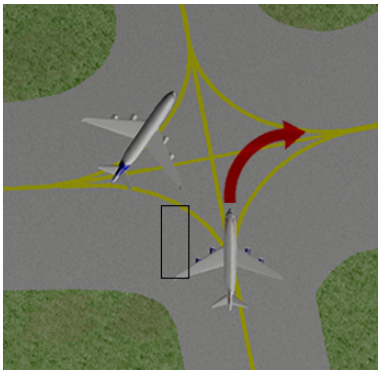
A plot of the actual and estimated TTC for the second scenario is presented in Figure 6.12. From Frame 21 onwards, the error between the actual and estimated TTC settles within $\pm 1s$ (At Frame 21, the actual distance between the ownship's left wingtip and the obstacle is 52.4m).

From the results presented in this section, it has been shown that the tracking algorithm is capable of tracking aircraft extremities, both within and outside the protection zone. The Kalman filter provides better obstacle position estimates when compared to the raw measurements and enables the algorithm to track obstacles reliably even when they go missing for a few frames.

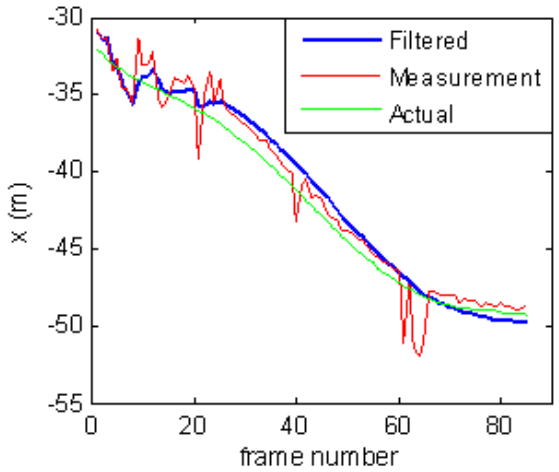
The tracker also estimates the TTC. This is one of the main parameters that can be used to decide whether and when to issue an alert. The accuracy of the TTC is important because it affects the performance of the system. If the TTC is



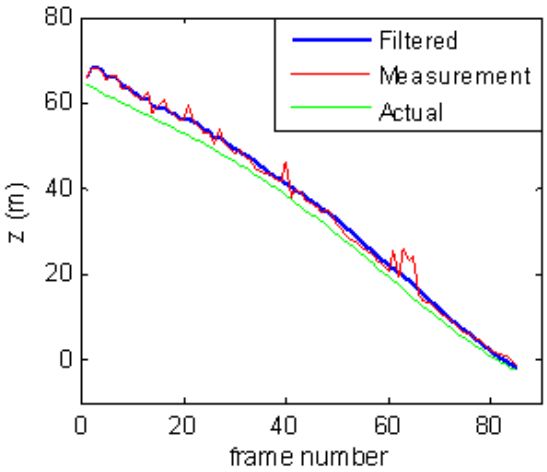
(a) Obstacle pixels superimposed on intensity image



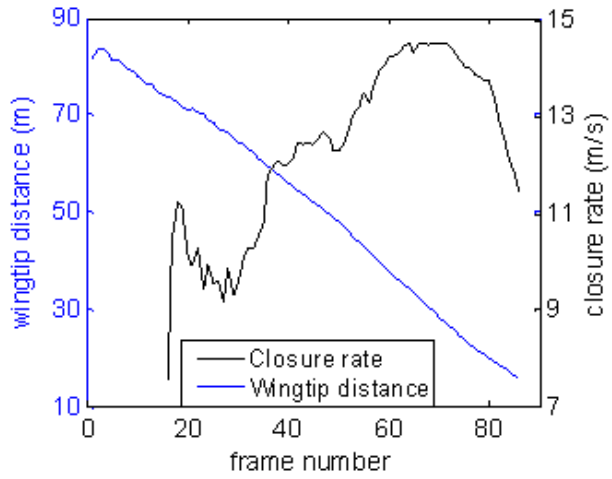
(b) Plan view



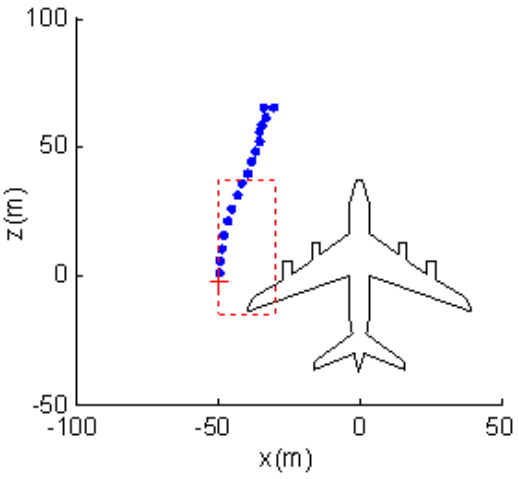
(c) x coordinate of tracked obstacle point in ARF



(d) z coordinate of tracked obstacle point in ARF



(e) Wingtip distance and closure rate



(f) Obstacle trajectory

Figure 6.9: Tracking (Example 1)

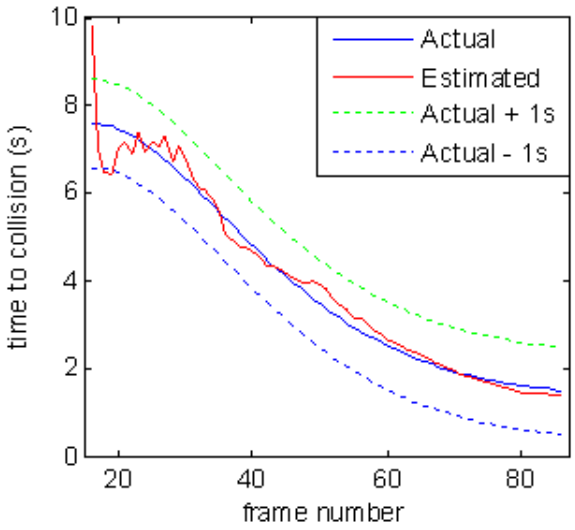
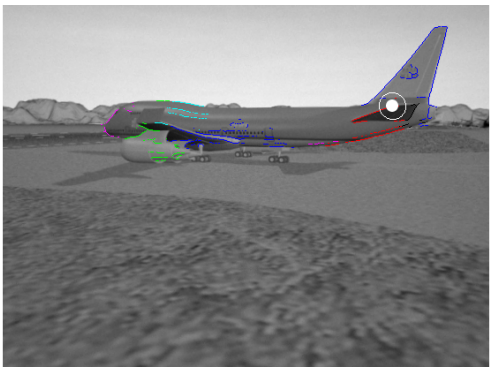
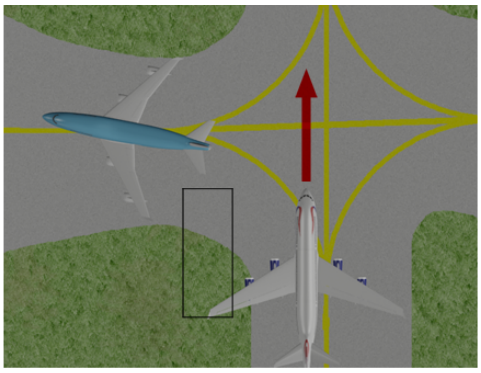


Figure 6.10: Tracking (Example 1): Time to collision

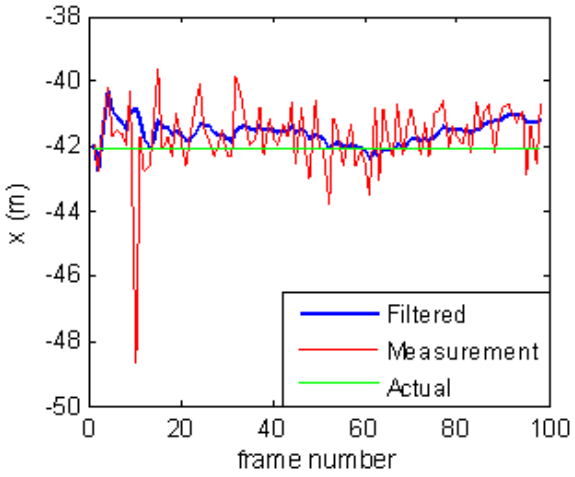
underestimated, then an alert might be generated earlier than necessary or might even be issued in scenarios where no real conflict exists. On the other hand, if the TTC is overestimated, then an alert might be delayed or might not be generated at all, potentially leading to a collision. Therefore, the error in the estimation of the TTC needs to be taken into consideration when defining the alerting strategy of the system, in order to minimise the occurrence of false (nuisance) alerts and missed alerts.



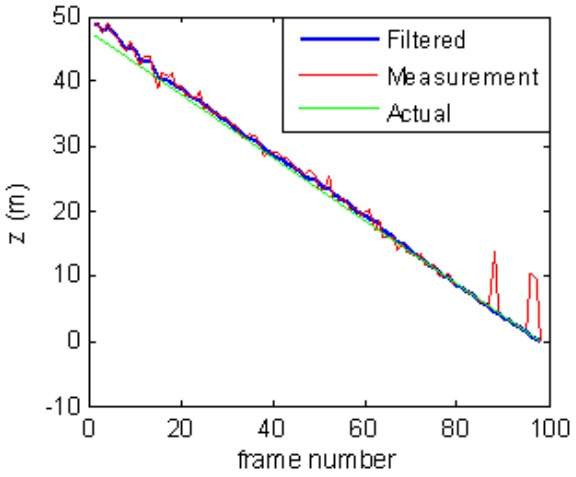
(a) Obstacle pixels superimposed on intensity image



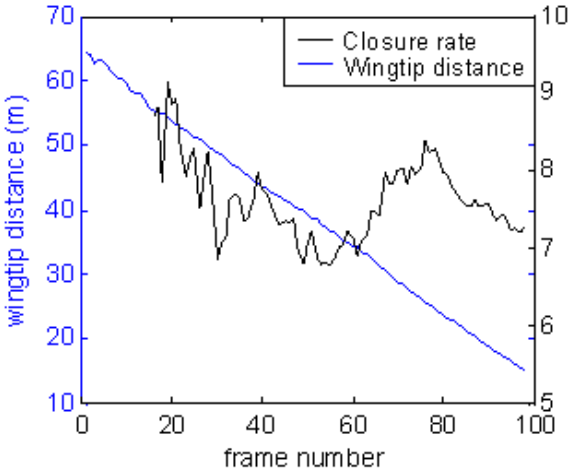
(b) Plan view



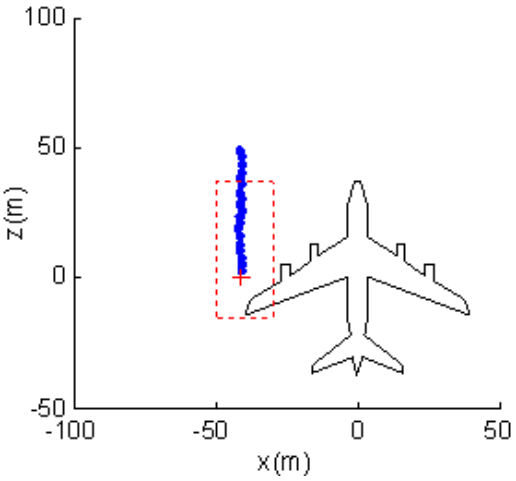
(c) x coordinate of tracked obstacle point in ARF



(d) z coordinate of tracked obstacle point in ARF



(e) Wingtip distance and closure rate



(f) Obstacle trajectory

Figure 6.11: Tracking (Example 2)

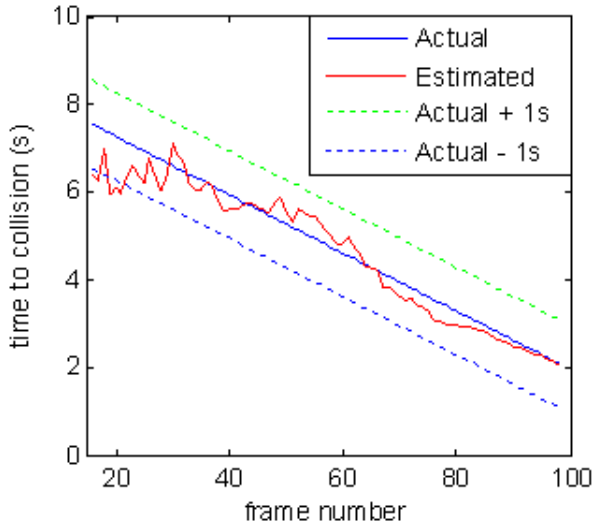


Figure 6.12: Tracking (Example 2): Time to collision

Chapter 7

Testing of the Overall System

In the previous chapters, the processing blocks of the stereo vision system were discussed individually and some results were presented for each block. The aim of this Chapter is to examine the overall performance of the system and to identify any limitations by testing it in several scenarios under different conditions, using both synthetic images and real images.

Section 7.1 looks at simulation testing. Simulation testing offers several advantages, including: (a) the possibility to study scenarios that are too difficult or dangerous to set up in a real environment, (b) the complete control over simulation parameters, which makes it possible to simulate a wide variety of conditions (such as different weather and lighting conditions), (c) the repeatability of experiments and (d) the availability of ground truth data. Section 7.1 first describes the experiments that were carried out using synthetic data. Then, the results of these experiments are presented and discussed.

Obviously, the system designed in this research is intended to be used in the real world. Therefore, in Section 7.2, the performance of the system is determined by testing it with images obtained using real cameras, in a typical aerodrome environment.

The simulated environment is only a model of the real world because certain aspects of the real world are difficult to replicate accurately through simulation. Therefore, one of the disadvantages of simulation is transferability, that is the

difficulty of designing a system solely on the basis of simulated data and then achieving the same performance in real-world scenarios as that predicted by the simulations. In order to check whether the performance of the system in the real world is similar to that predicted by the simulations (and therefore assess the fidelity of the simulated environment), Section 7.3 compares the results obtained from simulation testing and real-world testing.

7.1 Experiments with Synthetic Images

7.1.1 Design of Experiment

Experiments were carried out in order to test three main aspects of the system: (a) the effect of absolute extrinsic calibration errors on positional accuracy,[1] (b) generic obstacle detection capability in varying conditions and (c) generic obstacle tracking capability in varying conditions.

7.1.1.1 Sensitivity to absolute extrinsic calibration errors

The method of using individual targets for absolute extrinsic calibration is prone to human error. Errors are typically introduced in target position and orientation. To study the effect that such errors would have on the positional accuracy of the system, two experiments were carried out. In the first experiment, the targets were all placed in the correct position and the absolute extrinsic calibration parameters were found. Then, a well-textured object was placed at different positions in the WRF. The z coordinate of the object was varied between 12.5m and 55m and its x coordinate was varied between -10m and 10m. The position estimate of the object at each location was obtained and the error in the detected position was determined. The method used to estimate object position was the same as that described in Section 5.1.2.

[1]The positional accuracy of the system was discussed in detail in Chapter 5 in the discussion of the selection of the baseline distance and focal length.

In the second experiment, errors were deliberately introduced in the position and orientation of each of the targets used in the calibration process. The errors were selected from normal distributions with 0 mean and were generated for positional accuracy in the x and z axes, as well as in rotation about the vertical axis of each target. The standard deviations of the distributions are shown in Table 7.1. The absolute extrinsic calibration parameters were then found and the positional accuracy was determined as in the first experiment.

Table 7.1: Errors introduced in target position and orientation

Errors ($\pm 2\sigma$, 95.4% confidence)		
Δx (cm)	Δz (cm)	Δyaw (°)
10	10	10

The second experiment was repeated 50 times (by inserting a different error in the location and orientation of every target each time) in order to characterise the effect of random errors on repeated calibration processes. This effectively resulted in an uncertainty in the measurements that can be modeled by a normal distribution. Accordingly, at the end of the experiment, the mean and deviation of the error at each object position was determined.

7.1.1.2 Obstacle detection

The ability of the system to detect different types of obstacles has already been demonstrated in Chapters 5 and 6. However, the results presented in those chapters were obtained by testing the system on a small number of scenarios, in good visibility and lighting conditions, and with low levels of temporal image noise. Therefore, these results are not enough to gauge the performance of the system. As discussed in Section 1.3, one of the desirable properties of the system is to have low rates of Type I errors (false positives or false alarms) and Type II errors (false negatives or missed alarms). A reduction of Type I errors can often be achieved at the expense of an increase in Type II errors, and vice-versa. For the application considered in this

research, it is desirable to have a low false alarm rate at the potential expense of a slightly higher missed alarm rate.

In order to determine the robustness of the system in terms of missed detections and false detections, two image sequences - each 1500 frames long - were generated. Both image sequences were obtained by simulating several conflict scenarios on ramps and taxiways, including some of the most common incidents and accidents identified in Section 1.2. When simulating these scenarios, care was taken to (a) incorporate the most common visual aspects of ramps and taxiways, (b) use a variety of commercial aircraft and other obstacles to create a cluttered environment, and (c) simulate wing bending by randomly adjusting the vertical position of the cameras in each frame. In many of the scenarios, the separation between the ownship and obstacles is compromised due to one or more of the following reasons: (a) the ownship (or another aircraft) deviates from the taxiway centreline, (b) a vehicle (or aircraft) parks incorrectly on the ramp, (c) the ownship (or another aircraft) manoeuvres on a taxiway which is designed to handle smaller aircraft. In most of the scenarios, the ownship taxis at an average speed of 15kts whereas the obstacles are stationary. The details of all of the simulated obstacle detection scenarios are provided in Appendix E.1.

The first of the image sequences was used to determine the missed detection rate of the system. In each of the frames of this sequence, obstacles penetrated the protection zone of the ownship's left wingtip and the obstacles were within the stereo vision's common FOV. A missed detection occurred whenever the obstacles were not detected in a particular frame. After all the frames were processed, the missed detection rate was estimated from the percentage of frames of the image sequence in which a missed detection occurred.

The second image sequence was used to find the false detection rate of the system. In each of the frames of this sequence, obstacles came within a few metres of the boundary of the protection zone but no obstacle actually penetrated the protection zone. A false detection occurred whenever an obstacle was detected inside

the protection zone when, in reality, it was either outside the protection zone or corresponded to a ground or sky feature. The false detection rate was then estimated from the percentage of frames of the image sequence in which a false detection occurred.

In order to test the sensitivity of obstacle detection to variations in illumination, visibility and temporal image noise, the experiment was repeated for 9 different combinations of these variables, as shown in Table 7.2. These simulation test cases represent a very small subset of the lighting and visibility conditions that can be present in the scene. They are not intended to test the system for a specific illumination or visibility condition but to get a general idea of the expected performance of the system in three main illumination/visibility categories: good illumination (daylight), low illumination (night), and low visibility (fog). Also, the main idea of testing the system with different levels of image noise is not to determine the performance of the system for a particular noise level but to study the effect of an increase in image noise on overall performance.

In practice, the lighting conditions vary a lot depending on several factors, such as: the presence (or absence) of the sun in the camera FOV; the cloud cover; the position of the sun in the sky (which changes continuously throughout the day) and the presence of direct or diffuse illumination. The performance of the system is expected to change for each of these conditions, mainly because of differences in image contrast.

Table 7.2: The different combinations of illumination, visibility and image noise used when simulating the conflict scenarios

Illumination/visibility	Day			Night			Fog		
Image noise σ (intensity levels)	3	10	20	3	10	20	3	10	20

7.1.1.3 Tracking

The obstacle tracking capability of the system was determined by simulating 6 tracking scenarios on ramps and taxiways. Five of these scenarios were purposely chosen to test the system's ability to track different aircraft extremities, such as the wingtips, tail and nose cone. The details of the tracking scenarios are given in Appendix E.2.

In each scenario, an obstacle was selected for tracking by applying the tracking logic explained in Section 6.3 and the following parameters were recorded in each case: (a) the distance at which the obstacle started being tracked, (b) the number of tracked frames, (c) the number of missed frames[2] and (d) the total length of the tracking sequence. These parameters were then used in conjunction with the distance, closure rate, and time to collision profiles in order to gauge the performance of the tracking algorithm.

This experiment was repeated for the same illumination, visibility and image noise conditions used to test the obstacle detection capability of the system (Refer to Table 7.2).

7.1.2 Results

7.1.2.1 Sensitivity to absolute extrinsic calibration errors

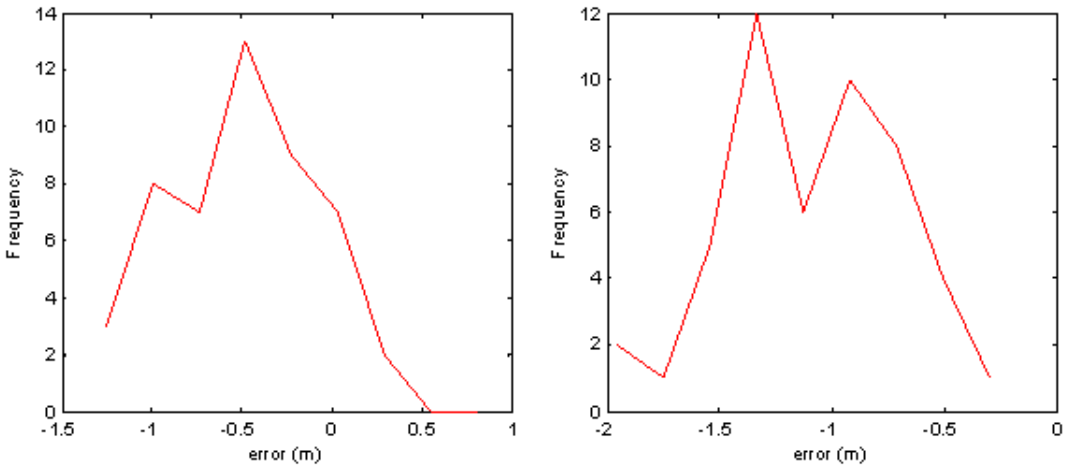
Figure 7.2 presents the measurement errors observed in the two experiments. Figures 7.2(a) and 7.2(b) show the results of the first experiment whilst Figures 7.2(c)-7.2(f) show the results of the second experiment. One general observation is that the positional error increases with distance from the cameras. This is due to the uncertainty of triangulation. The effect of the calibration errors in the second experiment can be clearly observed. The error at each obstacle position is, in essence, a random error and can be modeled by a Gaussian distribution with

[2]This is the number of frames (not necessarily consecutive frames) in which an obstacle went missing, but was still tracked, during the tracking sequence.

a mean value (mean measurement error) that depends on the object position with respect to the camera setup. The error distribution in the x and z axis observed for one particular object position is shown in Figure 7.1 and this confirms the Gaussian nature expected.

Referring to Figure 7.2, since the error at the calibration target positions was designed with 0 mean, the mean error at the output (Figures 7.2(c) and 7.2(d)) is very similar to the error obtained in the first experiment (Figures 7.2(a) and 7.2(b)). The mean error is less than 0.8m in the x axis and less than 2m in the z axis. The standard deviation of the error increases with distance and the 95.4% confidence interval (2σ) reaches a maximum of $\pm$1.1m in the x axis and $\pm$0.95m in the z axis, at a range of 55m from the origin of the WRF (Refer to Figures 7.2(e) and 7.2(f)).

From these results it can be observed that the calibration process is quite sensitive to errors because a small calibration error (of a few centimetres in position and a few degrees in orientation of the calibration targets) produces an additional positional error (on top of the error due to triangulation uncertainty) with a potential magnitude of a few metres. The decrease in positional accuracy due to this error can potentially result in more false alerts and missed alerts, particularly in the case of obstacles that are located close to the boundary of the protection zone of the ownship. For instance, as discussed in Section 1.3, if an obstacle is outside the protection zone but is detected inside, a false alert might be generated. On the other hand, if an obstacle is inside the protection zone but is detected outside, an alert might be delayed or might even not be generated at all. Therefore, in order to minimise these occurrences, extra care needs to be taken when positioning the targets during absolute extrinsic calibration.



(a) Error distribution in the x axis (b) Error distribution in the z axis

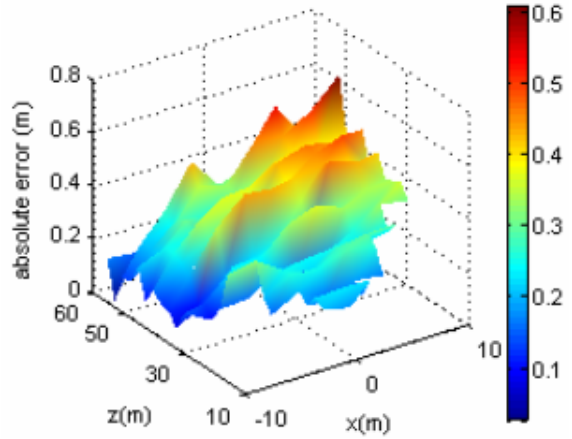
Figure 7.1: Error distribution in the x and z axes at target position (x=10m, z=45m)

7.1.2.2 Obstacle detection

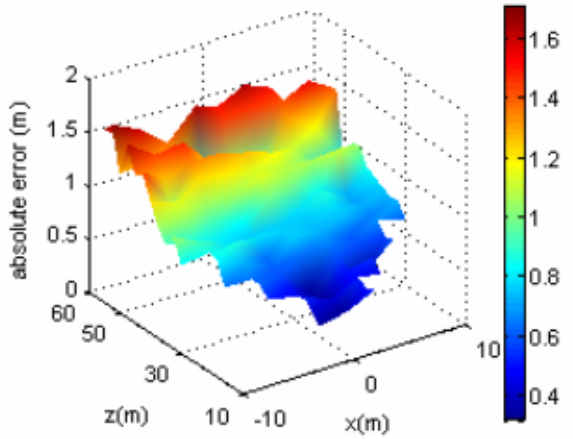
Figures 7.3-7.8 show two examples of the obstacle detection results obtained in different illumination and visibility conditions, with variable quantities of temporal image noise. In the left column of each figure, the obstacle points are superimposed over the left intensity image whereas, in the right column, they are plotted in the ARF. Each cluster of obstacle points is represented by a different colour such that it is possible to match each cluster in an intensity image with its corresponding cluster in the ARF.[3] The first example is taken from Obstacle Detection Scenario 4 (Refer to Table E.1), where the ownship is taxiing at 15kts behind a stationary A380 situated on a parallel taxiway. The second example is taken from Obstacle Detection Scenario 5, where the ownship is turning into the ramp area while an A380 is taxiing towards it at 15kts from the opposite direction.

In general, the best obstacle detection results are obtained in good illumination conditions, due to the high level of contrast of the intensity images. For example,

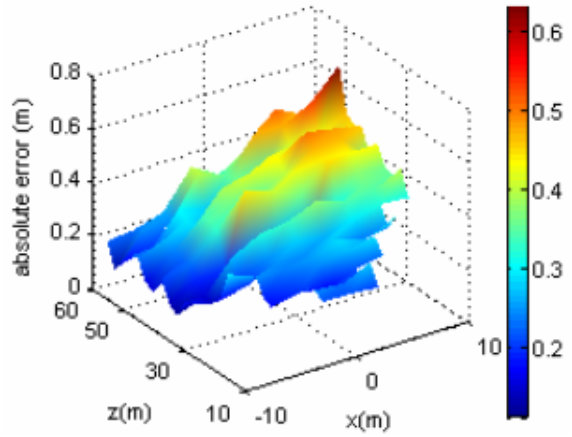
[3]Please note that the colours of the clusters are only consistent for a particular combination of image noise and illumination/visibility conditions. This means that the colour of a cluster representing a particular obstacle can change in different conditions.



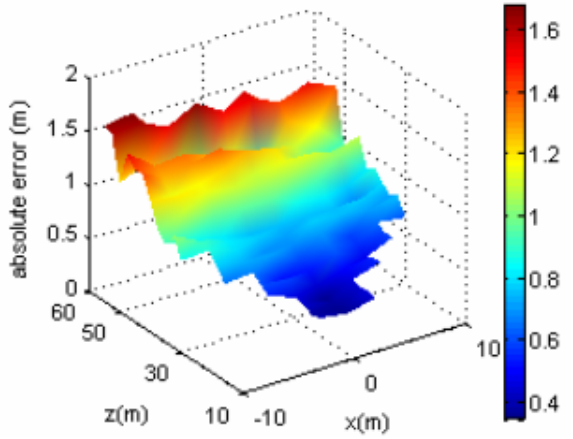
(a) Error in x axis wrt position in WRF



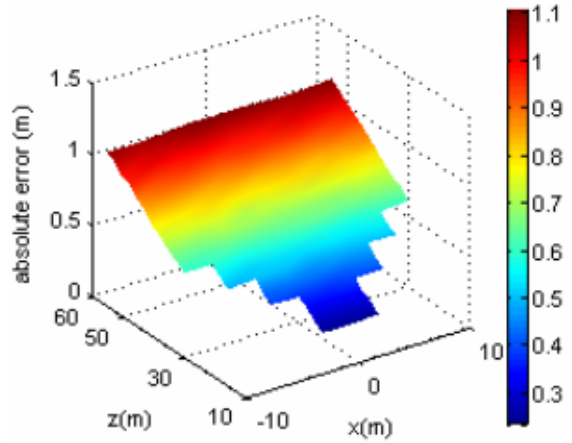
(b) Error in z axis wrt position in WRF



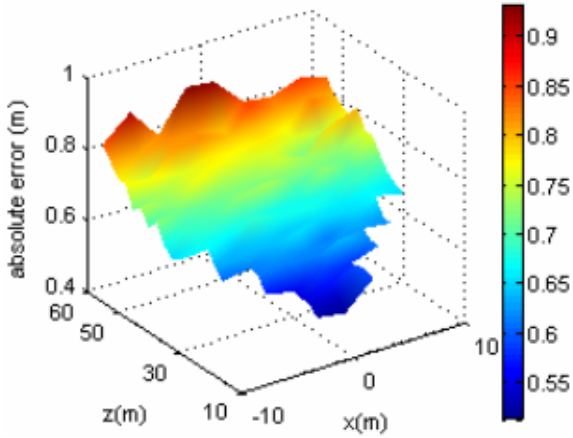
(c) Mean error in x axis wrt position in WRF



(d) Mean error in z axis wrt position in WRF



(e) 2σ of error in x axis wrt position in WRF



(f) 2σ of error in z axis wrt position in WRF

Figure 7.2: Errors observed in the WRF when (a, b) no error is made in target position and orientation and (c-f) when an error is introduced during absolute extrinsic calibration (Refer to Table 7.1)

in Figures 7.6(a) and 7.6(b), the overall shape and orientation of the approaching aircraft is quite clear from the projection of the obstacle points in the ARF. This implies that the triangulation errors are relatively small (compared to the dimensions of the aircraft) and that, therefore, the system achieves good positional accuracy. The aircraft is detected as three separate clusters (corresponding to the left wing, the tail section, and the right wing and fuselage, respectively) whereas the terminal building in the background is detected as a single cluster. The clusters are also a good representation of the obstacles they correspond to in Figure 7.3(b). Here, the system detects both the aircraft on the parallel taxiway as well as another aircraft crossing ahead at a taxiway/taxiway intersection. The aircraft on the parallel taxiway is detected as two clusters, where the smaller cluster corresponds to some windows on the aircraft. Due to errors during correspondence, the obstacle points in this cluster are mapped onto positions in the ARF that are further away from their actual locations. The aircraft that is crossing ahead is detected as two clusters, corresponding to the left wing and the tail section respectively. The section of the fuselage between the wings and the tail is not detected at all because it has few edge features and because of the repetitive pattern of the windows. As explained in Chapter 4, repetitive textures generally result in unreliable matches during correspondence and most of these matches fail the confidence tests defined in the correspondence algorithm (Refer to Section 4.2.4).

Under poor light conditions, the image contrast decreases and it becomes harder to find corresponding pixels. Correspondence matches become weaker and, therefore, more ambiguous. As a result, a greater number of disparities are rejected by the correspondence algorithm. In this case, although the edge detector is sensitive enough to detect even the weakest edges, only the strongest edge features are successfully detected as obstacles. This is because strong edges normally provide more reliable correspondence matches. For instance, in the first example, most of the windows of the closest aircraft are detected in good illumination (Figure 7.3(a)) but not in dark conditions (Figure 7.4(a)). Also, in the second example, less hangar and aircraft

features are detected in low illumination (Figure 7.7(a)) than in good illumination (Figure 7.6(a)). Apart from the fact that a larger number of disparities are rejected, the disparities that are retained are still not as accurate as those obtained in good illumination conditions. As a result, the positional accuracy of the system degrades and this, in turn, means that the obstacle clusters lose their shape, making it more difficult to identify the shape and orientation of an obstacle from its corresponding cluster/s. For instance, the shape of the aircraft represented in Figure 7.7(b) (obtained in dark conditions) is less defined than the shape of the same aircraft represented in Figure 7.6(b) (obtained in good light conditions).

In low visibility conditions, such as fog, the image contrast decreases as well and weaker edges are likewise not detected. Naturally, the extent of the reduction of image contrast depends on the density of the fog. In certain cases, however, fog can actually improve obstacle detection by enhancing the contrast between a foreground object and the background. A good example of this can be observed by comparing Figures 7.6(a), 7.7(a) and 7.8(a). In good and low illumination conditions, the contrast between the aircraft's wings and the background is poor. As a result, parts of the wings remain undetected. On the other hand, in low visibility conditions, the background is effectively removed by the fog and this improves the visibility of the wings. Hence, both wings are detected in their entirety. This highlights the variability of the performance of the proposed system and the difficulty of accurately predicting its expected infield performance.

When the image noise standard deviation σ is increased, the obstacle detection results obtained under different illumination and visibility conditions degrade. The positional accuracy decreases and the clusters gradually lose their shape and become less representative of the obstacles they correspond to. For example, from Figure 7.3(f) (when σ is 20 intensity levels) it can be observed that the shape, size and orientation of the green cluster in the ARF are very different from the actual properties of the front end of the aircraft represented by the green cluster in Figure 7.3(e). Also, in the second example, it can be noted that there is a big difference between the

aircraft represented in Figure 7.6(b) (when σ is 3 intensity levels) and that represented in Figure 7.6(f) (when σ is 20 intensity levels).

The increase in image noise also causes ground and sky features to be incorrectly detected as obstacles, either as separate clusters or as part of clusters corresponding to actual obstacles. For instance, in Figure 7.8(d), the red obstacle cluster corresponds to some of the ground features shown in Figure 7.8(c). In Figure 7.4(e), some sky features are grouped with pixels corresponding to the right wing of the closest aircraft. Together they form the red cluster shown in Figure 7.4(f). In the same example, some other sky features and ground features (represented by the blue and cyan clusters respectively) are incorrectly detected as obstacles, with the ground features being detected inside the protection zone.

The main reason for which the obstacle detection results get worse when the image noise is increased is that the SNR of the image is reduced. Consequently, the correspondence algorithm gives poorer results and the triangulation errors increase. In this case, one would expect the stereo vision system to detect less obstacles as the noise level is increased (because a larger number of disparities are rejected by the correspondence algorithm). However, this is not necessarily always the case. This is because an increase in image noise results in an increase of image contrast. Hence, the edge detector detects more edges as the noise level is increased. When σ is increased from 3 to 10 intensity levels, only a small percentage increase in the edge pixels detected is observed (Figures 7.9(a) and 7.9(b)). However, when σ is increased from 10 to 20 intensity levels, the percentage of edge pixels detected increases significantly (Figure 7.9(c)). Due to this large amount of 'false' edges, a larger number of points manage to pass through the system without being filtered out. Apart from degrading the obstacle detection performance, the increase in 'false' edges also results in an increase in computation time since more pixels need to be processed.

Table 7.3 summarises the results obtained when testing the system for missed detections and false detections under different conditions, using the two image sequences described in Section 7.1.1.2. Under good illumination and low visibility

conditions, the detection rate is practically unaffected by an increase in temporal image noise. This is because the image contrast and SNR are sufficiently high to enable good correspondence results. On the other hand, the detection rate decreases with noise under dark conditions. This is due to the fact that, when an increasing amount of noise is added to the low-contrast dark images, an increasing number of 'true' obstacle edges are rejected by the correspondence algorithm because their disparities do not satisfy the confidence tests.

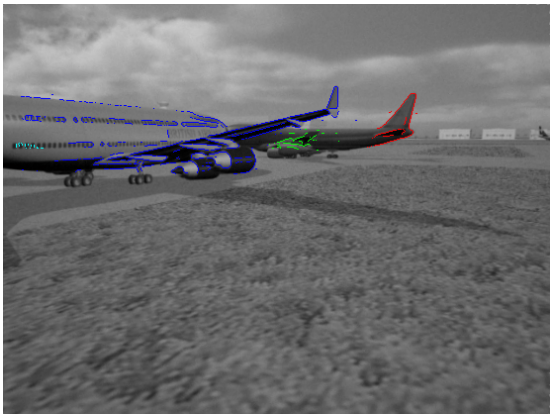
From Table 7.3 it can be observed that, in general, the false detection rate tends to increase with noise. The majority of false detections are due to the detection of ground or sky features. The rest are due to the incorrect localisation of 'true' obstacles. Under good and low illumination conditions, the false detection rate increases slightly when σ is increased from 3 to 10 intensity levels. However, when σ is increased to 20 intensity levels, the false detection rate increases sharply. This occurs because, under these conditions, the increase in the number of 'false' edge pixels far exceeds the number of outliers that are rejected by the system, even though the image quality is reduced. On the other hand, under low visibility conditions, a slightly different result is obtained. When σ is increased to 10 intensity levels, the false detection rate increases. However, when σ is increased again to 20 intensity levels, the false detection rate does not continue to rise (as expected) but decreases slightly. This occurs because, in this case, the correspondence algorithm rejects a large amount of the additional 'false' edges that are introduced by the increase in image noise, thus preventing the false detection rate from increasing any further.

By comparing the false detection and missed detection rates obtained under different conditions, it can be seen that the system tends to be more prone to false detections than missed detections. As mentioned earlier, for this application it is desirable to minimise false detections. Since the detection rate estimates obtained are very good (particularly when σ is less than 20 intensity levels), it is possible and affordable to increase the missed detection rate of the system in return for a lower false detection rate. This can be done by decreasing the sensitivity of the system to

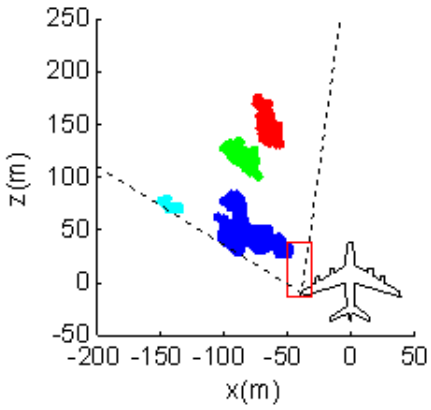
obstacles in three stages. The first stage consists of increasing the thresholds used for edge detection in order to detect only the strongest edge features. The second stage consists of adjusting the thresholds used by the confidence tests during correspondence in order to keep only the most reliable and accurate disparities. Finally, the third stage consists of modifying the thresholds used by the clustering algorithm in order to retain only the biggest and densest obstacle clusters.

Table 7.3: Obstacle detection results

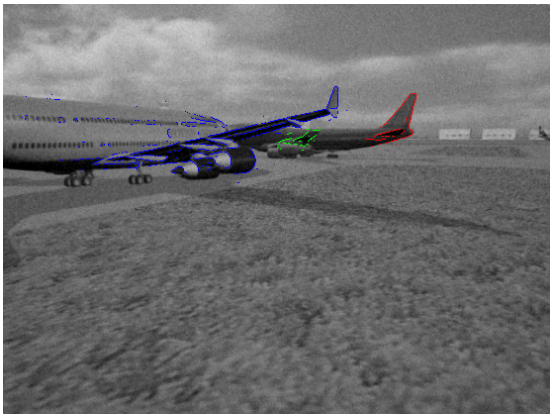
Illumination/visibility	Day			Night			Fog		
Noise σ (intensity levels)	3	10	20	3	10	20	3	10	20
Missed detection rate (%)	0	0	0	2	7	15	0	0	1
Detection rate (%)	100	100	100	98	93	85	100	100	99
False detection rate (%)	0	1	35	0	1	79	1	21	16



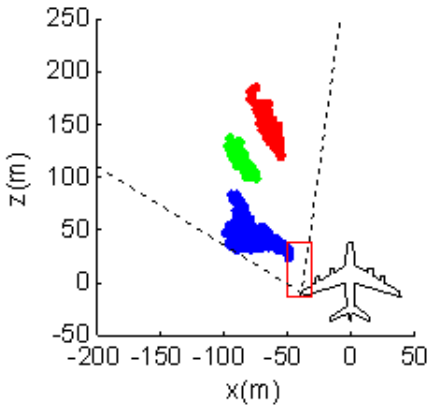
(a) $\sigma = 3$ intensity levels



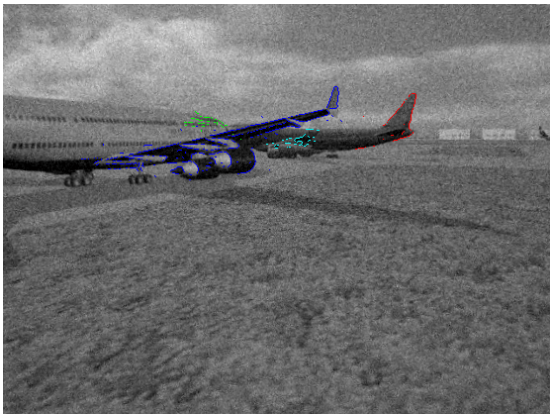
(b) $\sigma = 3$ intensity levels



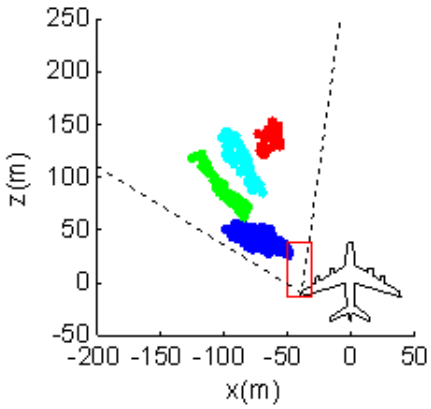
(c) $\sigma = 10$ intensity levels



(d) $\sigma = 10$ intensity levels

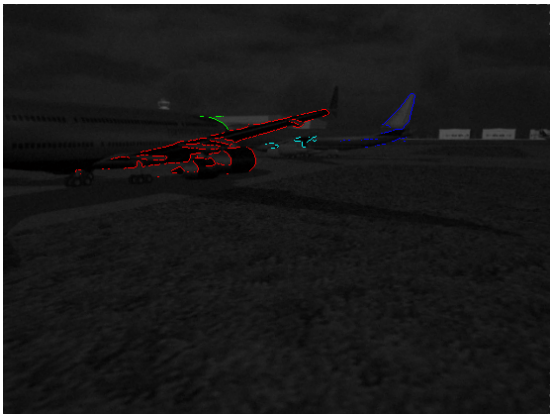


(e) $\sigma = 20$ intensity levels

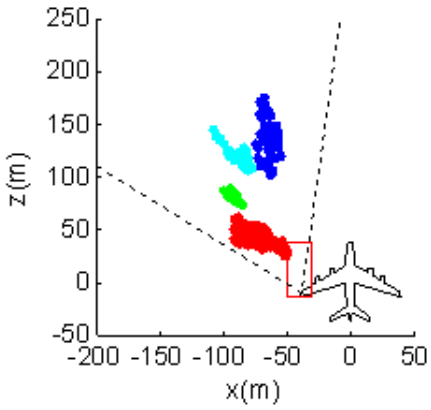


(f) $\sigma = 20$ intensity levels

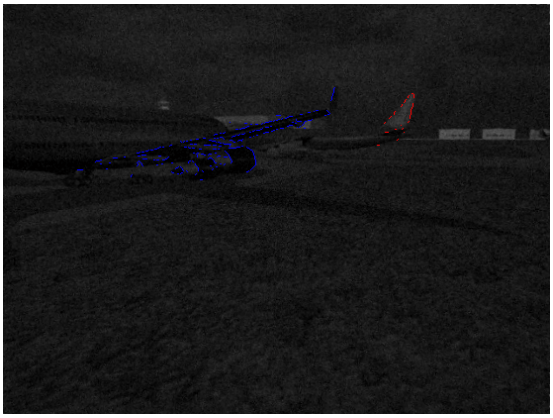
Figure 7.3: Obstacle detection under good illumination (day) for different values of image noise standard deviation σ (Example 1)



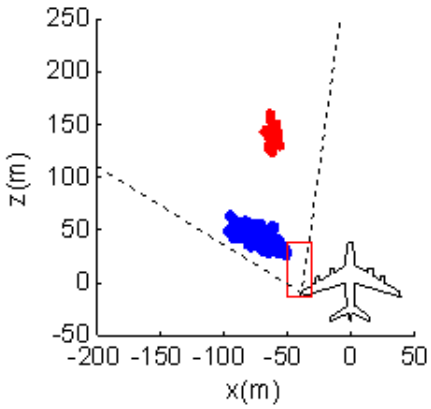
(a) $\sigma = 3$ intensity levels



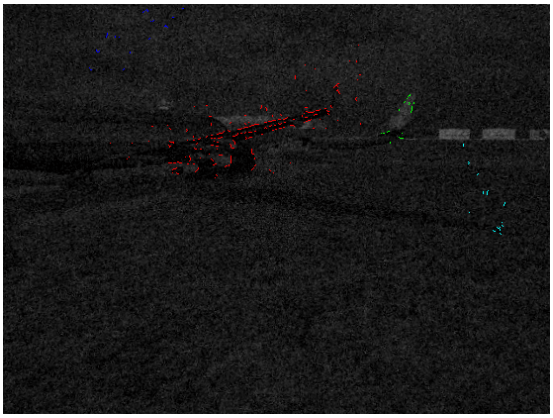
(b) $\sigma = 3$ intensity levels



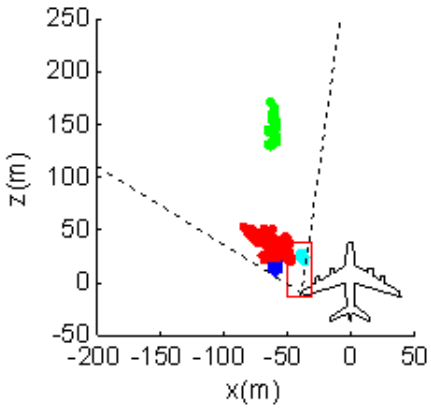
(c) $\sigma = 10$ intensity levels



(d) $\sigma = 10$ intensity levels

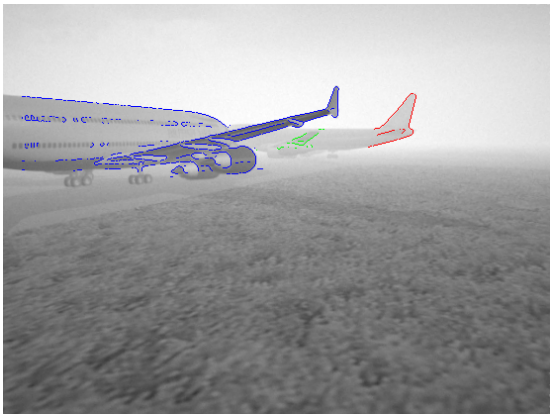


(e) $\sigma = 20$ intensity levels

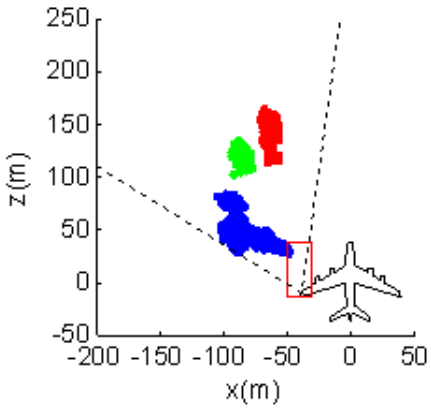


(f) $\sigma = 20$ intensity levels

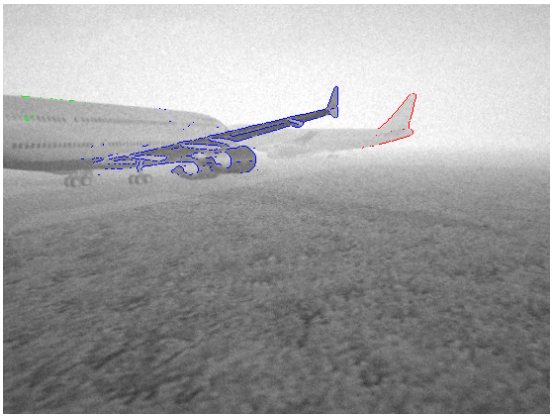
Figure 7.4: Obstacle detection under low illumination (night) for different values of image noise standard deviation σ (Example 1)



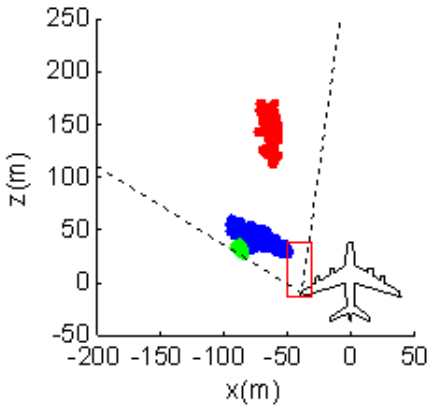
(a) $\sigma = 3$ intensity levels



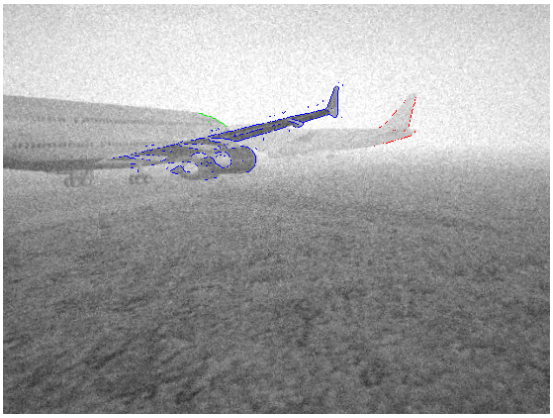
(b) $\sigma = 3$ intensity levels



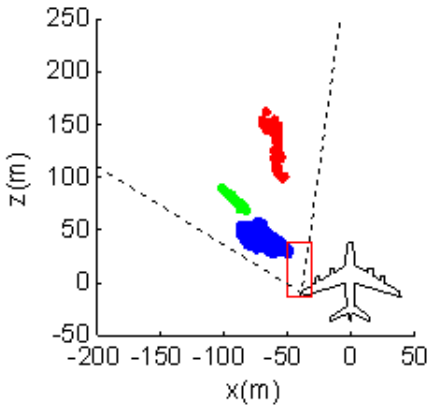
(c) $\sigma = 10$ intensity levels



(d) $\sigma = 10$ intensity levels

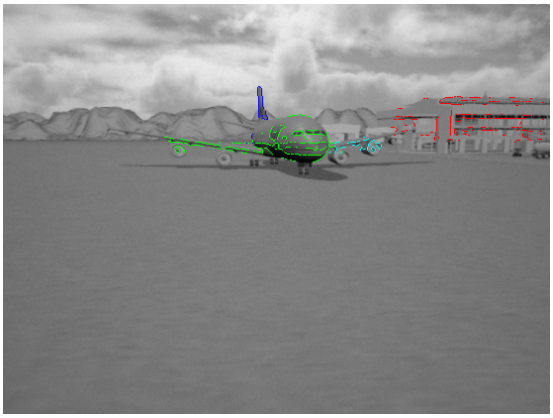


(e) $\sigma = 20$ intensity levels

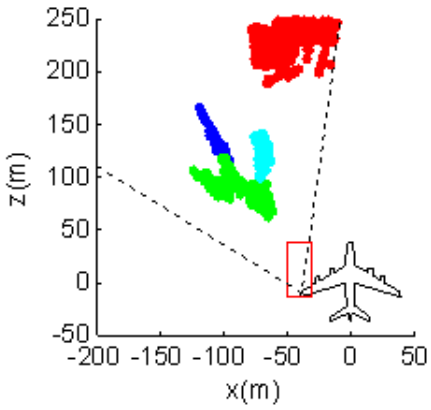


(f) $\sigma = 20$ intensity levels

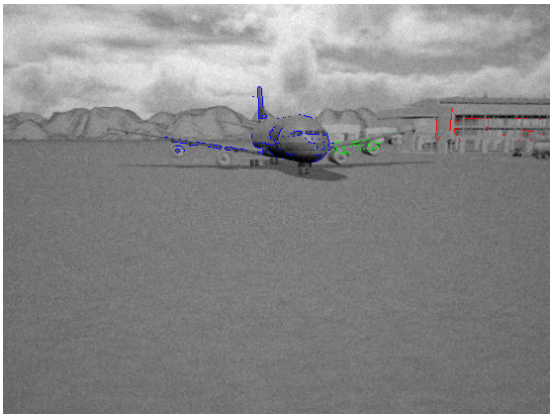
Figure 7.5: Obstacle detection under low visibility (fog) for different values of image noise standard deviation σ (Example 1)



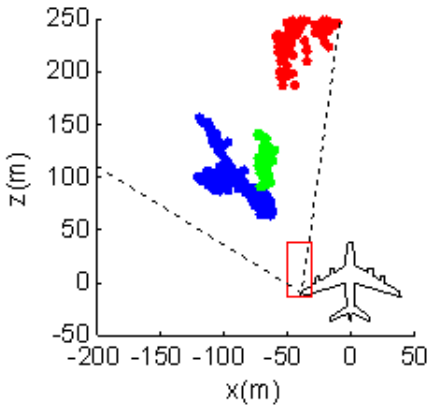
(a) $\sigma = 3$ intensity levels



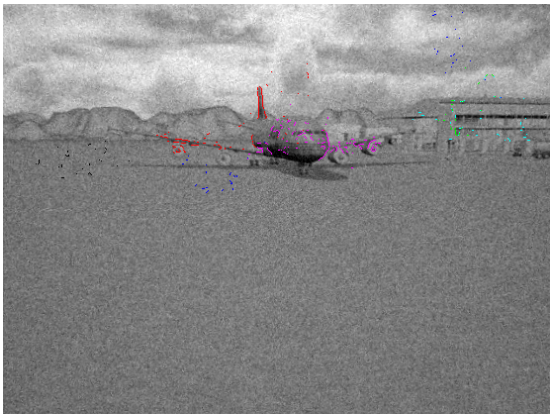
(b) $\sigma = 3$ intensity levels



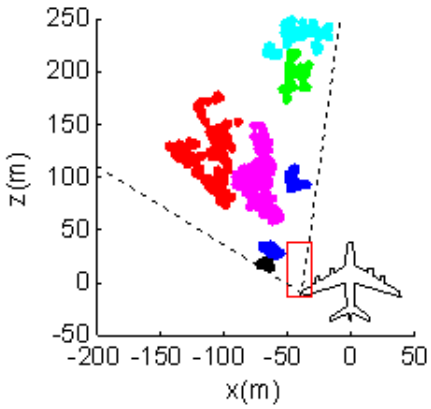
(c) $\sigma = 10$ intensity levels



(d) $\sigma = 10$ intensity levels

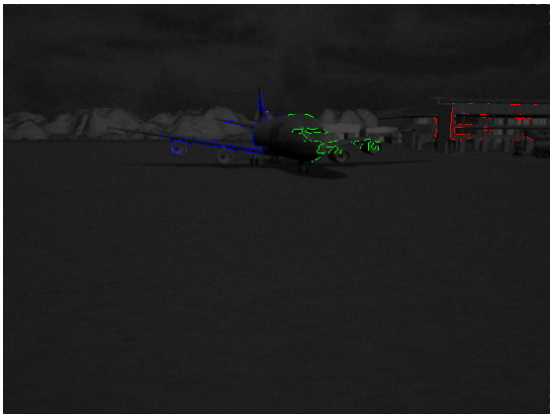


(e) $\sigma = 20$ intensity levels

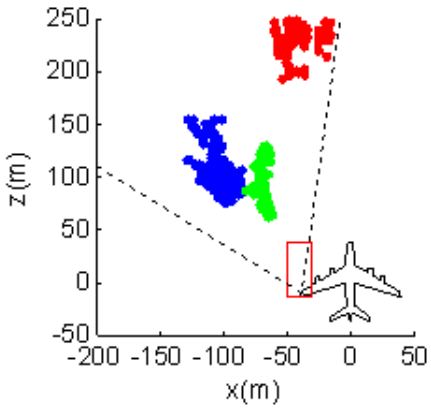


(f) $\sigma = 20$ intensity levels

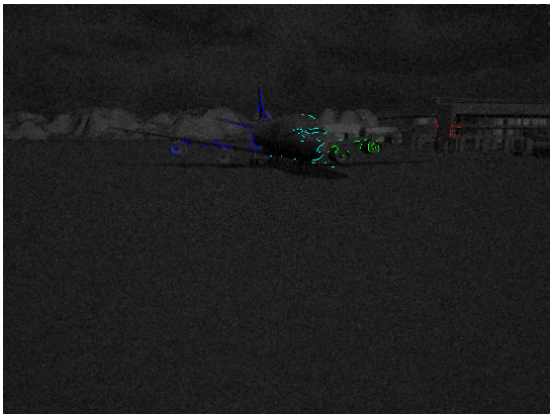
Figure 7.6: Obstacle detection under good illumination (day) for different values of image noise standard deviation σ (Example 2)



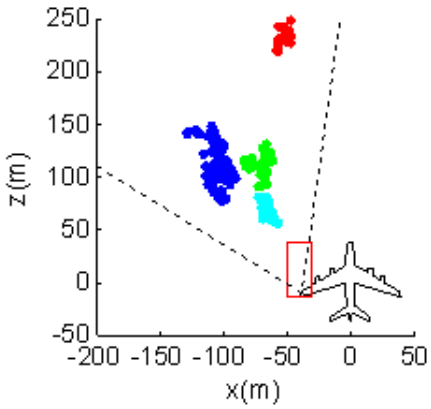
(a) $\sigma = 3$ intensity levels



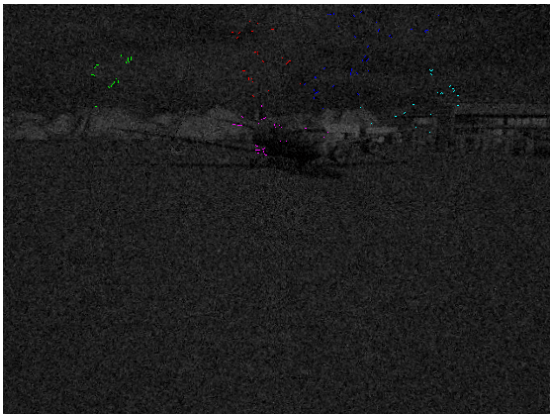
(b) $\sigma = 3$ intensity levels



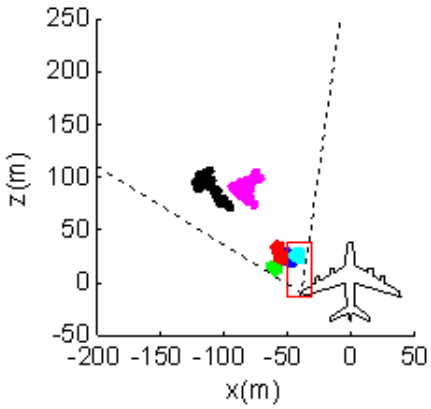
(c) $\sigma = 10$ intensity levels



(d) $\sigma = 10$ intensity levels



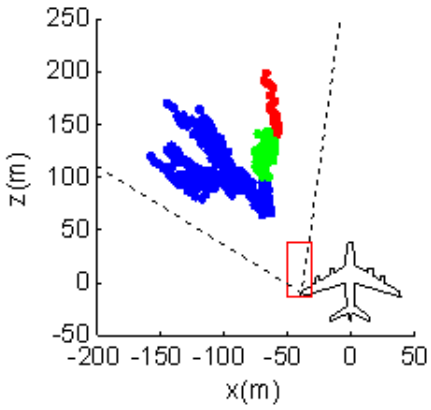
(e) $\sigma = 20$ intensity levels



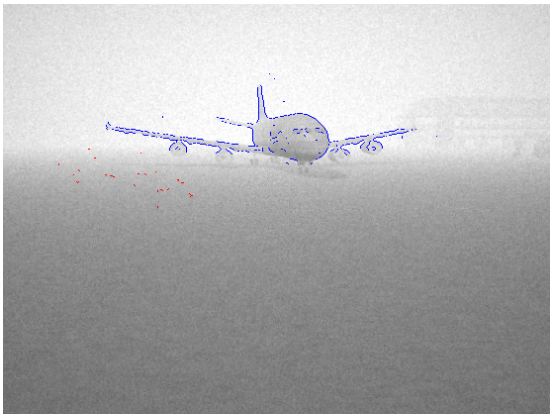
(f) $\sigma = 20$ intensity levels

Figure 7.7: Obstacle detection under low illumination (night) for different values of image noise standard deviation σ (Example 2)

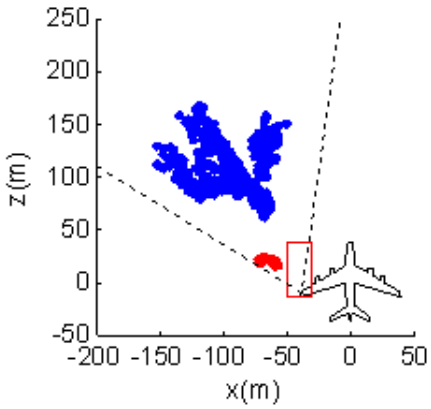
(a) $\sigma = 3$ intensity levels



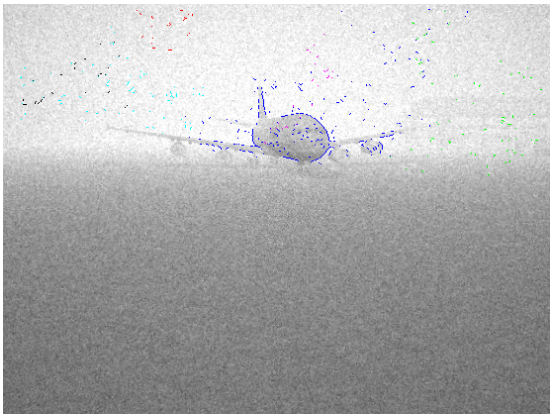
(b) $\sigma = 3$ intensity levels



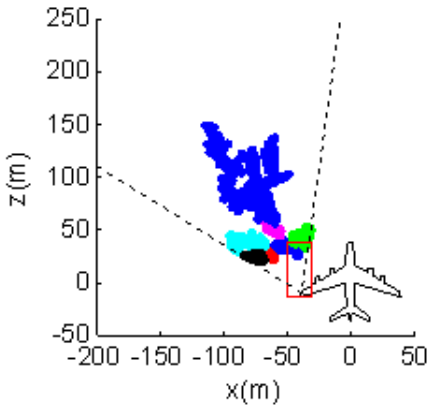
(c) $\sigma = 10$ intensity levels



(d) $\sigma = 10$ intensity levels

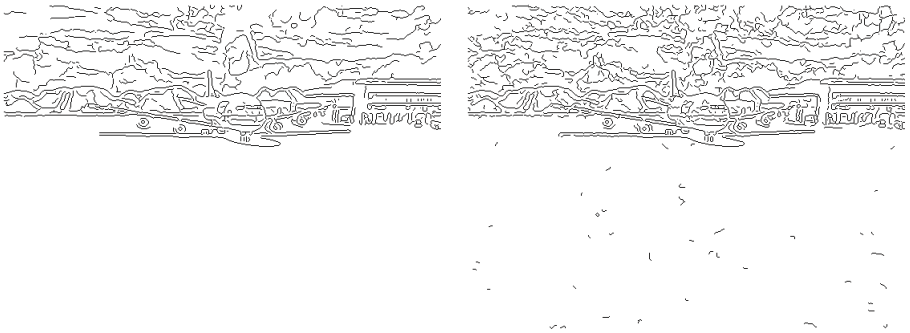


(e) $\sigma = 20$ intensity levels

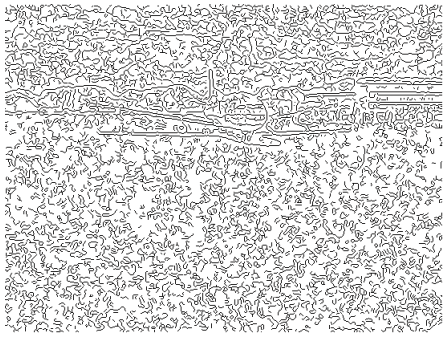


(f) $\sigma = 20$ intensity levels

Figure 7.8: Obstacle detection under low visibility (fog) for different values of image noise standard deviation σ (Example 2)



(a) $\sigma = 3$ intensity levels (b) $\sigma = 10$ intensity levels



(c) $\sigma = 20$ intensity levels

Figure 7.9: Edge detection under good illumination conditions and variable image noise

7.1.2.3 Tracking

Table 7.4 summarises the results of the tracking scenarios described in Appendix E.2. For a particular scenario, the greater the initial tracking distance and the number of tracked frames, and the lower the number of missed frames, the better the tracking is considered to be. Tracking is achieved in all cases when σ is either 3 intensity levels or 10 intensity levels. When σ is increased to 20 intensity levels, tracking is still successful in good illumination conditions. However, in one low visibility scenario and all of the low illumination scenarios, no tracking is achieved. As observed from Table 7.3, when the image noise is increased, the detection rate tends to decrease whereas the false detection rate tends to increase. This is particularly true in low illumination conditions, when σ is increased to 20 intensity levels. The increase in image noise makes tracking more challenging because 'true' obstacles are detected

less consistently and, even when they are detected, 'false' obstacles are more likely to be detected closer to the ownship's wingtip than the 'true' obstacles. Hence, in those scenarios where no tracking is achieved, the algorithm is constantly trying to track a different obstacle and never manages to track the same obstacle for more than a few consecutive frames.

Table 7.4: Tracking results

	Illumination/visibility	Day			Night			Fog		
	Noise σ (intensity levels)	3	10	20	3	10	20	3	10	20
Scenario 1	Initial tracking distance (m)	63	63	63	64	64	-	64	64	64
	Tracked frames	121	115	110	141	122	-	144	129	137
	Missed frames	14	12	19	9	29	-	10	22	16
	Total	135	127	129	150	151	-	154	151	153
Scenario 2	Initial tracking distance (m)	74	63	60	85	58	-	78	39	60
	Tracked frames	119	74	80	136	86	-	127	36	90
	Missed frames	3	25	14	3	1	-	1	15	4
	Total	122	99	94	139	87	-	128	51	94
Scenario 3	Initial tracking distance (m)	78	78	78	78	76	-	76	77	71
	Tracked frames	238	224	205	239	190	-	208	192	136
	Missed frames	7	10	33	6	15	-	4	34	20
	Total	245	234	238	245	205	-	212	226	156
Scenario 4	Initial tracking distance (m)	89	89	41	92	76	-	40	40	-
	Tracked frames	113	128	33	148	105	-	37	37	-
	Missed frames	23	10	6	2	6	-	1	2	-
	Total	136	138	39	150	111	-	38	39	-
Scenario 5	Initial tracking distance (m)	73	73	78	78	70	-	71	81	75
	Tracked frames	82	92	92	112	95	-	78	86	63
	Missed frames	27	18	29	8	9	-	24	35	27
	Total	109	110	121	120	104	-	102	121	90
Scenario 6	Initial tracking distance (m)	73	81	61	53	60	-	71	74	70
	Tracked frames	65	79	50	48	52	-	72	80	67
	Missed frames	2	11	4	1	2	-	1	3	2
	Total	67	90	54	49	54	-	73	83	69

It is expected that, the lower the level of image noise, the larger the number of tracked frames and the lower the number of missed frames that result during tracking. In fact, in the majority of the cases (61%), the least amount of missed frames and the largest quantity of tracked frames occur when σ is 3 intensity levels (the lowest

noise level used in the test cases).

In each of the tracking scenarios, the obstacle that is tracked is initially located outside the protection zone of the ownship and, in 94% of the successful tracking scenarios, the obstacle begins to be tracked outside this zone, at an initial tracking distance that exceeds 50m. The maximum initial tracking distance is 92m and is obtained in Scenario 4. This distance is close to the limit of what the tracking algorithm can achieve. Beyond this distance, the measurement accuracy is not good enough for the tracking algorithm to successfully track an obstacle. The fact that obstacles can be tracked outside the protection zone means that closure rate and Time to Collision (TTC) estimates are available before the obstacle potentially penetrates the protection zone. This is very important because, in the event of a potential collision, an alert needs to be generated before the obstacle enters the protection zone (the length and width of which are equal to the stopping distance and the minimum wingtip clearance, respectively, defined in Section 1.3) in order to ensure that the ownship can be stopped safely and that a collision is avoided. In other words, when an alert is generated, the TTC estimate needs to be greater than or equal to the time that is necessary to stop the ownship. Naturally, the TTC estimate needs to be sufficiently accurate in order to minimise false alerts and missed alerts.

Figures 7.11-7.15 show the distance, closure rate and TTC estimates obtained for Tracking Scenarios 1 and 6, and the distance and closure rate estimates obtained for Tracking Scenario 3. In Tracking Scenario 1, the ownship is initially taxiing on the ramp at 15kts and then turns left to park at a gate. The tracking algorithm tracks the right wingtip of a B747 that is parked to the left of the ownship. From Figure 7.11 it can be observed that, in general, the distance and closure rate estimates tend to become more accurate as the distance from the cameras decreases. It can also be observed that, up to around Frame 3015, the magnitude of the error in the closure rate estimate is equal to or greater than the actual closure rate (which is less than 2m/s). Due to this, the TTC estimates are very inaccurate and unreliable up to this point and are not shown in Figure 7.12. However, when the closure rate increases,

the error in the TTC estimates decreases significantly. When σ is 3 intensity levels and the illumination is good (Figure 7.12(a)), the TTC estimate settles within $\pm 1s$ of the actual TTC by Frame 3033. At Frame 3033, the distance between the tracked obstacle and the ownship's left wingtip is about 49m and the obstacle is just inside the protection zone. This implies that, in the event of a conflict, a reliable TTC estimate is available to generate an alert in time for the pilots to bring the ownship to a halt and avert a collision.[4]

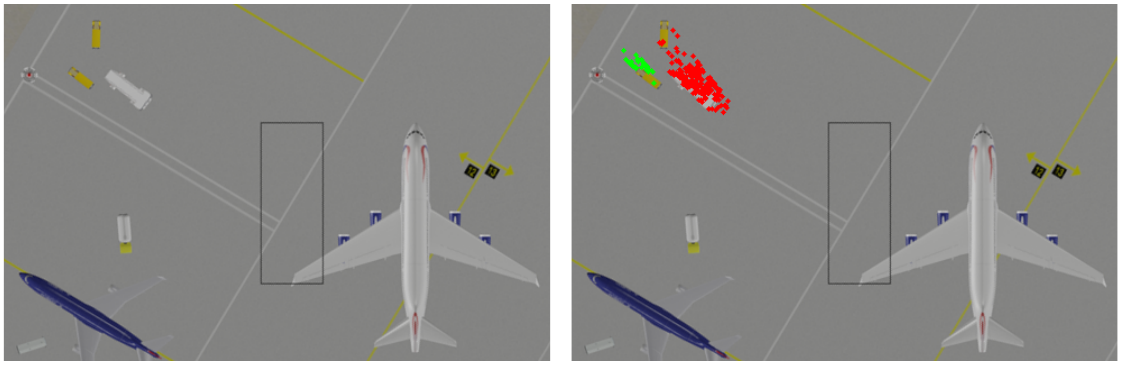
In Tracking Scenario 6 (Figures 7.13 and 7.14), the ownship is traveling on a taxiway at a speed of 25kts and a stationary A380 is located in front and to the left of the ownship, on a parallel taxiway. The tracking algorithm tracks the right wingtip of the A380. When σ is 3 intensity levels and the illumination is good (Figure 7.14(a)), the TTC estimates are all within $\pm 1s$ of the actual TTC values. The first TTC estimate is obtained in Frame 1019, when the distance between the tracked obstacle and the ownship's left wingtip is around 73m. As in the case of Tracking Scenario 1, this implies that, in the event of a potential collision, an alert can be reliably generated before the protection zone of the ownship is penetrated.

As observed in Figures 7.12 and 7.14, under low illumination conditions the TTC estimate takes longer (than under good light conditions) to settle within $\pm 1s$ of the actual TTC. This is because, as mentioned previously in the obstacle detection results, the positional accuracy of the system degrades in dark conditions, leading to greater errors in the distance and closure rate estimates. Similarly, when the image noise is increased for a particular illumination or visibility condition, the settling time of the TTC estimate also tends to increase because of a reduction in the positional accuracy of the system. Under low visibility conditions, the settling time of the TTC estimate is sometimes less than and sometimes greater than that obtained under good

[4]Note that the protection zone was defined for an aircraft traveling at 25kts whereas, in this scenario, the ownship is taxiing at 15kts. At this lower speed, the stopping distance would be smaller and any alert can be delayed until the obstacle gets closer to the ownship. In this research, the protection zone is assumed to be of constant size but, in practice, the protection zone can be adjusted dynamically depending on the speed of the ownship.

light conditions, implying that there is no significant difference between the tracking performance achieved under good light conditions and under low visibility conditions.

In Tracking Scenario 3 (Figure 7.15), the ownship is initially taxiing on the ramp at 15kts and then turns left to park at a gate. The tracking algorithm tracks passenger stairs that are situated at the gate. For most of the scenario, the actual closure rate is very low and, as in the case of Tracking Scenario 1, the magnitude of the error in the closure rate estimate is equal to or greater than the actual closure rate. As a result, the TTC estimates are very inaccurate and are therefore not presented here. From the distance profiles it can be observed that there is a bias in the estimated distance between the tracked obstacle and the ownship's left wingtip. This bias is present during the whole scenario (irrespective of the illumination, visibility and image noise conditions) and decreases with decreasing distance from the ownship's wingtip. The reason for this bias is understood by referring to Figure 7.10. The red cluster in Figure 7.10(b) corresponds to the passenger stairs. It can be observed that some of the points in this cluster are closer to (or further away from) the stereo cameras than the actual position of the stairs. Since the stairs are outside the protection zone, the tracking algorithm tracks the obstacle point that is closest to the protection zone boundary. Due to this, the algorithm tends to underestimate the distance between the ownship's left wingtip and the stairs.

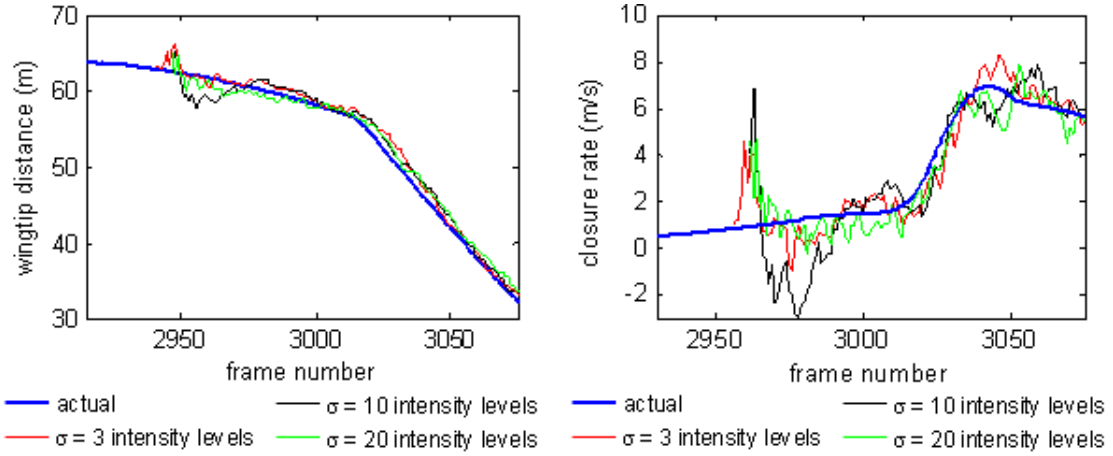


(a) Plan view (b) Obstacle points superimposed on plan view

Figure 7.10: Plan views corresponding to Frame 2950 in Tracking Scenario 3

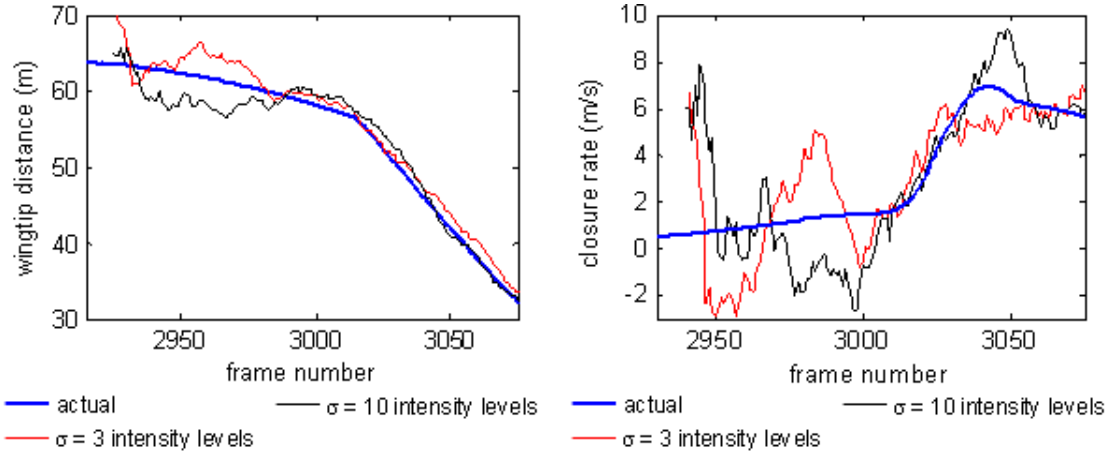
If only one type of obstacle was being tracked (such as vehicles) and each obstacle was being detected as a single cluster in each frame, then it would have been possible to track clusters from one frame to the next. The centroid of each cluster would have been used for tracking and the bias in the distance estimates would have been avoided. In this application, however, this approach would not work because of the many different types of obstacles that can be detected. As shown in the obstacle detection results presented in Chapter 5 and in this chapter, an obstacle can be detected as multiple clusters and some obstacles can also be combined into a single cluster. Therefore, it is not possible to track clusters from one frame to another. That is why the approach taken in this research was to track the obstacle point that is closest to the ownship's left wingtip or to the protection zone boundary.

The bias in the distance estimates is not observed in all of the tracking scenarios, implying that it is dependent on the type of obstacle being tracked. Any bias in the estimates is not corrected by the Kalman filter because this assumes that the measurement noise has a mean of 0. The bias observed in Tracking Scenario 3 can affect the performance of the system. Since the distance is underestimated, the TTC is also underestimated, potentially leading to an increase in the occurrence of false (nuisance) alerts. However, since the bias decreases as the obstacle approaches the cameras, the probability of such alerts is also expected to decrease in the vicinity of the ownship.



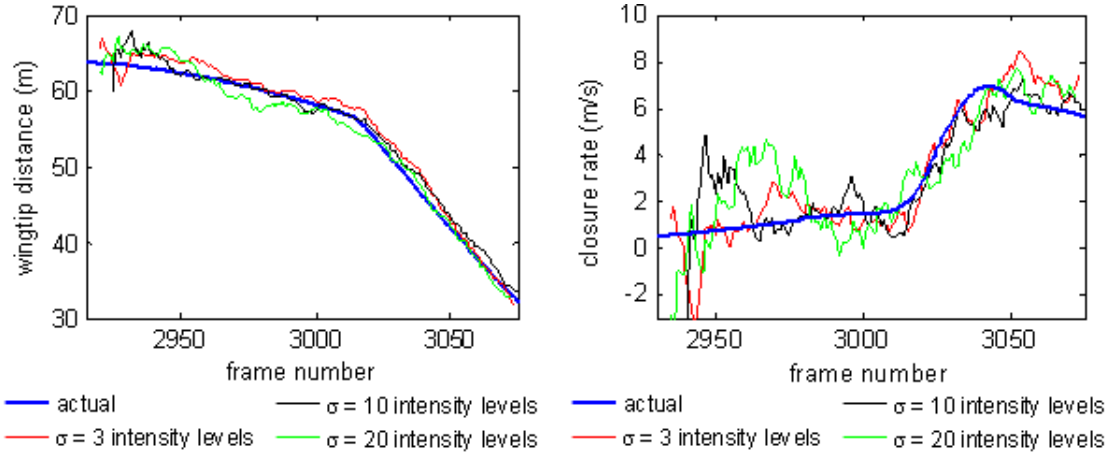
(a) Distance estimates in good illumination

(b) Closure rate estimates in good illumination



(c) Distance estimates in low illumination

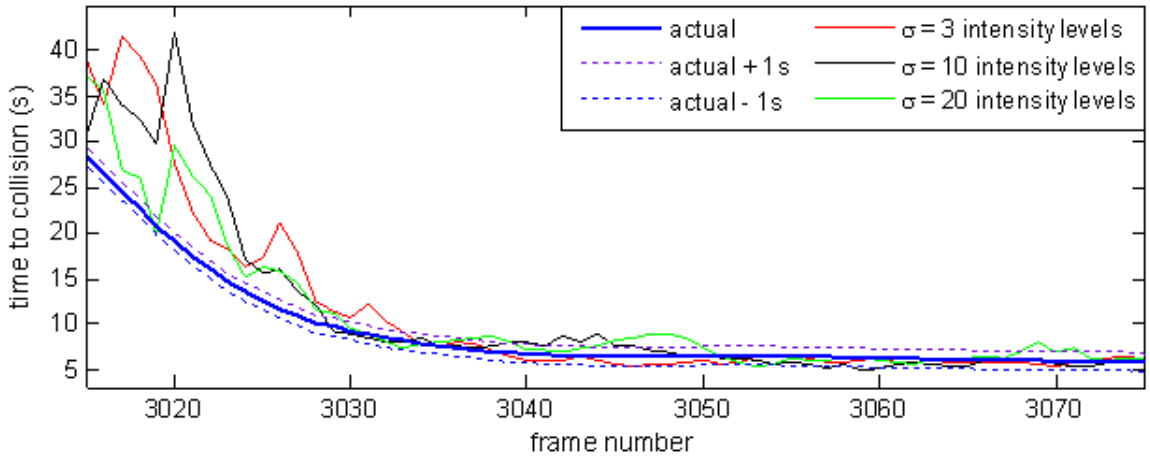
(d) Closure rate estimates in low illumination



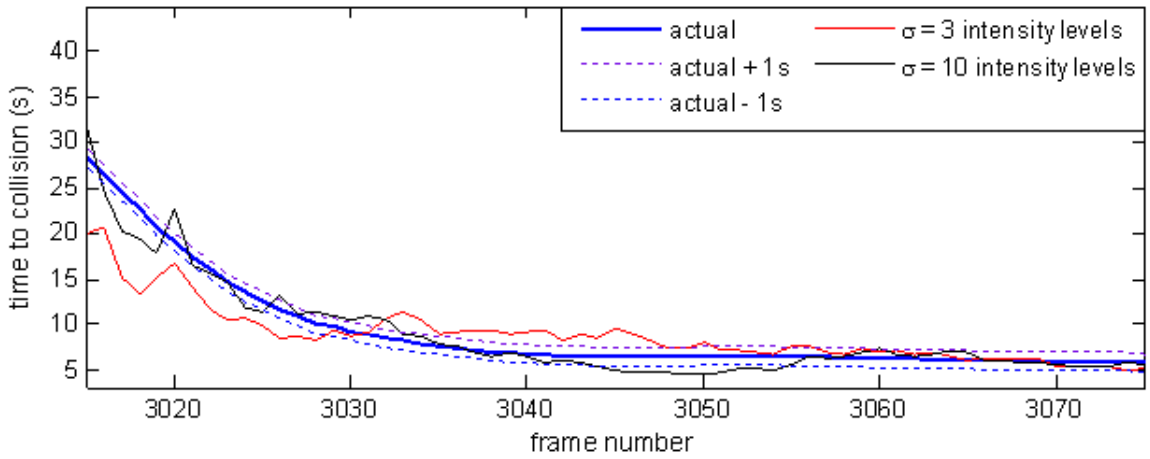
(e) Distance estimates in low visibility

(f) Closure rate estimates in low visibility

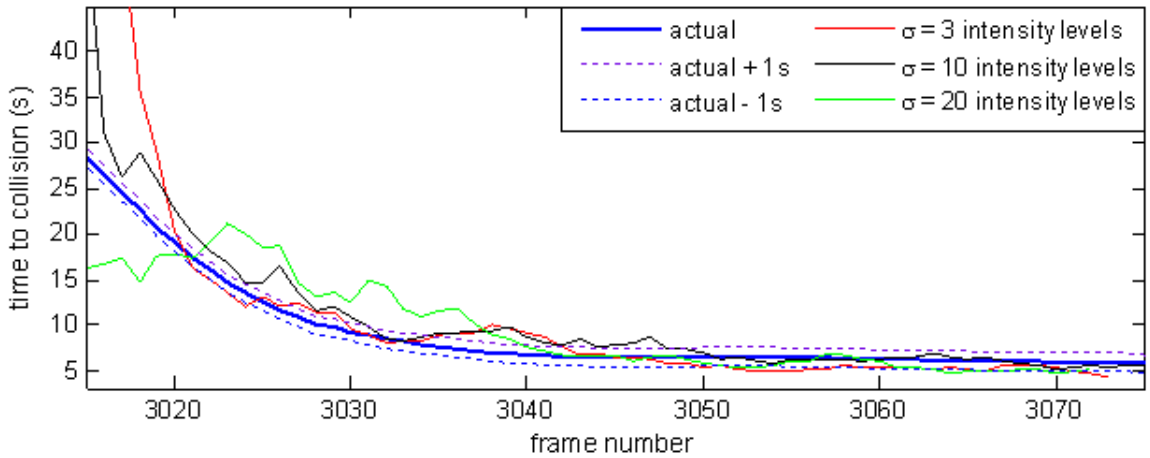
Figure 7.11: Distance and closure rate estimates obtained under different conditions during Tracking Scenario 1 (Refer to Table E.2 for scenario details)



(a) Time to collision estimation in good illumination

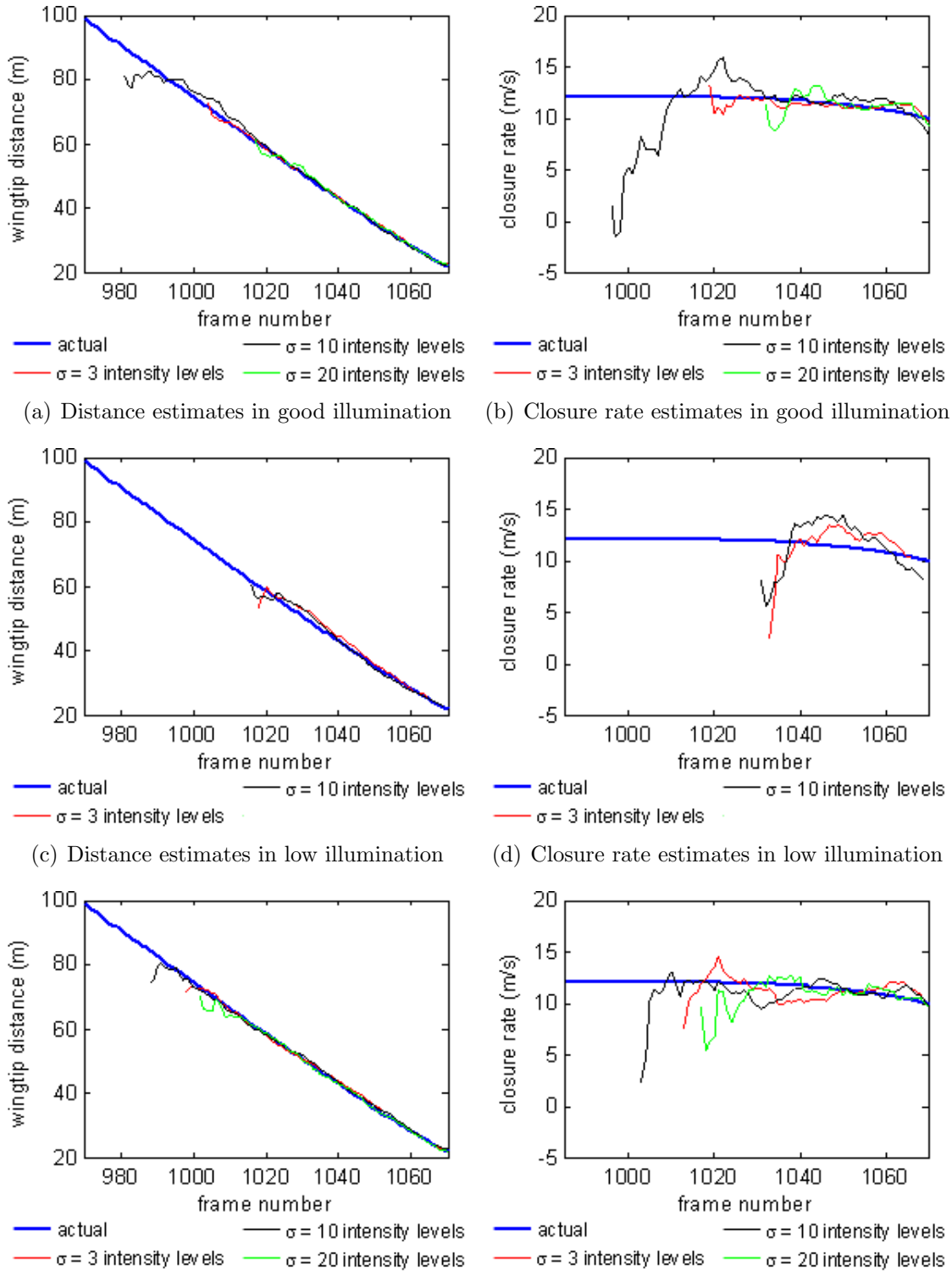


(b) Time to collision estimation in low illumination



(c) Time to collision estimation in low visibility

Figure 7.12: Time to collision estimates obtained under different conditions during Tracking Scenario 1



(a) Distance estimates in good illumination

(b) Closure rate estimates in good illumination

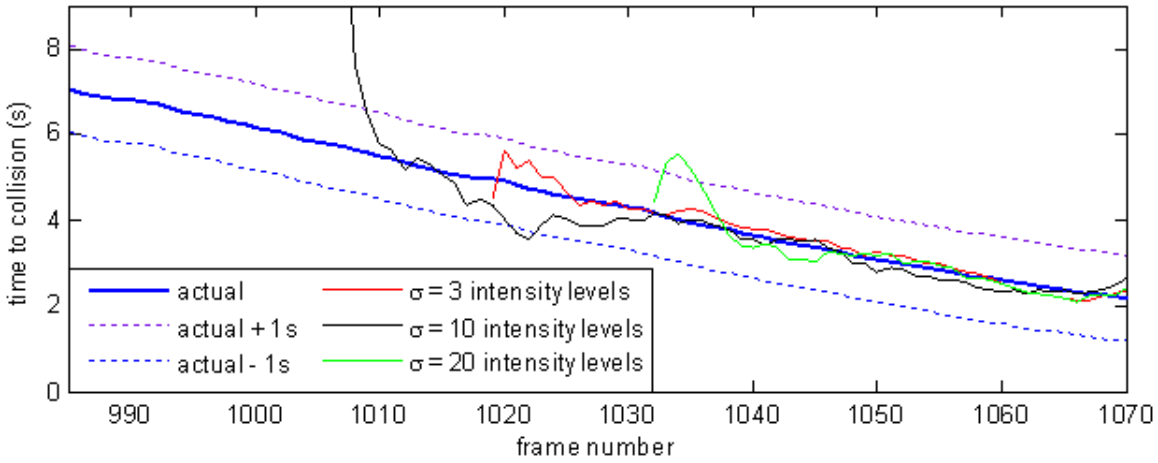
(c) Distance estimates in low illumination

(d) Closure rate estimates in low illumination

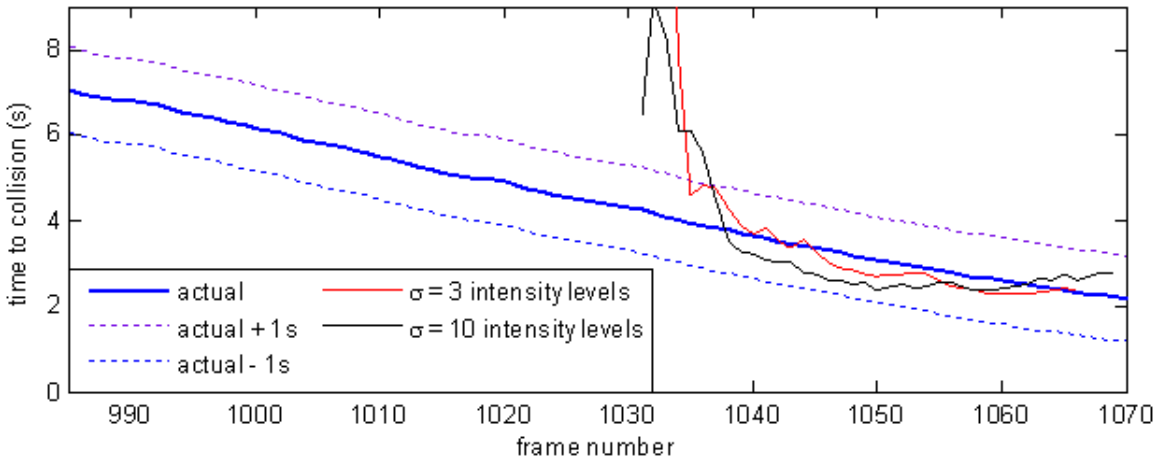
(e) Distance estimates in low visibility

(f) Closure rate estimates in low visibility

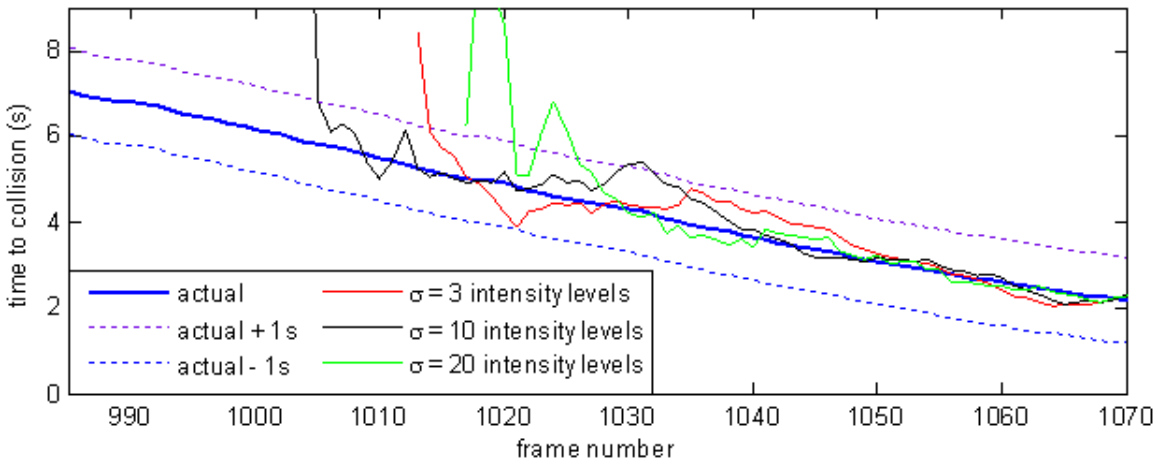
Figure 7.13: Distance and closure rate estimates obtained under different conditions during Tracking Scenario 6 (Refer to Table E.2 for scenario details)



(a) Time to collision estimation in good illumination

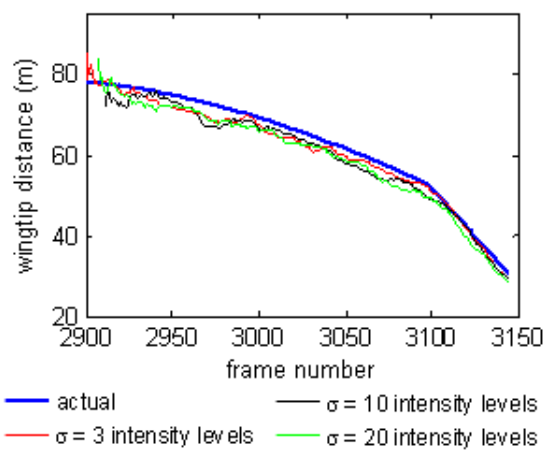


(b) Time to collision estimation in low illumination

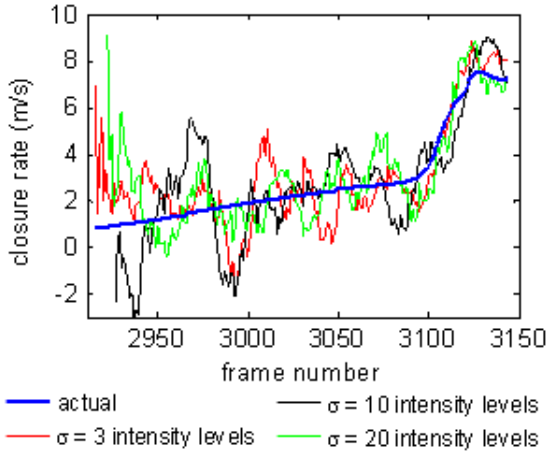


(c) Time to collision estimation in low visibility

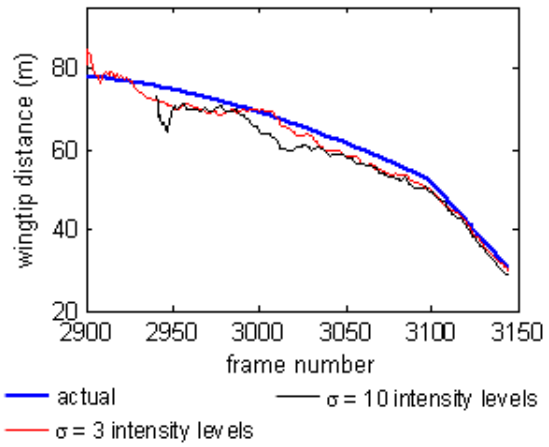
Figure 7.14: Time to collision estimates obtained under different conditions during Tracking Scenario 6



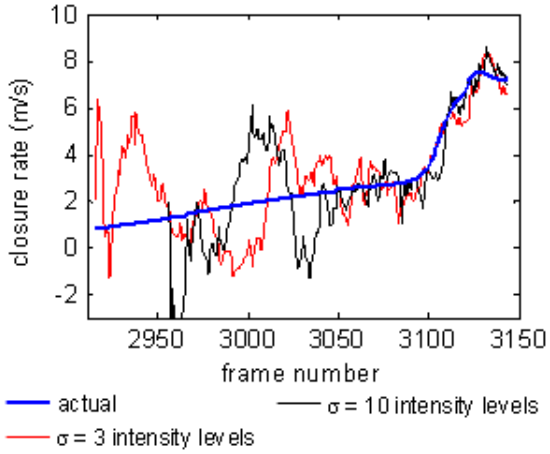
(a) Distance estimates in good illumination



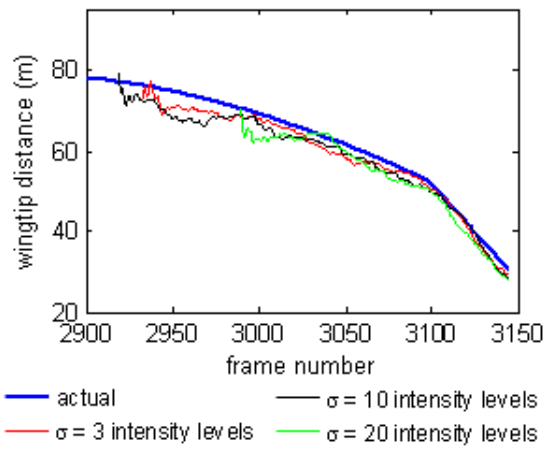
(b) Closure rate estimates in good illumination



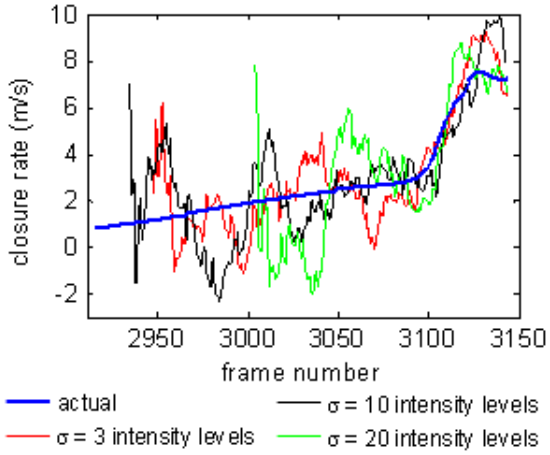
(c) Distance estimates in low illumination



(d) Closure rate estimates in low illumination



(e) Distance estimates in low visibility



(f) Closure rate estimates in low visibility

Figure 7.15: Distance and closure rate estimates obtained under different conditions during Tracking Scenario 3 (Refer to Table E.2 for scenario details)

7.2 Experiments with Real Images

7.2.1 Camera Setup and Calibration

The cameras used in these experiments were monochrome IEEE-1394a (FireWire) cameras with hardware synchronisation. A varifocal lens was attached to each camera. For the purpose of the airfield experiments, both cameras were set to a fixed focal length of approximately 3.6mm and the lens aperture was set to $F1.4$. The cameras were programmed to capture images with a resolution of 512x384 pixels at a frame rate of 10Hz. These settings were chosen after several unsuccessful attempts to capture higher-resolution images at a faster frame rate without losing any frames. The bottleneck was not caused by the cameras but by the computer hardware available to store the images (a laptop with an Intel® Pentium® M 1.73GHz processor and 1.5GB of RAM). The camera and lens specifications as well as information about camera synchronisation and image acquisition can be found in Appendix F.

The cameras were mounted on a horizontal aluminium bar and magnetically attached to the hood of a vehicle. Figures 7.16 and 7.17 show the test vehicle and stereo vision setup respectively. The cameras were deliberately positioned so as to point towards the left rather than in the direction of motion of the vehicle. This was done in order to replicate the orientation of the cameras in the simulated setup. After mounting the cameras on the vehicle, the stereo vision system was calibrated indoors using 10 images for intrinsic and relative extrinsic calibration (the same number of images as those used in the simulations) and 12 calibration targets for absolute extrinsic calibration.

The arrangement of the targets used for absolute extrinsic calibration can be seen in Figure 7.18. Due to the limited indoor space, the targets were confined to an area approximately 33m long and 13m wide. The origin of the WRF was defined as a point on the ground at the front of the vehicle, with the z and x axes aligned with the longitudinal and lateral axes of the vehicle respectively, and with the y axis pointing vertically downwards below the ground surface, as shown in a closeup of the test

vehicle in Figure 7.19. The 3D coordinates of the control points on the calibration targets were found by taking repeated readings using a measuring tape. Then, the 2D pixel coordinates of the control points and their 3D coordinates (expressed in the WRF) were input to the absolute extrinsic calibration algorithm.

Figure 7.16: Test vehicle

Figure 7.17: Camera setup

7.2.2 Design of Experiment

The main objectives of the experiments with the real cameras were: (a) to assess the generic obstacle detection and tracking capabilities of the system and (b) to estimate the temporal image noise of each camera. In order to meet these objectives, a number

Figure 7.18: Absolute extrinsic camera calibration setup

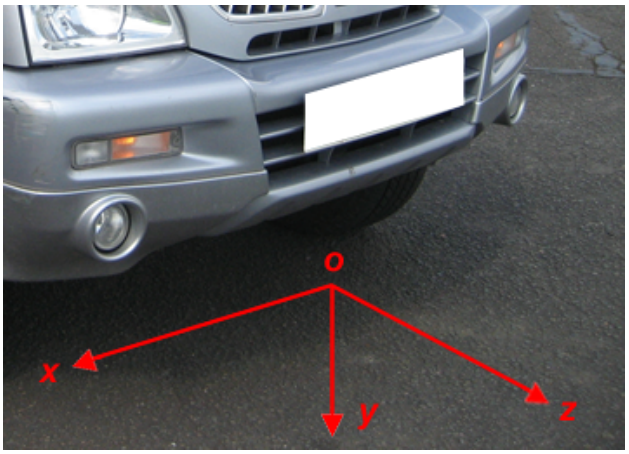


Figure 7.19: The World Reference Frame (WRF)

of outdoor image sequences were recorded at Cranfield Airport using the test vehicle and the calibrated camera setup. These recordings were made early in the afternoon. The weather was partly cloudy but the visibility was good.

7.2.2.1 Obstacle detection and tracking

In order to check the obstacle detection and tracking capabilities of the system, 8 image sequences were recorded at different areas of the airport, including the ramp and the taxiways. Three main types of obstacles were captured in these sequences: aircraft (such as Jetstream, Diamond, Cessna and Piper aircraft), vehicles (such as fuel trucks, fire engines and vans) and buildings (such as hangars and office blocks). In most of the image sequences, the obstacles were initially far away from the test

vehicle in order to be able to determine the detection range of the system. Also, in most of the image sequences, either the test vehicle or the obstacles were moving. A description of all of the image sequences captured at the airport is provided in Appendix G.

To study the tracking performance of the system, the first four image sequences were used. In three of these sequences, the system tracks an aircraft extremity (such as a wingtip or the nose) whereas, in the remaining sequence, a vehicle is tracked. The same tracking logic was applied as in the experiments using synthetic images. Also, the following parameters were recorded for each sequence: (a) the initial tracking distance, (b) the number of tracked frames, (c) the number of missed frames and (d) the total length of the tracking sequence.

7.2.2.2 Estimation of temporal image noise

In order to be able to estimate the temporal image noise of each camera, the test vehicle was kept fixed and image sequences of five static scenes were recorded, each a 100 frames long. These sequences were captured at different areas of the airport.[5] For each sequence, the average image was found and, under the assumption that the image noise had a mean of 0, this image was considered to be the noiseless image. Then, the image noise was estimated by finding the difference between each image in the sequence and the average image. The standard deviation σ of the noise in each image was then calculated. Finally, the average standard deviation $\bar{\sigma}$ of the image noise was found for the whole image sequence.

In total, around 6600 frames were recorded at the airport. These were then processed offline. The height threshold for obstacle detection was reduced from 1m to 0.5m because the change in camera height due to ground roughness was expected to be less than that due to wing flexing in the simulations. The rest of the algorithmic

[5]The type of scene was not important for this experiment because the temporal image noise is only dependent on the inherent properties of the camera sensors.

parameters were left the same as for the experiments with the synthetic images.

7.2.3 Results

7.2.3.1 Calibration

Tables 7.5-7.7 contain the calibration results of the stereo vision system. From Table 7.5 it can be observed that the cameras have similar intrinsic parameters. Their lenses have an average focal length of 389.4 pixels, which is less than that of the simulated cameras (554.3 pixels). The impact of the difference in focal length on the performance of the system is discussed in detail in Section 7.3. From Table 7.6 it can be observed that the horizontal distance T_x between the cameras is around 1.5m, which is equal to the baseline distance of the simulated camera setup. It can also be noticed that the cameras are not perfectly aligned with each other because the rest of the relative extrinsic parameters are not equal to 0. As explained in Chapter 4, this misalignment is corrected during rectification.

Table 7.5: Intrinsic calibration values of the optical setup

Calibration parameters		Left camera	Right camera
Focal length (pixels)	f_x	392.47 ± 5.65	384.03 ± 4.71
	f_y	393.79 ± 3.87	387.19 ± 3.30
Principal point (pixels)	c_x	261.66 ± 6.27	247.98 ± 5.72
	c_y	205.47 ± 5.47	199.76 ± 5.17
Skew	α	0 ± 0	0 ± 0
Lens distortion coefficients	k_1	-0.37 ± 0.02	-0.38 ± 0.04
	k_2	0.16 ± 0.01	0.21 ± 0.05
	k_3	0 ± 0	0 ± 0
	k_4	0 ± 0.01	0 ± 0.01
	k_5	0 ± 0	0 ± 0

Table 7.6: Relative extrinsic calibration values of the optical setup (The calibration values are expressed in the right CRF)

Calibration parameters		Stereo
Rotation (°)	θ	-2.72 ± 1.05
	ϕ	2.99 ± 1.18
	ψ	-0.25 ± 0.26
Translation (mm)	T_x	-1499.78 ± 6.83
	T_y	30.18 ± 4.83
	T_z	88.90 ± 23.59

Table 7.7: Absolute extrinsic calibration values of the optical setup (The calibration values for the left and right camera are expressed in the left and right CRF respectively)

Calibration parameters		Left camera	Right camera
Rotation (°)	θ	-3.74	-3.54
	ϕ	-24.56	-24.87
	ψ	-1.74	-1.47
Translation (m)	T_x	0.92	-0.56
	T_y	1.30	1.30
	T_z	0.25	0.30

7.2.3.2 Temporal image noise

Table 7.8 shows the average standard deviation $\bar{\sigma}$ of the noise in the left and right image sequences for each of the static scenes considered. For each scene, there is very little difference between the image noise estimates of the left and right image sequences. This is to be expected since the left and right cameras have exactly the same specifications. For each camera, the small changes in $\bar{\sigma}$ from one scene to another are due to the finite length of the image sequences, which means that the noiseless image of each sequence could only be approximated. Better noise estimates would have therefore been obtained with longer image sequences.

The maximum average noise standard deviation over all the image sequences is less than 3 intensity levels, which is less than the minimum level of temporal image noise added to the synthetic images. The temporal image noise is likely to vary

slightly depending on different factors such as the frame rate, the image resolution and the illumination conditions.

Table 7.8: Temporal image noise estimates for the left and right cameras (The image sequences used for this experiment were captured under good illumination and visibility conditions. The camera and lens specifications are provided in Tables F.1 and F.2 respectively.)

Image sequence	1	2	3	4	5
$\bar{\sigma}$ of left image sequence noise (intensity levels)	1.92	1.43	1.92	2.15	1.77
$\bar{\sigma}$ of right image sequence noise (intensity levels)	2.12	1.61	2.20	2.55	1.91

7.2.3.3 Obstacle detection

Figures 7.20-7.22 show the results of the obstacle detection experiments for the three main obstacle categories: aircraft, vehicles and buildings. In the left column of each figure, the obstacle points are superimposed over the left intensity image whereas, in the right column, they are plotted in the WRF. Each obstacle cluster is represented by a different colour such that it is possible to match each cluster in an intensity image with its corresponding cluster in the WRF. The dotted lines in the WRF represent the boundaries of the common FOV of the stereo vision system.

The closer the obstacles are to the origin of the WRF, the more the clusters are representative of the actual obstacles they correspond to. This is because the triangulation uncertainty decreases closer to the cameras and, therefore, the positional accuracy improves. On the other hand, as the distance between the obstacles and the cameras increases, the clusters become less compact and start losing their distinctive shapes. For example, in Figure 7.20(d) it can be observed that the shape of the pink cluster correlates well with the actual shape of the left wing of the aircraft detected in Figure 7.20(c). In Figure 7.20(a), all of the extremities of the closest aircraft are detected and the shape, size and orientation of the aircraft are quite clear from the four clusters representing it in the WRF in Figure 7.20(b). In contrast, even though

the extremities of the other aircraft (the one that is further away from the test vehicle) are detected, the red cluster associated with it in Figure 7.20(b) is less representative of the actual aircraft. For example, due to the increased positional errors, the obstacle points are more dispersed in the WRF and the aircraft appears to be larger than it actually is.

The variation of the positional accuracy and the shape of the clusters with range is also evident in Figures 7.21 and 7.22. For instance, from Figures 7.21(c) and 7.21(d) it can be observed that the shape, size and orientation of the green cluster representing the fire engine are a good representation of the actual obstacle. On the other hand, the shape of the red cluster representing the minivan (which is further away from the test vehicle) is less compact and less defined. In Figure 7.22, the orientation of the buildings and hangars is quite clear from the projection of the obstacle points in the WRF. This correlates well with what can be inferred from the corresponding intensity images. In Figure 7.22(a), some pixels (shown in blue) corresponding to the building on the right of the image are correctly detected as obstacle points. However, due to errors in their disparities, they are projected onto an area of the WRF in Figure 7.22(b) which is far away from their actual location.

From the plots of the obstacle points in the WRF, it can be observed that the maximum detection range of the system is about 50m. The reason for this limit is discussed in further detail when comparing the results obtained with the synthetic images and the real images. As expected, the obstacles that are easiest to detect at long distance are buildings (due to their size) whereas the obstacles that are hardest to detect are the wings of light aircraft (due to their size and aspect ratio). For example, in Figure 7.20(c), the right wing of the aircraft is not detected. Similarly, in Figure 7.20(e), the left wingtip of the closest aircraft is missed. In both cases, the wings are detected by the edge detector but are rejected during correspondence and clustering. The way to improve the detection of these obstacles (without changing the optical setup) is to relax the filtering mechanisms used in these algorithms, so as to retain more obstacle points. However, this will result in more false detections.

False detections result when ground or sky features are mapped onto incorrect 3D positions above the ground, over the height threshold used to classify obstacle points. This is mainly due to correspondence errors and the uncertainty of the calibration parameters. Some false obstacles are mapped onto positions outside the protection zone whereas others are mapped onto positions inside the zone. Figure 7.23 shows two instances of false detections. In Figures 7.23(a) and 7.23(b), some ground features (represented by the cyan cluster) are classified as an obstacle within the protection zone. Similarly, in Figures 7.23(c) and 7.23(d), some sky features (represented by the green cluster) are classified as an obstacle inside the protection zone. After all of the real images were processed, it was found that false detections occurred in 10% of the frames. Naturally, the impact of false detections is that they can potentially lead to more false alerts. This is because, during tracking, if the false obstacles are detected closer to the cameras than the true obstacles, the algorithm will attempt to track them instead of the true obstacles.

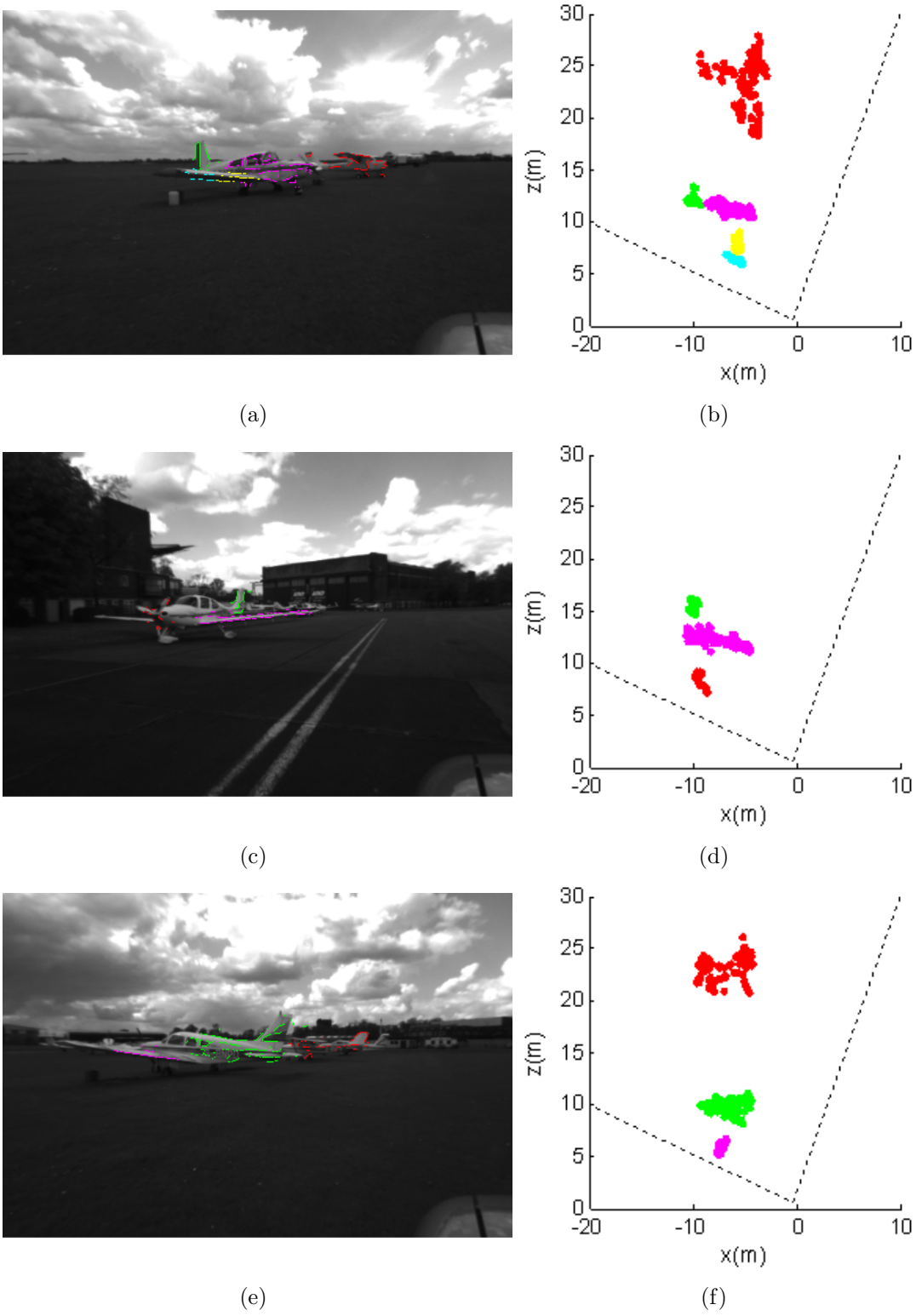


Figure 7.20: Obstacle detection (aircraft): obstacle pixels superimposed on intensity image (left), obstacle points in WRF (right)

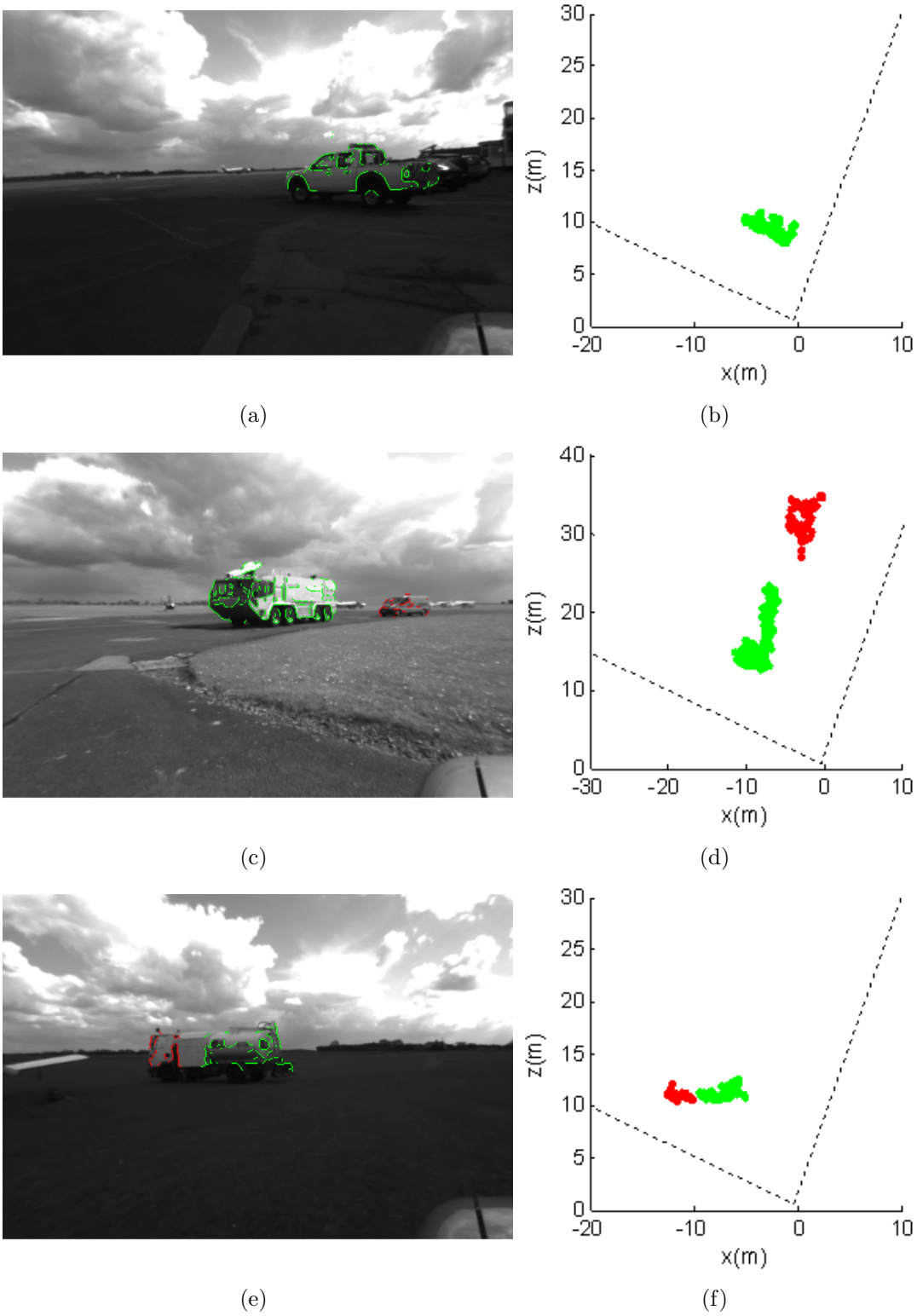


Figure 7.21: Obstacle detection (vehicles): obstacle pixels superimposed on intensity image (left), obstacle points in WRF (right)

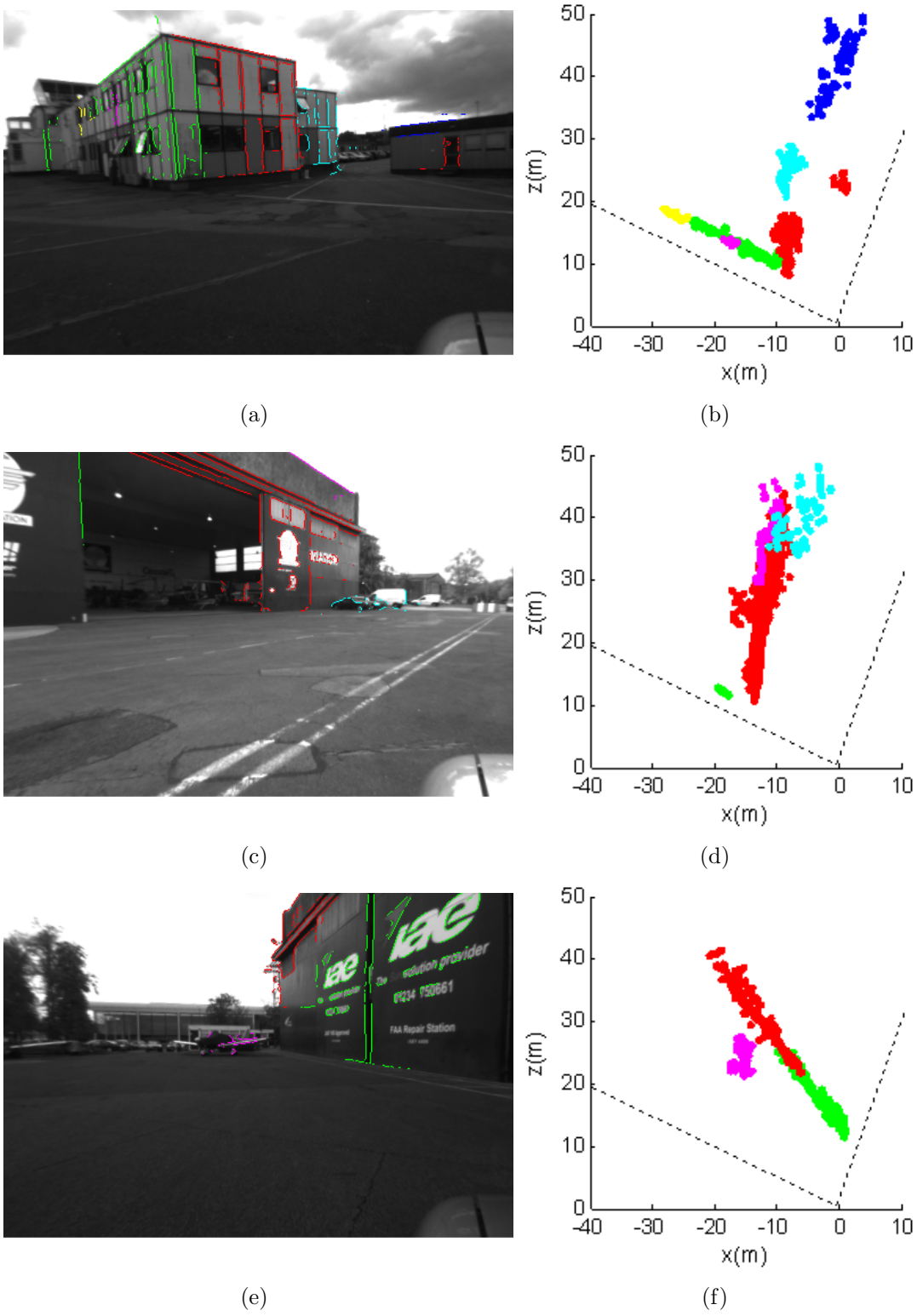
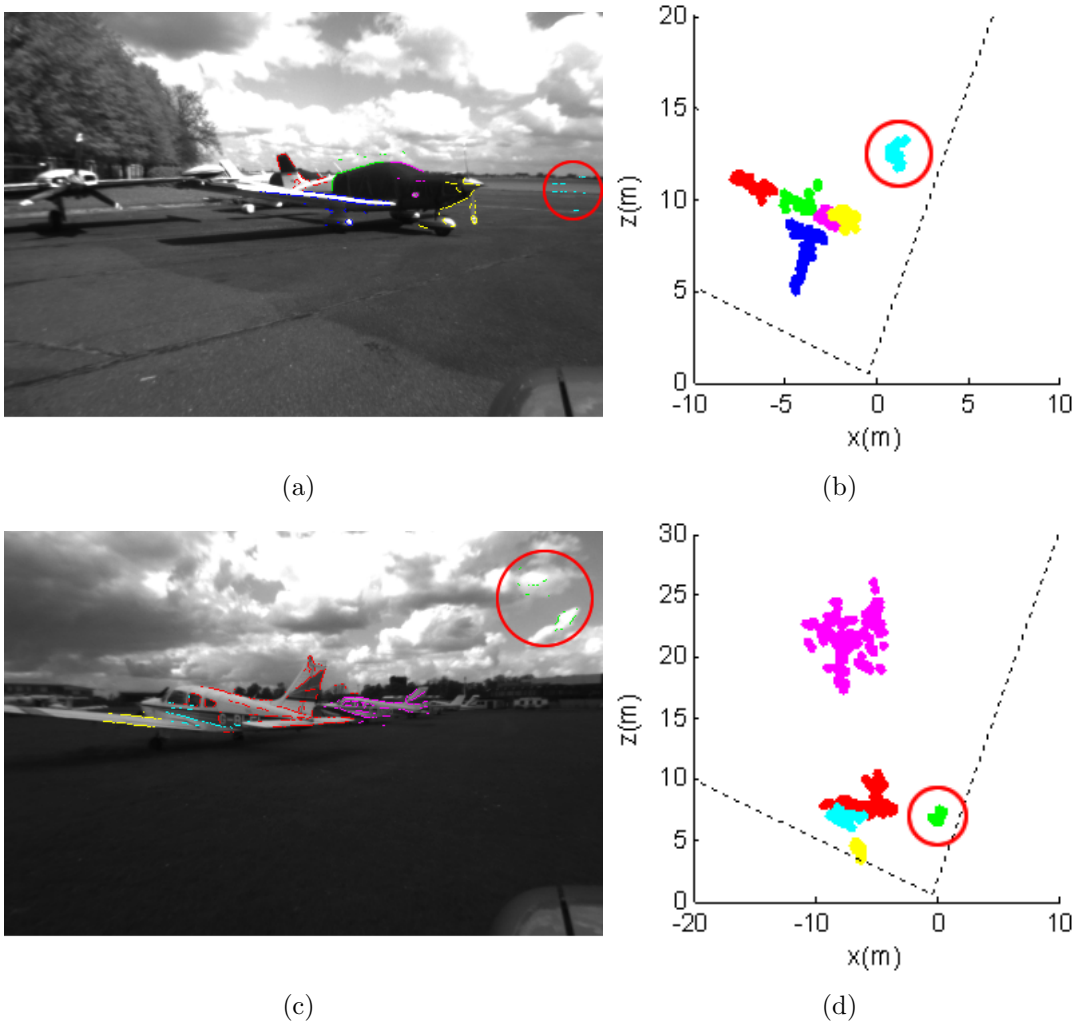


Figure 7.22: Obstacle detection (buildings): obstacle pixels superimposed on intensity image (left), obstacle points in WRF (right)



(a) (b)

(c) (d)

Figure 7.23: Incorrect detection of ground features (a,b) and sky features (c,d) (Areas of incorrect detection are enclosed in circles)

7.2.3.4 Tracking

Figures 7.24-7.26 show the tracking results obtained for Image Sequences 1, 2 and 4 and Table 7.9 summarises the tracking results. In Image Sequence 1 (Figure 7.24), the test vehicle is driven along the centre of a taxiway towards a Lightning aircraft that is parked at the edge of the taxiway, to the left of the vehicle. The nose of the aircraft is selected for tracking until it exits the common FOV of the stereo cameras at the end of the sequence. The position history of the tracked obstacle (relative to the test vehicle) is shown in Figure 7.24(b) and corresponds to the movement of the test vehicle during the image sequence.

In Image Sequence 1, there are 8 frames in which the tracked obstacle goes missing. Some of these frames (Frames 96, 97 and 100) can be identified by 'spikes' in the plots of the x and z coordinates of the tracked obstacle point (Figures 7.24(c) and 7.24(d)). In some of the frames, the reason why the obstacle goes missing is that the nose of the aircraft is not detected. In the other frames, some ground features are detected as obstacles and are mapped onto positions in the WRF that are closer to the cameras than the nose of the aircraft. In all of these frames, the distance between the measurements and the estimated position of the aircraft's nose exceeds a certain threshold and, therefore, the tracking algorithm assumes that the obstacle is missing. In these cases, the algorithm still tracks the aircraft's nose by ignoring the measurements and relying on the predictions of the Kalman filter.

In Image Sequence 2 (Figure 7.25), the test vehicle is initially stationary on one side of a taxiway. Then it is driven towards a stationary light aircraft that is situated ahead, on the other side of the taxiway. During most of the image sequence, the nose cone of the aircraft is selected for tracking. Then, when the distance between the aircraft and the test vehicle falls below 10m, the left wingtip of the aircraft is selected for tracking because it becomes closer to the cameras than the nose. The plots shown in Figures 7.25(b)-7.25(f) correspond to the tracking of the nose cone.

There are 9 missed frames in Image Sequence 2. As in the previous example, some of these frames (Frames 33 and 43) can be identified by spikes in Figures 7.25(c)

and 7.25(d). In most of these frames, the reason for which the tracked obstacle goes missing is that the nose cone is not detected. The obstacle is still tracked successfully by estimating its position from the Kalman filter predictions.

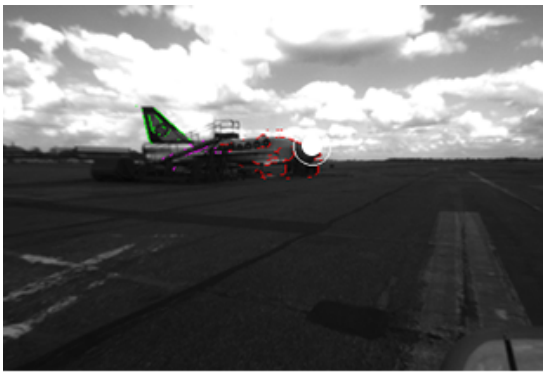
In Image Sequence 4 (Figure 7.26), the test vehicle is parked at the edge of a taxiway while two vehicles (a fire engine and a minivan) pass in front of it. In this case, the tracking algorithm tracks the front end of the fire engine until it exits the common FOV of the stereo cameras. No missed frames occur during this tracking sequence.

From the distance profile and the obstacle trajectory of each of the image sequences used for tracking, it can be observed that an obstacle begins to be tracked inside the protection zone. In the case of Image Sequences 1 and 2, this obstacle is well within the common FOV of the stereo vision system at the beginning of the sequence, at a distance from the origin that is much larger than the initial tracking distance. On the other hand, in Image Sequences 3 and 4, an obstacle enters the common FOV of the system and crosses the protection zone boundary at the beginning of the image sequence.[6] Naturally, the impact of tracking obstacles only after they enter the protection zone is that, in the event of a potential collision, alerts will be delayed. Then, when an alert is eventually generated, the TTC might be less than the time required to stop the vehicle and avoid a collision.

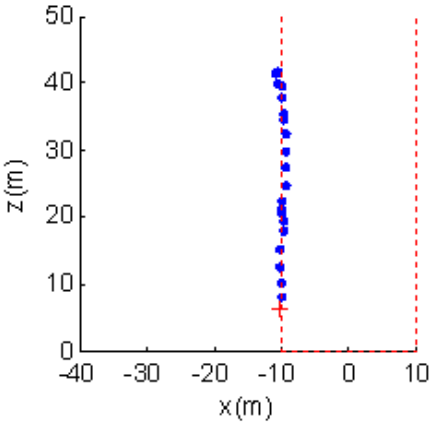
Table 7.9: Tracking results

Image sequence	1	2	3	4
Initial tracking distance (m)	42	28	16	29
Missed frames	8	9	2	0
Tracked frames	96	116	53	68
Total	104	125	55	68

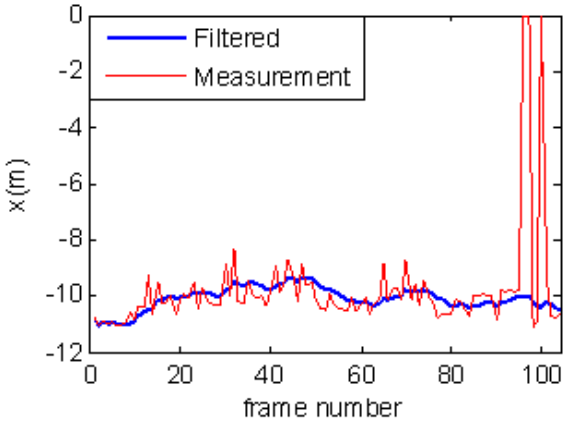
[6] If the cameras are mounted on a large commercial aircraft as proposed in this research, the situations where an obstacle enters the common FOV of the stereo setup and penetrates the protection zone at the same time, will be highly unlikely. In the most common conflict scenarios, obstacles will enter the common FOV before they can cross the protection zone boundary.



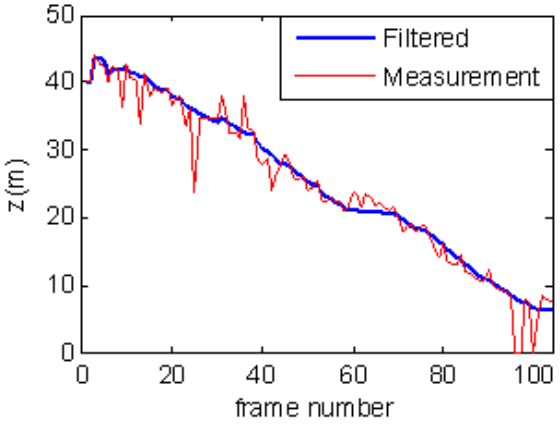
(a) Obstacle pixels superimposed on intensity image



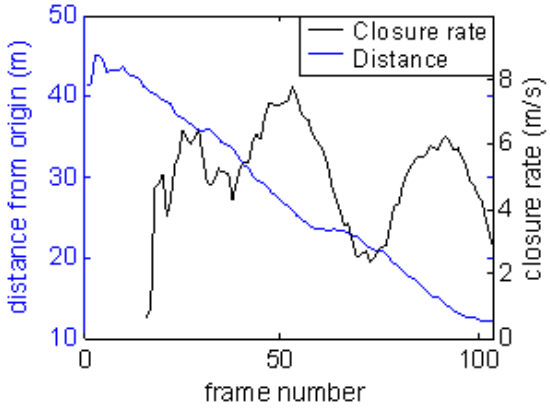
(b) Obstacle trajectory



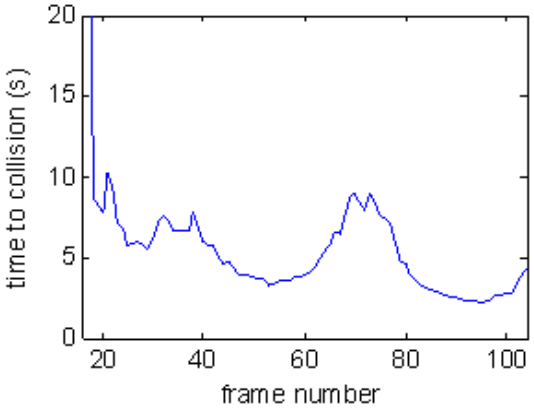
(c) x coordinate of tracked obstacle point in WRF



(d) z coordinate of tracked obstacle point in WRF

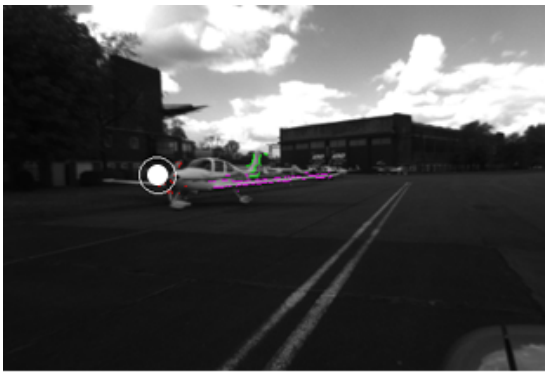


(e) Distance from origin and closure rate

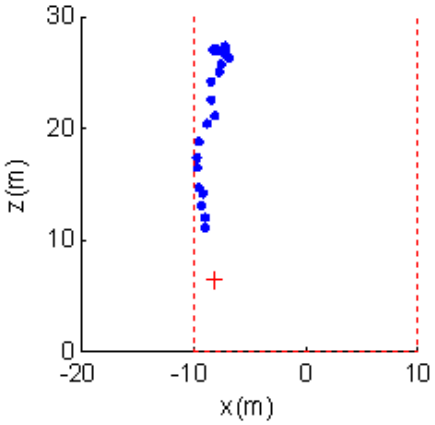


(f) Time to collision

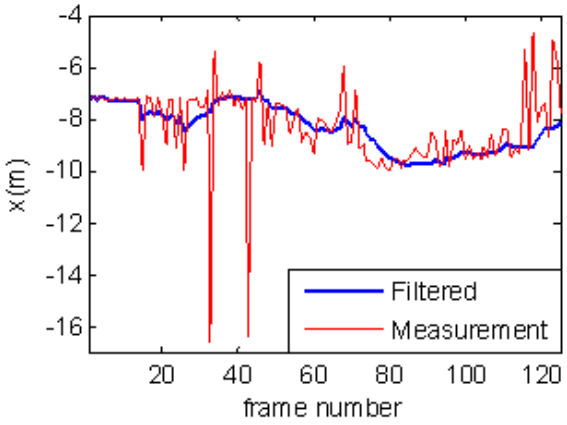
Figure 7.24: Tracking (Image Sequence 1)



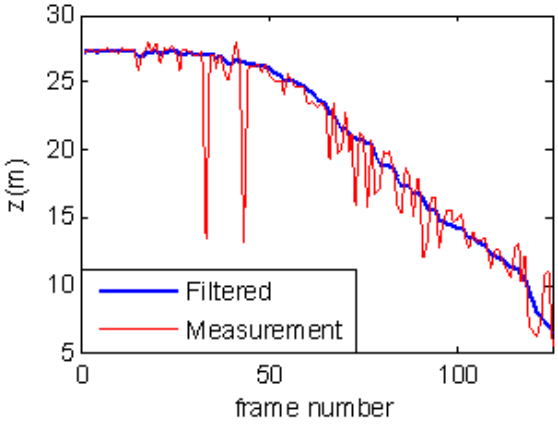
(a) Obstacle pixels superimposed on intensity image



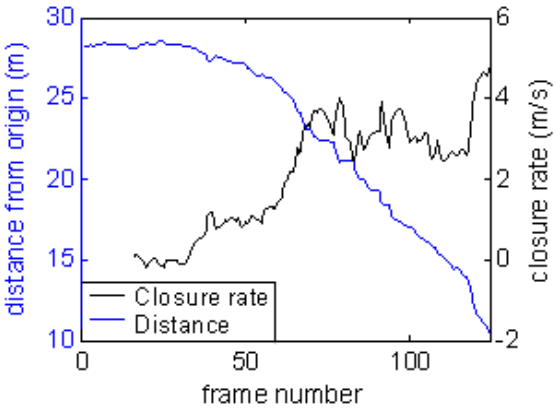
(b) Obstacle trajectory



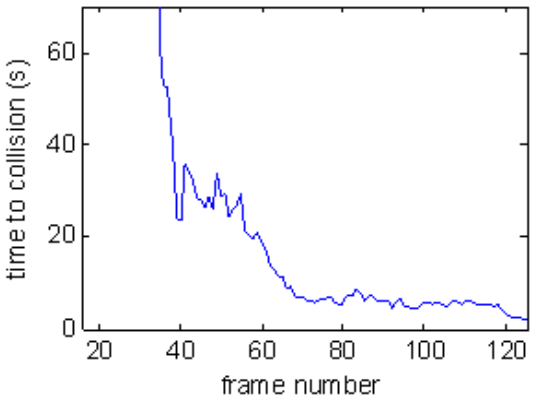
(c) x coordinate of tracked obstacle point in WRF



(d) z coordinate of tracked obstacle point in WRF



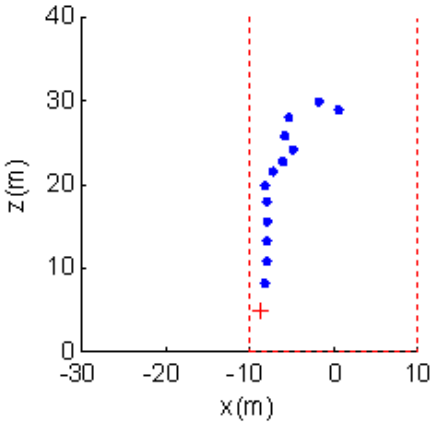
(e) Distance from origin and closure rate



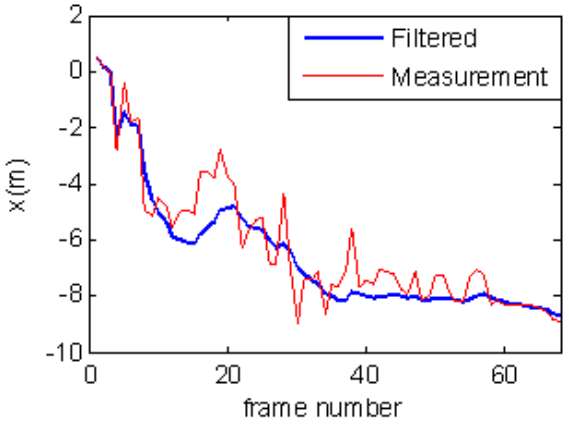
(f) Time to collision

Figure 7.25: Tracking (Image Sequence 2)

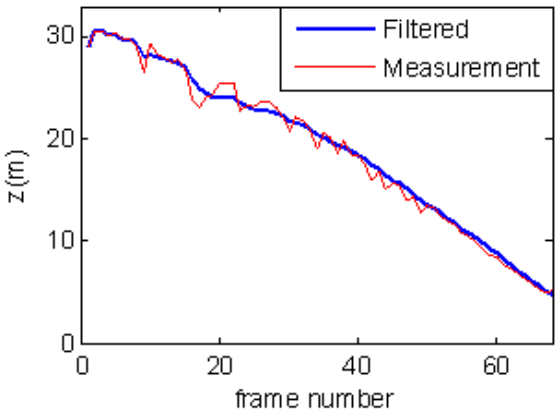
(a) Obstacle pixels superimposed on intensity image



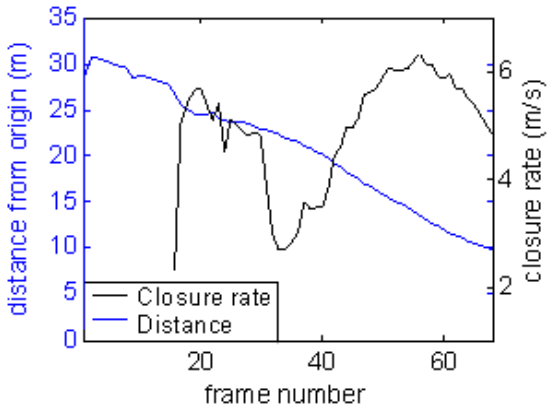
(b) Obstacle trajectory



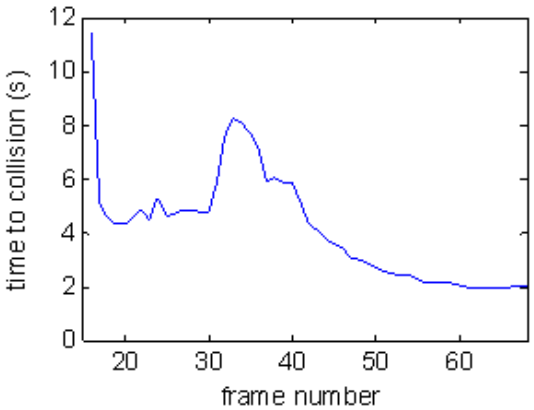
(c) x coordinate of tracked obstacle point in WRF



(d) z coordinate of tracked obstacle point in WRF



(e) Distance from origin and closure rate



(f) Time to collision

Figure 7.26: Tracking (Image Sequence 4)

7.3 Comparison of Results obtained with Synthetic Images and Real Images

When tested with real images, the system manages to detect and track generic obstacles. However, the overall performance achieved with the real images is inferior to that predicted by the simulations. Both the detection range and positional accuracy of the system are less than those achieved in the simulations. Also, there is a greater occurrence of false detections. As a result of these factors, the tracking performance is affected and obstacles only begin to be tracked once they enter the protection zone. There are a number of reasons for which the obstacle detection and tracking results obtained with the two sets of images differ from each other:

1. The main reason is that the real cameras have a shorter focal length. The simulated stereo cameras have a lens focal length of 31.2mm (554.3 pixels)(Table 3.1) whereas the real cameras have an average focal length of 3.6mm (389.4 pixels)(Table 7.5), which is almost a factor of 10 lower than the focal length of the simulated cameras. As explained in Section 5.1.2, the magnitude of the focal length affects the disparity range and the triangulation uncertainty. Therefore, although the baseline distance of the real camera setup is the same as that of the simulated camera setup, the shorter focal length of the real cameras results in a smaller disparity range and a larger triangulation uncertainty. Consequently, the detection range and positional accuracy are reduced.

2. During absolute extrinsic calibration, no positional errors are inserted in the simulated setup of the calibration targets. However, in the real setup, small positional errors (in the order of a few cm) are likely to occur. Also, because of the shorter focal length of the real cameras, the calibration targets appear smaller in the images. This makes it more difficult to accurately determine the pixel coordinates of the control points on the targets. Due to these errors in the calibration process, the uncertainty of the estimated calibration

parameters increases. In order to determine the accuracy of the calibration parameters estimated using synthetic images and real images, the control points on the calibration targets were projected from the stereo images onto the WRF using the estimated calibration parameters. Then, the distance error between the projected and actual position of each of the control points was found. Figure 7.27 shows how the distance error varies with range when using the calibration parameters estimated with the synthetic images and the real images. It can be observed that the distance error increases at a larger rate with range when using the calibration parameters estimated with the real images.

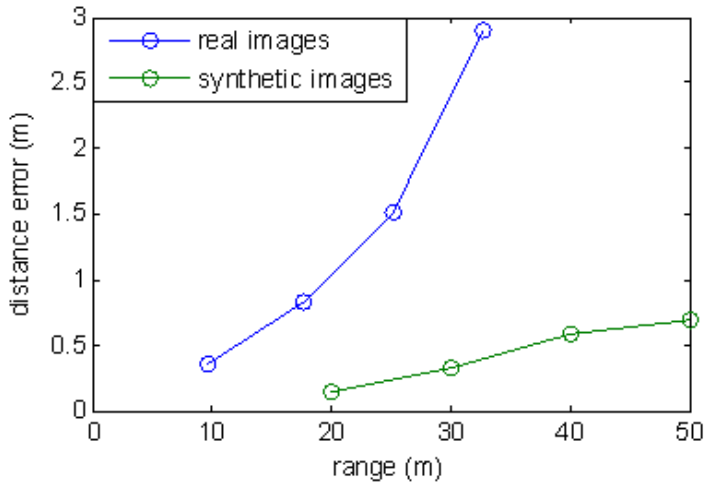


Figure 7.27: Variation of distance error with range

3. In the real images, the contrast between obstacles and the ground changes quite a lot between frames. In certain frames, the cameras expose for the ground and obstacles because they occupy a large section of the image, as in the example shown in Figure 7.28(a). In this case, the contrast between obstacles (such as the hangar) and the ground is good and this improves the results of correspondence. At the same time, the sky is overexposed and appears white. Hence, no edge features corresponding to the sky are detected and this reduces the possibility of false detections. In other frames, however, the cameras expose for the sky because it occupies a large part of the image, as in the example shown in Figure 7.28(b). In this case, the clouds are clearly visible but the ground and

the obstacles (such as the parked vehicles) are underexposed and appear dark. As a result, the obstacles are harder to detect because of the reduced contrast between the obstacles and the ground.

(a) Correct exposure of the ground and obstacles

(b) Underexposure of the ground and obstacles

Figure 7.28: Camera exposure

The problems discussed in this section highlight the challenges involved when working with real images. Although they do not undermine the results obtained during the simulations, they imply that great care is required when working with real cameras if the system is to perform as predicted by the simulations. In particular, care must be taken in order to ensure that:

1. The baseline distance and focal length of the real stereo setup are the same as those of the simulated camera setup.

2. The calibration errors are minimised, particularly during absolute extrinsic calibration. The magnitude of the acceptable calibration errors depends on the required positional accuracy. The calibration errors should be reduced until that positional accuracy is achieved. Naturally, the positional accuracy also depends on other factors, such as the performance of the correspondence algorithm.

3. The ground and obstacles are correctly exposed in each frame in order to have good contrast between them. One simple way of achieving this is by tilting the

cameras downwards such that the ground occupies a larger part of the image plane. Another method is to capture the same scene using different exposures and then combining the images using High Dynamic Range (HDR) techniques. However, this method increases the processing time.

Chapter 8

Conclusion

8.1 Strengths and Limitations of the System

When tested with either synthetic images or real images, the system is able to detect and track generic obstacles, including aircraft extremities.

During the simulations, obstacles are detected not only if they are situated inside the protection zone but also if they are outside. Good positional accuracy is achieved and, as a result, the obstacle clusters are representative of the actual obstacles. At a distance of 55m (which is slightly larger than the length of the protection zone) along the z axis of the WRF, the absolute positional error is less than 0.8m and 2m in the x and z axis respectively.

During the simulations, obstacles are also tracked outside the protection zone. Therefore, the system can potentially provide reliable TTC estimates before obstacles enter the protection zone. This implies that, in the event of a conflict, pilots can be warned in time to avoid a potential collision. Good TTC estimates are obtained when the closure rate between the tracked obstacle and the ownship is greater than about 2m/s. Generally, the larger the closure rate, the more accurate the TTC estimates are.

The tracking logic is robust enough to reject noisy obstacles by taking advantage of the temporal consistency of true obstacles. Whenever obstacles go missing, the system still tracks the true obstacle by relying on the predictions of the Kalman filter.

However, in certain tracking scenarios, the distance between the tracked obstacle and the cameras tends to be underestimated. This can potentially lead to an increase in the number of false (nuisance) alerts.

From the nine test cases used to test the system under different combinations of illumination, visibility and temporal image noise, it was observed that the best detection and tracking results are generally obtained in the test case where the illumination is good and the standard deviation σ of the image noise is 3 intensity levels. The performance of the system tends to degrade mostly under low light conditions. The performance also degrades when the image noise is increased, particularly when σ is increased from 10 to 20 intensity levels. Under low visibility conditions, the results obtained are not always predictable. The results are sometimes better and sometimes worse than those achieved in good illumination conditions.

From the simulations, it was found that the system tends to be more prone to false detections than missed detections. Since the detection rate estimates obtained are very good (particularly when σ is less than 20 intensity levels), it is possible and affordable to increase the missed detection rate of the system in return for a lower false detection rate. This can be done by decreasing the sensitivity of the system to obstacles in three stages. The first stage consists of increasing the thresholds used for edge detection in order to detect only the strongest edge features. The second stage consists of adjusting the thresholds used by the confidence tests during correspondence in order to keep only the most reliable and accurate disparities. Finally, the third stage consists of modifying the thresholds used by the clustering algorithm in order to retain only the biggest and densest obstacle clusters.

The system is quite sensitive to errors in absolute extrinsic calibration. These errors affect the positional accuracy of the system and can potentially result in more missed detections and false detections, particularly in scenarios where obstacles are situated close to the boundary of the protection zone. Therefore, great care needs to be taken to minimise calibration errors.

When tested with real images, the performance of the system is not as good as

that observed in the simulations. Both the detection range and positional accuracy of the system are less than those achieved in the simulations. Also, there is a greater occurrence of false detections. As a result of these factors, the tracking performance is also affected and obstacles only begin to be tracked once they enter the protection zone. Nevertheless, the performance predicted by the simulations can still be achieved in practice by using a longer lens focal length, reducing the calibration errors, and improving the exposure of the ground and obstacles in each frame.

8.2 Contributions

As a result of this research, three main contributions to the field of avionics can be identified:

1. **The detection and tracking of generic obstacles around large commercial aircraft in ramps and taxiways through the use of stereo vision**. This is a new stereo vision application and the operation of the system has been shown to be effective by testing it in several conflict scenarios using both synthetic and real images. The system has been designed by selecting and developing image processing and computer vision techniques that meet the specific requirements of this application, particularly the need to detect and track aircraft extremities, such as wings and wingtips. New techniques have been developed to perform correspondence and clustering. For correspondence, a modified multiresolution approach is proposed which reduces the processing time (in comparison with the time taken to process the full resolution images directly) while avoiding the problems associated with a multiresolution scheme. For clustering, a new method of grouping and filtering obstacle points, on the basis of multiple weighted criteria, is presented. This method makes use of thresholds that adjust dynamically according to obstacle distance, in such a way as to increase the detection range of the system while minimising false detections.

2. **A detailed study of aircraft safety on ramps and taxiways**. This study

consists of (a) a discussion of current procedures for aircraft separation assurance, (b) an analysis of incidents and accidents on ramps and taxiways, (c) the identification of the most common conflict scenarios and obstacle threats and (d) a discussion of the design issues of an onboard non-collaborative system for the prevention of such accidents.

3. **The development of a simulated 3D aerodrome environment to generate synthetic stereo images in order to test the obstacle detection and tracking system**. Several types of obstacles are included in this environment, such as aircraft, vehicles and buildings. Realism is enhanced through the use of high quality models and textures as well as the simulation of shadows, lens distortion, vignetting, image noise, and camera oscillations due to wingtip bending. Conflict scenarios can be simulated in different illumination, visibility, and noise conditions. With the availability of ground truth data, this environment was used to determine the accuracy, sensitivity, and robustness of the system.

8.3 Suggestions for Future Work

In order to really effective, the system needs to operate in real-time. On a 3GHz processor with 2GB of memory, each frame currently takes between 10s and 30s to process, depending on the quantity of edges in the images. Therefore, further work needs to be aimed towards the real-time implementation of the system. The main reason for the long processing time is that the whole system is currently coded in Matlab. Matlab code is interpreted, not compiled. It is optimised for carrying out matrix operations but is very inefficient during loop operations, for instance. Therefore, whenever possible, care was taken to implement the code in the form of matrix manipulations. Correspondence is the most time-intensive operation, consuming an average of 50% of the total processing time. The processing time can be reduced significantly simply by writing the code in a compiled language, such as C or C++. However, as mentioned in Chapter 4, great time savings can also be

achieved by exploiting the parallelism inherent in the stereo vision problem. For instance, multiple pixels can be processed simultaneously during correspondence and triangulation. This can be done either on a multi-core processor or on a single core processor through SIMD techniques (described in Section 4.2.1.3). In future work, there are also plans to achieve real-time operation by implementing the proposed system on an FPGA.

The system developed in this research is intended to be used on the flight deck. Therefore, an alerting function, based on the outputs of the obstacle detection and tracking system, needs to be designed. Alerts would be generated on the basis of obstacle position and TTC information. For example, if (a) the TTC is less than or equal to the time that is necessary to bring the aircraft to a complete stop (that is, an obstacle has penetrated the protection zone around the wingtips) and (b) the TTC is decreasing, then an alert is generated. A decision needs to be made on the alerting strategy, that is whether to use the visual channel or the aural channel, or a combination of both. Care must be taken to ensure that the alerting function is coherent with the current flight deck philosophy and that it does not interfere with pilot operations. Due to these (and other) human factor issues, this research is ideally conducted with the collaboration of human factors specialists and the end-users themselves.

Currently, the calibration process lacks an assessment of the goodness-of-fit or residuals. This analysis should be included in a future update of the system in order to be able to gauge the performance of calibration.

As discussed in Section 2.1.1, IR cameras and visible cameras have complementary properties and the fusion of these two sensor technologies can potentially result in a system with a better performance and reliability than the current implementation, which relies only on visible cameras. Another technology that is worth considering is Automatic Dependant Surveillance-Broadcast (ADS-B). It is envisaged that, in the near future, ADS-B will become mandatory. ADS-B data packets contain several parameters, including altitude, heading and position. Position is obtained

using Global Navigation Satellite System (GNSS) technology (such as the Global Positioning System (GPS)). One advantage of ADS-B is that it is not affected by weather conditions. Another advantage is that the positional accuracy can potentially be very high. In the case of Differential GPS, for example, decimeter (10 cm) accuracy can be achieved [104]. One limitation of ADS-B is that some aircraft (and probably any other type of obstacle) might not be equipped with ADS-B transponders. Even if an aircraft is equipped with an ADS-B transponder, this might be switched off (for instance if the aircraft is parked on the ramp). Moreover, the current ADS-B update rate is limited to 1Hz [105]. On the other hand, the stereo vision system can potentially run in real-time and can detect generic obstacles. However, its performance tends to degrade in poor weather conditions and its accuracy decreases with increasing distance between obstacles and the cameras. Therefore, by combining stereo vision and ADS-B, the overall accuracy and robustness of the system can be improved.

If the proposed solution is to be installed on an aircraft, one of the key issues that need to be taken into account is that of certifiability. In order to certify the system, it is necessary to demonstrate (amongst other things) that the system is reliable and deterministic. With the current implementation of the system, these characteristics cannot be guaranteed. For instance, the only type of sensors that are currently used are visible cameras. In order to improve the reliability of the system and make it less prone to individual sensor failures, the use of different types of sensors (as described in the previous paragraph) would be recommended. Determinism is another desirable characteristic because it makes it possible to accurately predict the performance of the system and ensures that it operates consistently, at a certain update rate. Any random elements in the algorithms used will render the system less deterministic. For example, in the current implementation of the clustering algorithm, the clusters obtained may vary depending on which 3D point is selected as the 'root' at the beginning of the process. Therefore, each of the algorithms used in the proposed system need to be analysed and potentially modified in order to ensure that it is

deterministic.

8.4 Conclusion

The main objectives of this research have been met. Incidents and accidents on ramps and taxiways have been analysed and it has been shown that current safety systems and procedures are inadequate to ensure a safe separation distance between large commercial aircraft and obstacles. The problem of collisions between aircraft and obstacles (including other aircraft) has been addressed by successfully developing and testing a stereo vision system that can be installed onboard an aircraft to detect and track generic obstacles around the wingtips during taxi manoeuvres. If implemented, the proposed system can safeguard the separation between aircraft and obstacles and can therefore effectively reduce the risk of collisions in an aerodrome environment.

References

[1] University of Malta (Malta). Internal Safety Study. Unpublished. 2006.

[2] Labayrade R., Aubert D., and Tarel J. P. Real Time Obstacle Detection in Stereovision on non Flat Road Geometry Through 'V-disparity' Representation. In *Procs. IEEE Intelligent Vehicles Symposium 2002*, Paris, France, 2002.

[3] Labayrade R. and Aubert D. Robust and Fast Stereovision Based Road Obstacles Detection for Driving Safety Assistance. In *IEICE Transactions on Information and Systems*, volume E87-D, pages 80–88, Japan, 2004.

[4] European Aeronautics: A Vision for 2020. Technical report, Advisory Council for Aeronautics Research in Europe, 2001.

[5] CAP 637 Visual Aids Handbook. Technical report, Civil Aviation Authority (UK), 2007.

[6] Airbus A380-800F Wide-Bodied Freighter, Europe. `http://www.aerospace-technology.com/projects/airbus_a380/`. (last accessed June, 2009).

[7] Rules of the Air Regulations 2007. Technical report, Ministry for Transport (UK), 2007.

[8] AAIB – Extract from Bulletin No. 7/1996: EW/C1995/11/04. Technical report, Air Accident Investigation Board (AAIB), 1996.

[9] AAIB – Extract from Bulletin No. 9/2005: EW/C2004/07/03. Technical report, Air Accident Investigation Board (AAIB), 2005.

[10] AAIB – Extract from Bulletin No. 12/2005: EW/C2004/11/01. Technical report, Air Accident Investigation Board (AAIB), 2005.

[11] ATSB Transport Safety Investigation Report, Aviation Occurrence Report 200600524. Technical report, Australian Transport Safety Bureau (Australia), 2006.

[12] Ground collision between two Airbus A320s, Denver, August 3, 2005. Technical report, National Transportation Safety Board (NTSB), 2005.

[13] Wing Tip Clearance Hazard. `http://www.skybrary.aero/index.php/Wing_Tip_Clearance_Hazard`. (last accessed January, 2009).

[14] Interim Aerodrome Requirements for the A380. Technical report, Civil Aviation Authority of New Zealand, 2004.

[15] Veram S., Lozito S., Kozon T., Ballinger D., and Resnick H. Procedures for Off-Nominal Cases: Very Closely Spaced Parallel Runway Operations. In *27th IEEE/AIAA Digital Avionics Systems Conference*, pages 2.C.4–1–2.C.4–11, St. Paul, Minnesota, USA, 2008.

[16] Taylor J. B. and Kuchar J. K. Experimental study of helmet-mounted display symbology for terrain avoidance during low-level maneuvers. In *17th AIAA/IEEE/SAE Digital Avionics Systems Conference*, volume 1, pages E41/1–E41/8, Belleview, Washington, USA, 1998.

[17] Shepherd M. J., MacDonald A., Gray W. R., and Cobb R. G. Limited Simulator Aircraft Handling Qualities Evaluation of an Adaptive Controller. In *2009 IEEE Aerospace Conference*, pages 1–12, Big Sky, Montana, USA, 2009.

[18] Airbus A380-800 Brake test. `http://www.youtube.com/watch?v=m1dv_y_3EK0`. (last accessed January, 2010).

[19] Large Flight Simulator. `http://www.cranfield.ac.uk/soe/facilities/page5268.jsp`. (last accessed January, 2010).

[20] Pang H.Y., Sundareshan M.K., and Amphay S. Superresolution of millimeter-wave images by iterative blind maximum likelihood restoration. In *SPIE Conference on Passive Millimeter-Wave Imaging Technology*, volume 3064, pages 227–238, Orlando, FL, USA, 1997.

[21] Singh M. K., Park H., Kim S. H., Tiwary U. S., and Kim Y. H. Linear and Nonlinear Methods for Passive Millimeter-wave Image Deblurring. In *IEEE International Symposium on Geoscience and Remote Sensing*, volume 5, pages 3685–3688, Seoul, South Korea, 2005.

[22] Shoucri M., Davidheiser R., Hauss B., Lee P., Mussetto M., Young S., and Yujiri L. A passive millimeter wave camera for landing in low visibility conditions. In *13th AIAA/IEEE Digital Avionics Systems Conference*, pages 93–98, Phoenix, Arizona, USA, 1994.

[23] Lin C. S, Amphay S. A., and Sundstrom B. M. Sensor fusion with passive millimeter-wave and laser radar for target detection. In *Proceedings of the SPIE Passive Millimeter-Wave Imaging Technology III*, volume 3703, pages 57–67, Orlando, Florida, USA, 1999.

[24] Corken R. A. and Evans M. A. Urban area navigation using active millimeter-wave radar. In *Proceedings of the SPIE Targets and Backgrounds VIII: Characterization and Representation*, volume 4718, pages 312–323, Orlando, Florida, USA, 2002.

[25] Tokoro S., Kuroda K., and Kawakubo A. Automotive Electronically Scanned Millimeter-wave Radar. In *SICE 2003 Annual Conference*, volume 1, pages 42–47, Fukui, Japan, 2003.

[26] Lalonde J., Vandapel N., Huber D., and Hebert M. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of Field Robotics*, 23(10):839–861, 2006.

[27] Vadlamani A., Smearcheck M., and Uijt de Haag M. Preliminary design and analysis of a lidar based obstacle detection system. In *The 24th Digital Avionics Systems Conference*, volume 1, Washington, D.C., USA, 2005.

[28] Chan P. W. and Kuo M. L. New Developments of LIDAR-Based Windshear Detection. In *24th International Laser Radar Conference*, Boulder, Colorado, USA, 2008.

[29] Bertozzi M., Broggi A., Carletti M., Fascioli A., Graf T., Grisleri P., and Meinecke M. IR Pedestrian Detection for Advanced Driver Assistance Systems. In *Proceedings of the 25th Pattern Recognition Symposium*, volume 2781, pages 582–590, Magdeburg, Germany, 2003.

[30] Andreone L., Antonello P. C., Bertozzi M., Broggi A., Fascioli A., and Ranzato D. Vehicle Detection and Localization in Infra-Red Images. In *Proceedings of the IEEE 5th International Conference on Intelligent Transportation Systems*, pages 141–146, Singapore, 2002.

[31] Dellaert F. and Thorpe C. Robust Car Tracking using Kalman filtering and Bayesian templates. In *Conference on Intelligent Transportation Systems*, 1997.

[32] Imaoka H. and Sakamoto S. A Face Recognition Algorithm Robust Against Illumination Variations Using 3-Dimensional Face Shape. In *IAPR Workshop on Machine Vision Applications*, Tokyo, Japan, 2000.

[33] Prazenica R. J., Watkins A., Kurdila A. J., Ke Q. F., and Kanade T. Vision-Based Kalman Filtering for Aircraft State Estimation and Structure from Motion. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, California, USA, 2005.

[34] Trisiripisal P., Parks M. R., Abbott A. L., Liu T., and Fleming G. A. Stereo Analysis for Vision-based Guidance and Control of Aircraft Landing. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, USA, 2006.

[35] Blanc C., Aufrere R., Malaterre L., Gallice J., and Alizon J. Obstacle Detection and Tracking by Millimeter Wave Radar. In *5th IFAC Symposium on Intelligent Autonomous Vehicles IAV 04*, Lisbon, Portugal, 2004.

[36] Ewald A. and Willhoeft V. Laser Scanners for Obstacle Detection in Automotive Applications. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 682–687, Dearborn, MI, USA, 2000.

[37] Nedevschi S., Danescu R., Frentiu D., Marita T., Oniga F., Pocol C., Schmidt R., and Graf T. High accuracy stereo vision system for far distance obstacle detection. In *IEEE Intelligent Vehicles Symposium*, pages 292–297, 2004.

[38] Williamson T. A. *A High-Performance Stereo Vision System for Obstacle Detection*. PhD thesis, Carnegie Mellon University, Robotics Institute, Pittsburg, PA, USA, 1998.

[39] Kato T., Ninomiya Y., and Masaki I. An Obstacle Detection Method by Fusion of Radar and Motion Stereo. *IEEE Transactions on Intelligent Transportation Systems*, 3(3):182–188, 2002.

[40] Sugimoto S., Tateda H., Takahashi H., and Okutomi M. Obstacle Detection Using Millimeter-wave Radar and Its Visualization on Image Sequence. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 3, pages 342–345, Cambridge, UK, 2004.

[41] Perrollaz M., Labayrade R., Royere C., Hautiere N., and Aubert D. Long Range Obstacle Detection Using Laser Scanner and Stereovision. In *Intelligent Vehicles Symposium 2006*, Tokyo, Japan, 2006.

[42] Fasano G., Accardo D., Moccia A., Carbone C., Ciniglio U., Corraro F., and Luongo S. Multisensor based Fully Autonomous Non-Cooperative Collision Avoidance System for UAVs. In *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, California, USA, 2007.

[43] Caron F., Duflos E., Pomorski D., and Vanheeghe P. GPS/IMU Data Fusion using Multisensor Kalman Filtering : Introduction of Contextual Aspects. *Information Fusion*, 7(2):221–230, 2006.

[44] Yadaiah N., Singh L., Bapi R. S., Rao V. S., Deekshatulu B. L., and Negi A. Multisensor Data Fusion Using Neural Networks. In *International Joint Conference on Neural Networks*, pages 875–881, Vancouver, BC, Canada, 2006.

[45] Stover J. A., Hall D. L., and Gibson R. E. A Fuzzy-Logic Architecture for Autonomous Multisensor Data Fusion. *IEEE Transactions on Industrial Electronics*, 43(3):403–410, 1996.

[46] Hou Z. G. Principal Component Analysis (PCA) for Data Fusion and Navigation of Mobile Robots. *Lecture notes in computer science*, 3495:610–611, 2005.

[47] Green W. E., Oh P. Y., and Barrows G. L. Flying insect inspired vision for autonomous aerial robot maneuvers in near-earth environments. In *IEEE International Conference on Robotics and Automation (ICRA 2004)*, volume 3, pages 2347–2352, New Orleans, LA, USA, 2004.

[48] Low T. and Wyeth G. Obstacle Detection using Optical Flow. In *Australasian Conference on Robotics and Automation*, Sydney, Australia, 2005.

[49] Stein G. P., Mano O., and Shashua A. A Robust Method for Computing Vehicle Ego-motion. In *IEEE Intelligent Vehicles Symposium (IV2000)*, Dearborn, MI, USA, 2000.

[50] Yamaguchi K. and Takeo Kato Yoshiki Ninomiya. Vehicle ego-motion estimation and moving object detection using a monocular camera. In *18th International Conference on Pattern Recognition*, volume 4, pages 610–613, Hong Kong, 2006.

[51] Giachetti A., Campani M., and Torre V. The Use of Optical Flow for Road Navigation. *IEEE Transactions on Robotics and Automation*, 14(1):34–48, 1998.

[52] Demonceaux C. and Kachi-Akkouche D. Robust obstacle detection with monocular vision based on motion analysis. In *IEEE Intelligent Vehicles Symposium*, pages 527–532, Parma, Italy, 2004.

[53] Roderick A. R., Kehoe J. J., and Lind R. Vision-based Navigation using Multi-Rate Feedback from Optic Flow and Scene Reconstruction. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, California, USA, 2005.

[54] Coifman B., Beymer D., McLauchlan P., and Malik J. A Real-Time Computer Vision System for Vehicle Tracking and Traffic Surveillance. *Transportation research. Part C, Emerging technologies*, 6(4):271–288, 1998.

[55] Manzanera A. and Richefeu J. C. A new motion detection algorithm based on deltasigma background estimation. *Pattern Recognition Letters*, 28(3):320–328, 2007.

[56] Vargas M., Toral S. L., Barrero F., and Milla J. M. An Enhanced Background Estimation Algorithm for Vehicle Detection in Urban Traffic Video. In *11th International IEEE Conference on Intelligent Transportation Systems*, Beijing, China, 2008.

[57] Bertozzi M., Broggi A., Fascioli A., and Nichele S. Stereo vision-based vehicle detection. In *IEEE Intelligent Vehicles Symposium*, pages 39–44, Dearbon, MI, USA, 2000.

[58] Andersen H. J., Kirk K., Dideriksen T. L., Madsen C., and Holte M. B. Obstacle detection by stereo vision, introducing the pq method. In *Second International Conference on Informatics in Control, Automation and Robotics*, pages 250–257, Barcelona, Spain, 2005.

[59] Broggi A., Caraffi C., Porta P., and Zani P. The Single Frame Stereo Vision System for Reliable Obstacle Detection used during the 2005 DARPA Grand Challenge on TerraMaxTM. In *IEEE Intelligent Transportation Systems Conference*, Toronto, Canada, 2006.

[60] Hrabar S. E. *Vision-based three-dimensional navigation for an autonomous helicopter*. PhD thesis, University of Southern California, Los Angeles, CA, USA, 2006.

[61] Sull S. and Sridhar B. Runway Obstacle Detection by Controlled Spatiotemporal Image Flow Disparity. *IEEE Transactions on Robotics and Automation*, 15(3), 1999.

[62] Mills S. Stereo-Motion Analysis of Image Sequences. In *Digital Image and Vision Computing: Techniques and Applications (DICTA'97)*, pages 515–520, 1997.

[63] Franke U. and Heinrich S. Fast Obstacle Detection for Urban Traffic Situations. *IEEE Transactions on Intelligent Transportation Systems*, 3(3), 2002.

[64] Kolb C., Mitchell D., and Hanrahan P. A Realistic Camera Model for Computer Graphics. pages 317–324, 1995.

[65] Heikkil J. and Silvn O. A Four-step Camera Calibration Procedure with Implicit Image Correction. pages 1106–1112, San Juan, Puerto Rico, 1997.

[66] Bouguet J. Camera Calibration Toolbox for Matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/`. (last accessed March, 2010).

[67] Zhang Z. Flexible Camera Calibration By Viewing a Plane From Unknown Orientations. pages 666–673, Corfu, Greece, 1999.

[68] Sturm P. F. and Maybank S. J. On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications. volume 1, pages 432–437, Fort Collins, CO, USA, 1999.

[69] Gurdjos P. and Payrissat R. Plane-based Calibration of a Camera with Varying Focal Length: the Centre Line Constraint. In *12th British Machine Vision Conference (BMVC01)*, pages 623–632, 2001.

[70] Matsunaga C. and Kanatani K. Calibration of a moving camera using a planar pattern: Optimal computation, reliability evaluation and stabilization by model selection. pages 595–609, 2000.

[71] Bouguet J. and Perona P. Camera Calibration from Points and Lines in Dual-Space Geometry. In *5th European Conference on Computer Vision*, number 1, pages 2–6, 1998.

[72] Weisstein Eric W. Singular Value Decomposition. `http://mathworld.wolfram.com/SingularValueDecomposition.html`. (last accessed February, 2010).

[73] Marita T., Oniga F., Nedevschi S., Graf T., and Schmidt R. Camera Calibration Method for Far Range Stereovision Sensors Used in Vehicles. In *IEEE Intelligent Vehicles Symposium*, pages 356–363, Tokyo, Japan, 2006.

[74] Trucco E. and Verri A. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[75] Fusiello A., Trucco E., and Verri A. A Compact Algorithm for Rectification of Stereo Pairs. 12(1):16–22, 2000.

[76] Loop C. and Zhang Z. Computing Rectifying Homographies for Stereo Vision. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, volume 1, page 131, Fort Collins, CO, USA, 1999.

[77] Belongie S. Rodrigues' Rotation Formula. `http://mathworld.wolfram.com/RodriguesRotationFormula.html`. (last accessed March, 2010).

[78] Scharstein D. and Szeliski R. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47(1):7–42, 2002.

[79] Nedevschi S., Danescu R., Frentiu D., Marita T., Oniga F., Pocol C., Graf T., and Schmidt R. High Accuracy Stereovision Approach for Obstacle Detection on Non-Planar Roads. In *IEEE Intelligent Engineering Systems (INES)*, pages 211–216, 2004.

[80] Bensrhair A., Bertozzi M., Broggi A., Fascioli A., Mousset S., and Toulminet G. Stereo Vision-based Feature Extraction for Vehicle Detection. In *IEEE Intelligent Vehicles Symposium*, pages 465–470, 2002.

[81] Kubota S., Nakano T., and Okamoto Y. A Global Optimization Algorithm for Real-Time On-Board Stereo Obstacle Detection Systems. In *IEEE Intelligent Vehicles Symposium*, pages 7–12, Istanbul, Turkey, 2007.

[82] Di Stefano L., Marchionni M., and Mattoccia S. A PC-based Real-Time Stereo Vision System. *Machine Graphics and Vision International Journal*, 13(3):197–220, 2004.

[83] Middlebury Stereo Datasets. `http://vision.middlebury.edu/stereo/data/`. (last accessed May, 2009).

[84] Fusiello A., Roberto V., and Trucco E. Efficient Stereo with Multiple Windowing. pages 858–863, 1997.

[85] Iocchi L. and Konolige K. A Multiresolution Stereo Vision System for Mobile Robots. In *AIIA (Italian AI Association) Workshop on New Trends in Robotics*, 1998.

[86] Magarey J. and Dick A. Multiresolution Stereo Image Matching Using Complex Wavelets. In *14th International Conference on Pattern Recognition (ICPR)*, volume I, pages 4–7, 1998.

[87] Broggi A., Bertozzi M., and Fascioli A. Self-Calibration of a Stereo Vision System for Automotive Applications. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3698–3703, Seoul, Korea, 2001.

[88] Huang Y., Fu S., and Thompson C. Stereovision-Based Object Segmentation for Automotive Applications. *EURASIP Journal on Applied Signal Processing*, 2005(14):2322–2329, 2005.

[89] Bensrhair A., Bertozzi M., Broggi A., Fascioli A., Mousset S., and Toulminet G. Stereo Vision-based Feature Extraction for Vehicle Detection. In *IEEE Intelligent Vehicles Symposium*, volume 2, pages 465–470, 2002.

[90] Yu Q., Araujo H., and Wang H. Stereo-Vision Based Real time Obstacle Detection for Urban Environments. In *11th International Conference on Advanced Robotics*, volume 2, Coimbra, Portugal, 2003.

[91] Collado J., Hilario C., Escalera A., and Armingol J. Self-calibration of an On-Board Stereo-vision System for Driver Assistance Systems. In *Intelligent Vehicles Symposium*, pages 156–162, Tokyo, Japan, 2006.

[92] Nedevschi S., Vancea C., Marita T., and Graf T. On-Line Calibration Method for Stereovision Systems Used in Vehicle Applications. In *Intelligent Transportation Systems Conference*, pages 957–962, Toronto, Canada, 2006.

[93] Badenas J., Sanchiz J. M., and Pla F. Motion-based segmentation and region tracking in image sequences. *Pattern Recognition*, 34(3):661–670, 2001.

[94] Buch N., Yin F., Orwell J., Makris D., and Velastin S. A. Urban Vehicle Tracking using a Combined 3D Model Detector and Classifier. In *13th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, Santiago, Chile, 2009.

[95] Koller D., Weber J., and Malik J. Robust Multiple Car Tracking with Occlusion Reasoning. In *Third European Conference on Computer Vision*, pages 189–196, Stockholm, Sweden, 1994.

[96] Call B., Beardy R., and Taylorz C. Obstacle Avoidance For Unmanned Air Vehicles Using Image Feature Tracking. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, Colorado, USA, 2006.

[97] Censi A., Fusiello A., and Roberto V. Image stabilization by features tracking. In *International Conference on Image Analysis and Processing*, pages 665–667, 1999.

[98] Song X. and Nevatia R. Detection and Tracking of Moving Vehicles in Crowded Scenes. In *IEEE Workshop on Motion and Video Computing*, 2007.

[99] Arulampalam M. S., Maskell S., Gordon N., and Clapp T. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[100] Chen Z. Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond. Technical report, McMaster University, 2003.

[101] Bertozzi M. and Broggi A. GOLD: A Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection. *IEEE Transactions on Image Processing*, 7(1):62–81, 1998.

[102] Myers K. and Tapley B. Adaptive sequential estimation with unknown noise statistics. *IEEE Transactions on Automatic Control*, 21(4):520–523, 1976.

[103] Lippiello V., Siciliano B., and Villani L. Adaptive extended kalman filtering for visual motion estimation of 3d objects. *Control Engineering Practice*, 15(1):123–134, 2007.

[104] The NASA Global Differential GPS System. `http://www.gdgps.net/`. (last accessed November, 2009).

[105] Automatic Dependent Surveillance-Broadcast (ADS-B). `http://www.faa.gov/news/fact_sheets/news_story.cfm?newsid=7131`. (last accessed November, 2009).

[106] U J. and Suter D. Using Synchronised FireWire Cameras For Multiple Viewpoint Digital Video Capture. Technical report, Department of Electrical and Computer Systems Engineering, Monash University, 2004.

Appendix A

Experiment to determine Braking Deceleration of B747

The aim of this experiment was to determine the braking deceleration of the B747 flight model (of Cranfield University's flight simulator) in the low speed regime. For this purpose, the aircraft was set in takeoff configuration (with a flap setting of 20° and a takeoff weight of 260,000kg) and was accelerated from rest to 70kts. At 70kts, the engines' thrust was reduced to idle and full brakes were applied (No reverse thrust was used and no spoilers were deployed). This experiment was repeated three times and all of the simulation parameters were recorded in each case. Figure A.1 shows the deceleration profile obtained in one of the experiment runs.

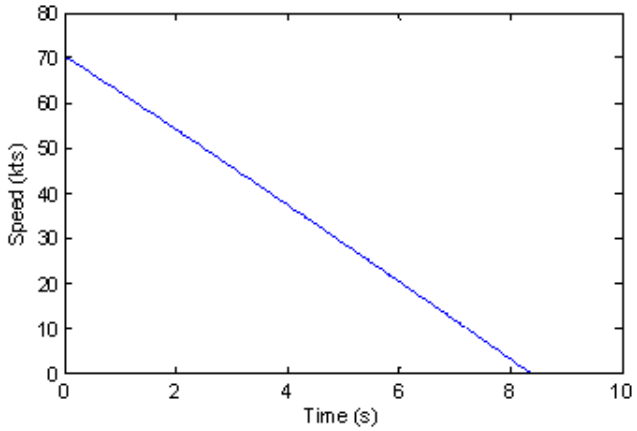


Figure A.1: Deceleration profile of the B747 flight model (low speed regime)

In the conflict scenario described when discussing the monitoring zone in Section 1.3, the ownship is taxiing at 25kts. Table A.1 shows the average braking

deceleration of the B747 for the speed range 0-25kts, for each of the experiment runs. The overall average deceleration in this speed regime is $-4.56 m/s^2$.

Table A.1: Average braking deceleration of the B747 flight model in the speed range 0-25kts

Experiment run	1	2	3
Average deceleration (m/s^2)	-4.40	-4.87	-4.40

Appendix B

Calibration

B.1 Estimation of Focal Length through the Principle of Orthogonality of Vanishing Points

This section describes the algorithm proposed in [71] in order to estimate the focal length of the cameras. In 3D space, parallel lines meet at infinity. However, in the image plane, parallel lines converge to a point known as the *vanishing point*. The planar calibration object has two sets of parallel lines, corresponding to the horizontal and vertical axes. These converge to two vanishing points as shown in Figure B.1. Let V_1 and V_2 be the points at infinity (in 3D space) corresponding to these vanishing points. The calibration object also has two other sets of parallel lines, corresponding to the diagonals of the pattern. Let V_3 and V_4 be the points at infinity corresponding to these vanishing points.

One of the properties of vanishing points is that if two sets of parallel lines are mutually orthogonal in 3D space, the coordinate vectors of the two corresponding vanishing points in the CRF are also orthogonal. Hence, the vanishing point vectors corresponding to V_1 and V_2 are orthogonal. Since the calibration pattern is a square, the diagonals are also orthogonal. Therefore, the vanishing point vectors corresponding to V_3 and V_4 are also orthogonal.

By rearranging Equation (3.1.11), the following relation is obtained between a

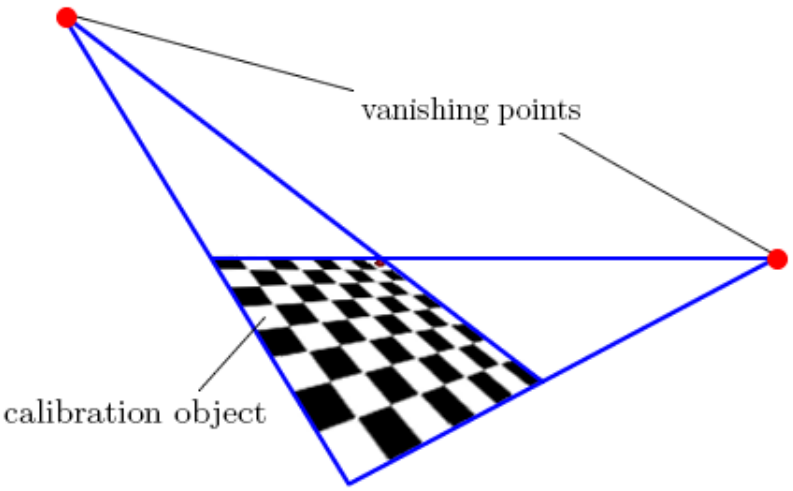


Figure B.1: Convergence of parallel lines to vanishing points in the image plane

point $P_c = (\frac{X_c}{Z_c}, \frac{Y_c}{Z_c}, 1)$ in the CRF and a point $p_i = (x_p, y_p, 1)$ in the IRF:

$$P_c = \begin{pmatrix} 1/f_x & 0 & 0 \\ 0 & 1/f_y & 0 \\ 0 & 0 & 1 \end{pmatrix} (p_i - c) = A_m^{-1}(p_i - c) = K(p_i - c) \tag{B.1.1}$$

where A_m is the modified intrinsic matrix with α and c set to 0 (Note that the principal point coordinates are subtracted from the pixel coordinates).

Referring to Figure B.2, let $v_i = (a_i, b_i, c_i)^T$ (i=1..4) be the homogeneous image coordinates (after subtraction of the principal point) of the pixels corresponding to the vanishing points V_i (i=1..4). As observed from the diagram, the image projections of the vanishing points all lie on the same line (called the *horizon line*). From Equation B.1.1, the projection of the vanishing points in the CRF is Kv_i. Using the principle of orthogonality of vanishing points, the following constraints are obtained:

$$\begin{aligned} (Kv_1)^T(Kv_2) &= v_1^T(K^TK)v_2 = 0 \\ (Kv_3)^T(Kv_4) &= v_3^T(K^TK)v_4 = 0 \end{aligned} \tag{B.1.2}$$

After substituting the values of K and v_i in Equation (B.1.2), the following equations

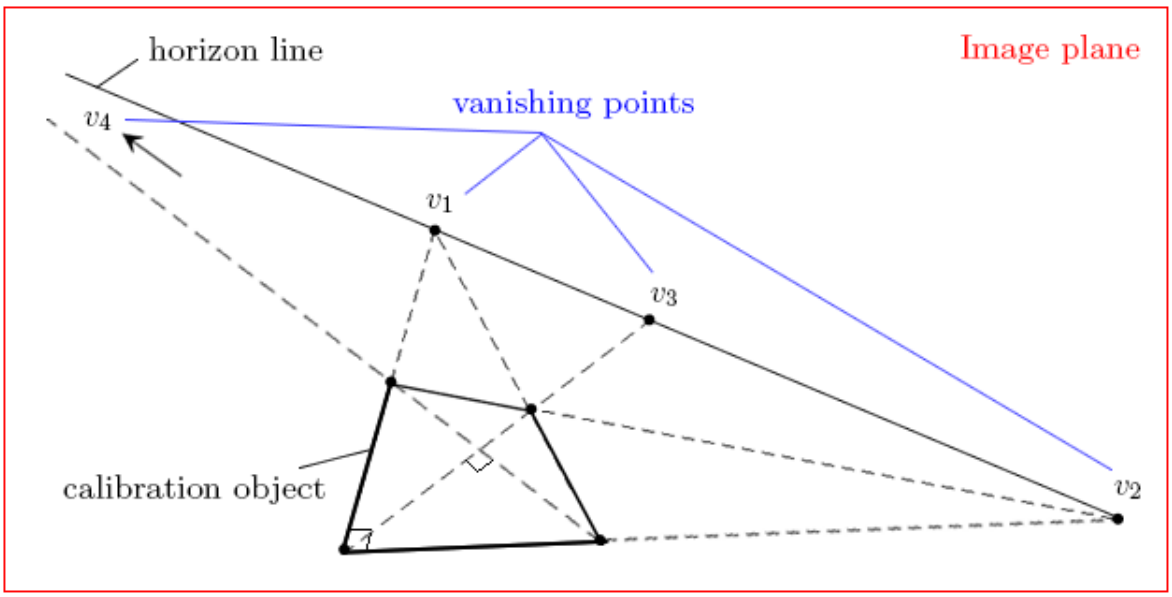


Figure B.2: Vanishing points of the calibration pattern

are obtained:

$$\begin{aligned} \frac{a_1 a_2}{{f_x}^2} + \frac{b_1 b_2}{{f_y}^2} + c_1 c_2 = 0 \\ \frac{a_3 a_4}{{f_x}^2} + \frac{b_3 b_4}{{f_y}^2} + c_3 c_4 = 0 \end{aligned} \tag{B.1.3}$$

From Equation (B.1.3) it can be observed that the focal length can be estimated from one image. However, since multiple images are used for intrinsic calibration, the focal length is estimated using a least-squares method.

B.2 Calibration Results

Table B.1 shows how the calibration error varies for different intrinsic and relative extrinsic parameters when using different numbers of calibration images.

Table B.1: Variation of error of calibration parameters with number of calibration images

	Error in calibration parameter estimation											
# of	f (pixels)		c (pixels)		k_1		T_x (mm)		T_y (mm)		T_z (mm)	
images	μ	3σ	μ	3σ	μ	3σ	μ	3σ	μ	3σ	μ	3σ
2	1.84	9.33	1.33	7.81	0.00	0.04	3.90	11.82	0.68	4.06	1.78	30.25
3	0.88	3.08	1.08	3.96	0.02	0.03	2.24	4.49	0.62	2.01	0.08	13.48
4	0.39	2.79	0.49	3.79	0.01	0.03	1.77	3.74	0.25	1.69	0.49	12.80
5	0.88	2.59	0.42	3.62	0.01	0.02	1.82	3.45	0.23	1.63	2.36	12.09
6	0.47	2.01	0.86	2.99	0.00	0.02	0.95	2.64	0.14	1.28	0.38	10.41
7	0.06	1.80	0.47	2.74	0.00	0.02	0.93	2.12	0.23	1.12	1.67	9.02
8	0.25	1.52	0.00	2.55	0.00	0.02	1.10	1.94	0.18	1.02	1.37	8.51
9	0.42	1.33	0.84	2.31	0.00	0.01	1.15	1.84	0.20	0.93	0.53	7.35
10	0.43	1.41	0.43	2.38	0.00	0.01	1.36	1.95	0.02	0.98	0.02	7.72
11	0.17	1.34	0.54	2.24	0.00	0.01	1.12	1.82	0.16	0.94	0.99	7.46
12	0.09	1.34	1.00	2.33	0.00	0.01	1.63	1.82	0.71	0.92	3.35	7.67
13	0.16	1.38	0.70	2.40	0.01	0.01	1.51	1.85	0.66	0.94	2.82	7.87
14	0.36	1.35	0.34	2.28	0.01	0.01	1.74	1.80	0.76	0.91	3.62	7.61
15	0.33	1.31	0.30	2.19	0.01	0.01	1.72	1.79	0.78	0.89	4.22	7.41
16	0.41	1.23	0.25	2.07	0.01	0.01	1.70	1.73	0.77	0.85	3.81	6.95
17	0.23	1.05	0.27	2.05	0.01	0.01	1.30	1.66	0.64	0.84	3.11	6.80
18	0.20	1.00	0.18	1.97	0.01	0.01	1.25	1.54	0.54	0.81	3.16	6.49
19	0.31	1.03	0.16	2.05	0.01	0.01	1.16	1.59	0.41	0.84	3.15	6.77

Appendix C

Rectification and Correspondence

C.1 Computation of New Values for the Focal Length and Principal Point during Rectification

C.1.1 Computation of the Focal Length

The new focal length of the stereo cameras is calculated as follows:

1. A value for the vertical focal length f_{yLeft} of the left camera is found using Equation (C.1.1):

$$f_{yLeft} = f_y \left(1 + \frac{k_1({n_x}^2 + {n_y}^2)}{4{f_y}^2}\right) \tag{C.1.1}$$

   where:

   f_y is the old vertical focal length of the camera,

   k_1 is the first lens distortion coefficient of the camera,

   n_x and n_y are the number of columns and rows in the image, respectively.

2. Similarly, a value for the vertical focal length f_{yRight} of the right camera is found using Equation (C.1.1).

3. The new focal lengths of both cameras are set to equal values. The value of the vertical focal length f_{yNew} of each camera is set to the minimum of f_{yLeft} and f_{yRight} and the horizontal focal length is made equal to f_{yNew}.

C.1.2 Computation of the Principal Point

The new coordinates of the principal point of each camera are calculated as follows:

1. The normalised image projection of each of the corners of the image plane is found by first substituting the old intrinsic parameters and the corner pixel coordinates (x_p, y_p) into Equation (3.1.10) to obtain the distorted normalised projection (x_d, y_d), and then substituting the old intrinsic parameters and (x_d, y_d) into Equation (3.1.8) to obtain the normalised image projection (x_n, y_n).

2. The normalised points are rotated by the global rotation matrix given in Equation (4.1.5).

3. The rotated normalised points are projected onto the image plane by first substituting the new intrinsic parameters[1] and the rotated points into Equation (3.1.8) to obtain (x_d, y_d), and then substituting the new intrinsic parameters and (x_d, y_d) into Equation (3.1.10) to obtain the pixel coordinates (x_p, y_p) of the projected corners.

4. The coordinates (c_x, c_y) of the principal point are found using Equation (C.1.2):

$$(c_x, c_y) = \left(\frac{n_x - 1}{2} - \bar{x}, \frac{n_y - 1}{2} - \bar{y} \right) \tag{C.1.2}$$

   where $\bar{x}$ and $\bar{y}$ are the average column and row coordinates of the projected corners, respectively.

5. The principal point coordinates of both cameras are set to the same values by making them equal to the average of the principal point locations of the left and right cameras.

C.2 Edge Detection Results

Table C.1 shows the percentage of edge pixels in 15 images captured with real cameras in different regions of an airfield. The average percentage of edge pixels over the set

[1]The principal point coordinates are set to (0,0).

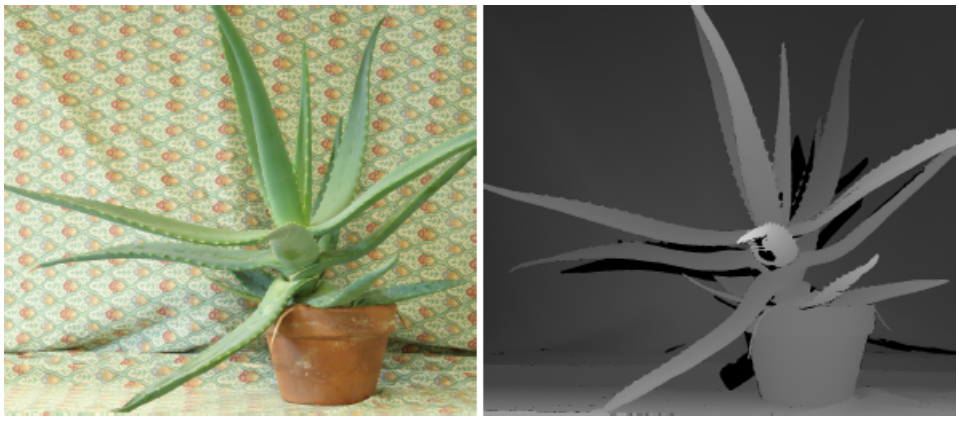
of the test images is 11.1%. Edge detection was carried out on each image using the Canny edge detector. This detection method is explained in detail in Section 4.2.2.

Table C.1: Percentage of edge pixels in a number of images captured in the ramp and taxiway regions of an airfield

Image #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Edge pixels (%)	14	12	10	10	10	14	11	11	12	11	10	10	11	11	9

C.3 Correspondence Test Images

Figures C.1-C.3 show the images that were used in the experiment described in Section 4.2.3 in order to select a suitable window size for the correspondence algorithm. These are standard test stereo images and are available online [83].

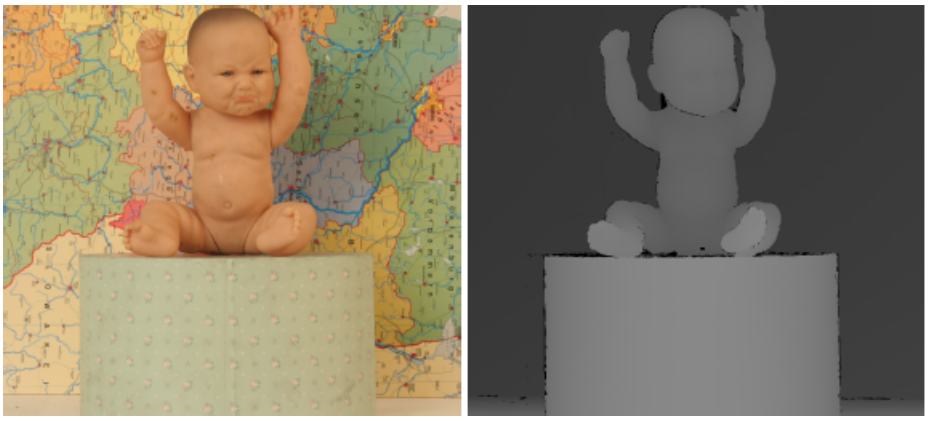


(a) Aloe

(b) Art

(c) Cones

Figure C.1: Part 1 of the correspondence test images: left color image (left) and ground truth disparity map (right)



(a) Baby1

(b) Books

(c) Dolls

Figure C.2: Part 2 of the correspondence test images: left color image (left) and ground truth disparity map (right)

(a) Midd1

(b) Teddy

Figure C.3: Part 3 of the correspondence test images: left color image (left) and ground truth disparity map (right)

Appendix D

Results of Experiment to select the Baseline Distance and Focal Length

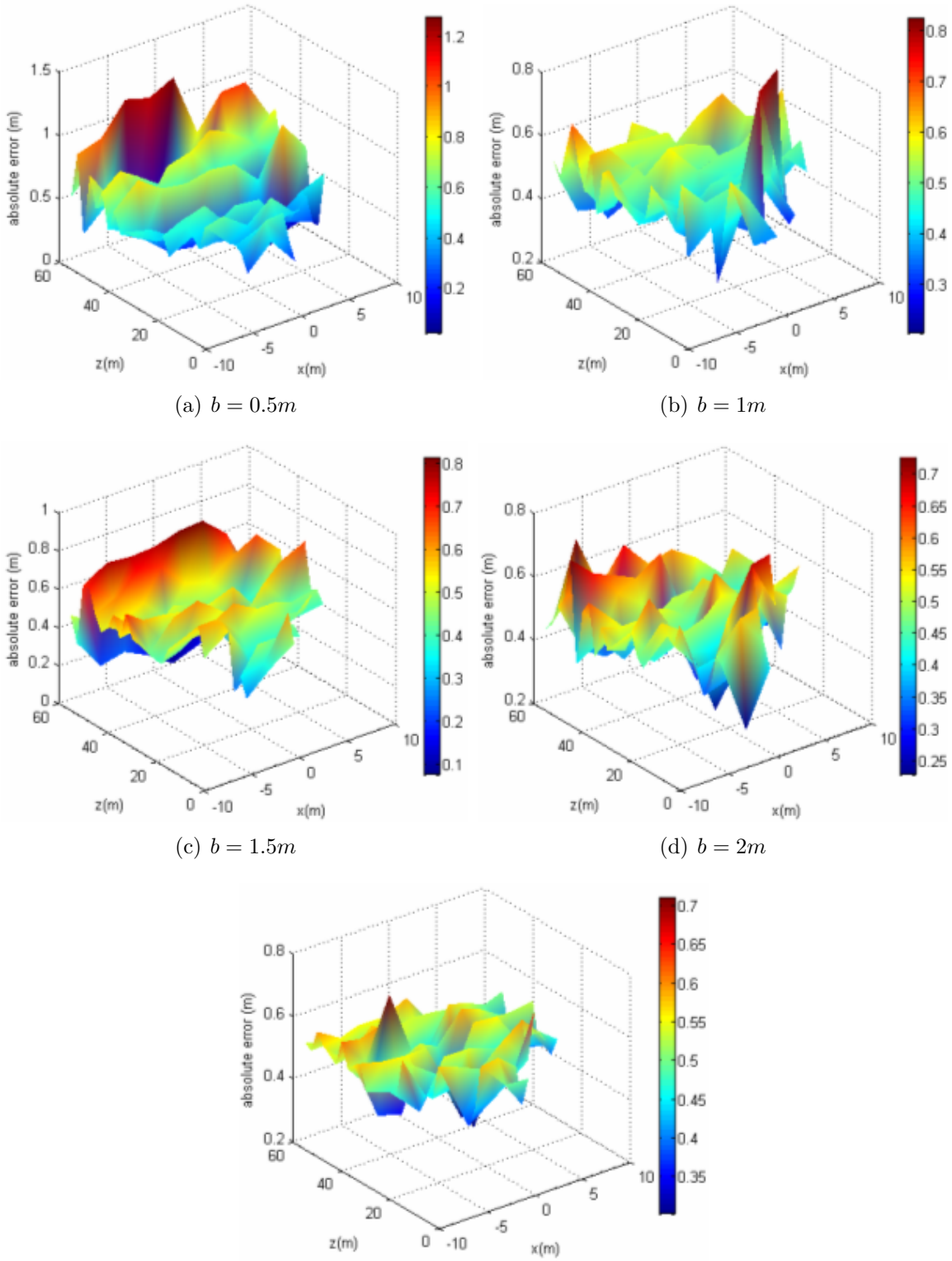
Table D.1 contains the values of baseline distance and focal length that were used in the experiment described in Section 5.1.2 in order to select an appropriate combination of these two parameters. Figures D.1-D.4 show the distance error obtained at different positions in the WRF for each combination of baseline distance and focal length. Table D.2 summarises the results by presenting only the *total* distance error obtained for each combination.

Table D.1: Values used for baseline distance and focal length tests

b (m)	f (pixels)
0.5, 1, 1.5, 2, 2.5	773 (horizontal FOV = 45°), 554 (horizontal FOV = 60°), 417 (horizontal FOV = 75°), 320 (horizontal FOV = 90°)

Table D.2: Total distance error for each combination of baseline distance and focal length

	b=0.5m	b=1m	b=1.5m	b=2m	b=2.5m
Total distance error (m) when FOV = 45°	66.50	61.37	60.91	62.68	64.63
Total distance error (m) when FOV = 60°	67.53	64.40	59.77	65.03	67.43
Total distance error (m) when FOV = 75°	85.49	70.55	60.24	67.04	66.61
Total distance error (m) when FOV = 90°	127.13	81.98	67.39	72.02	70.86



(a) $b = 0.5m$

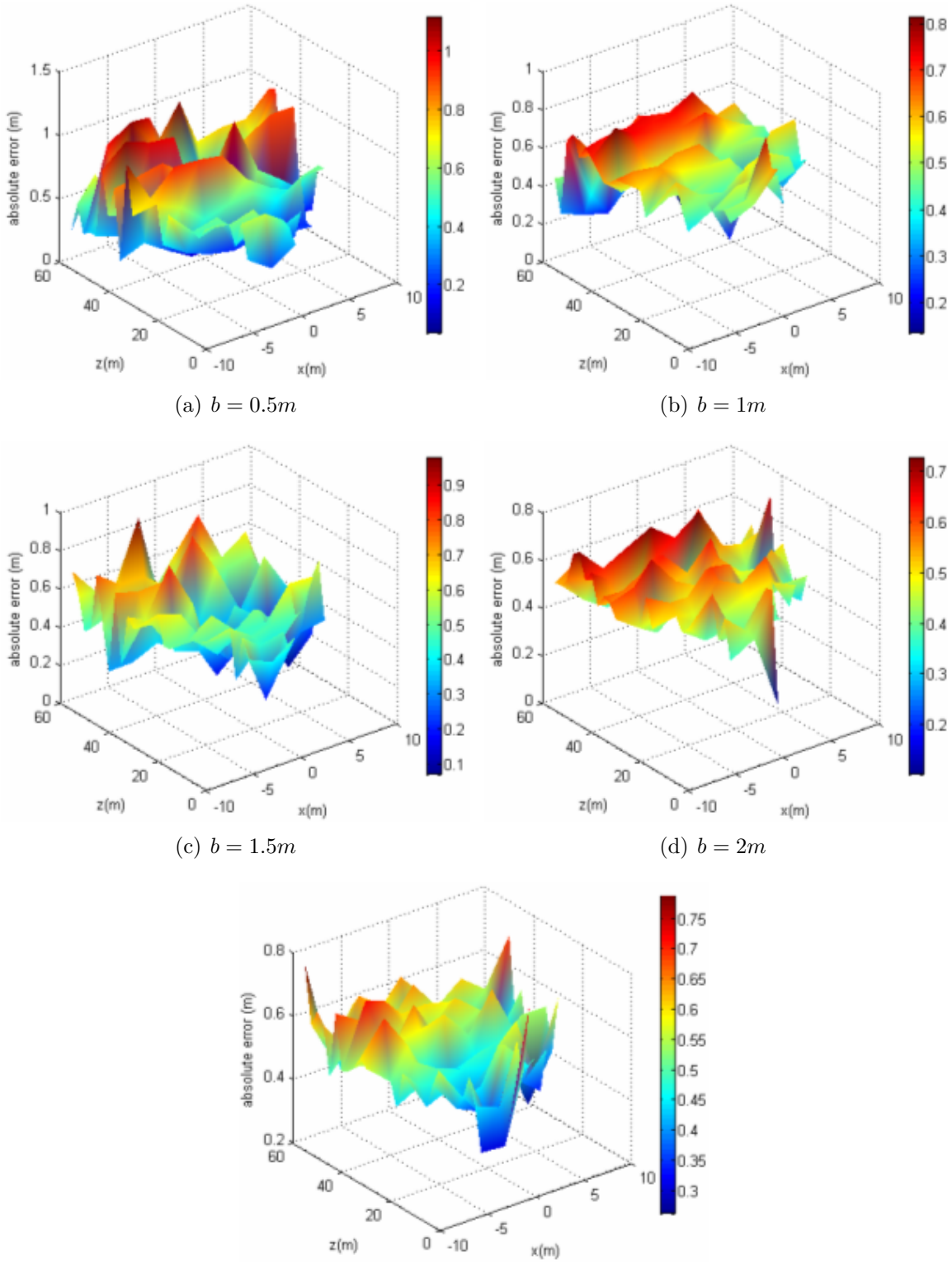
(b) $b = 1m$

(c) $b = 1.5m$

(d) $b = 2m$

(e) $b = 2.5m$

Figure D.1: Plot of distance error with respect to position in the WRF (FOV = 45°)



(a) $b = 0.5m$

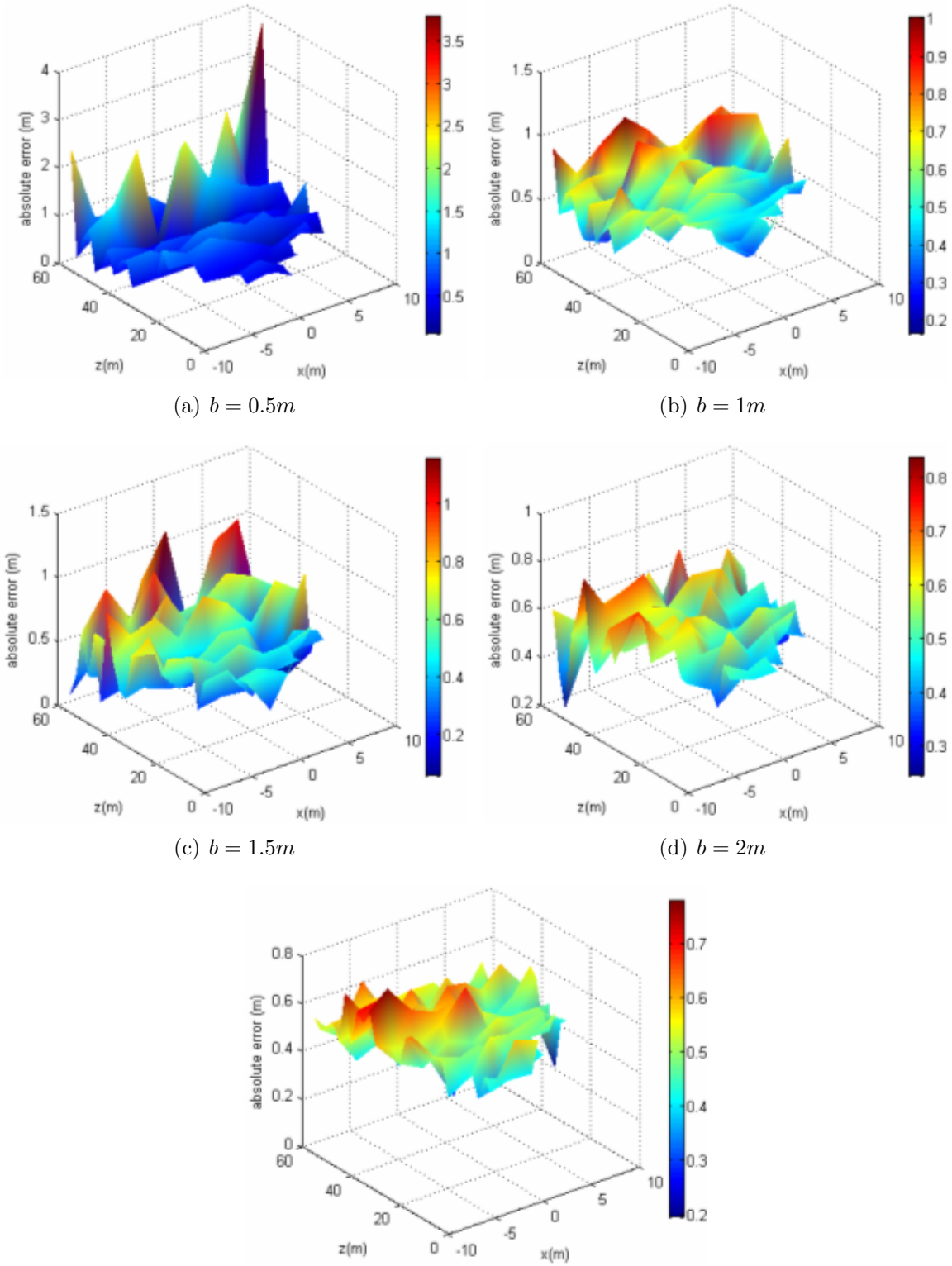
(b) $b = 1m$

(c) $b = 1.5m$

(d) $b = 2m$

(e) $b = 2.5m$

Figure D.2: Plot of distance error with respect to position in the WRF (FOV = 60°)



(a) $b = 0.5m$

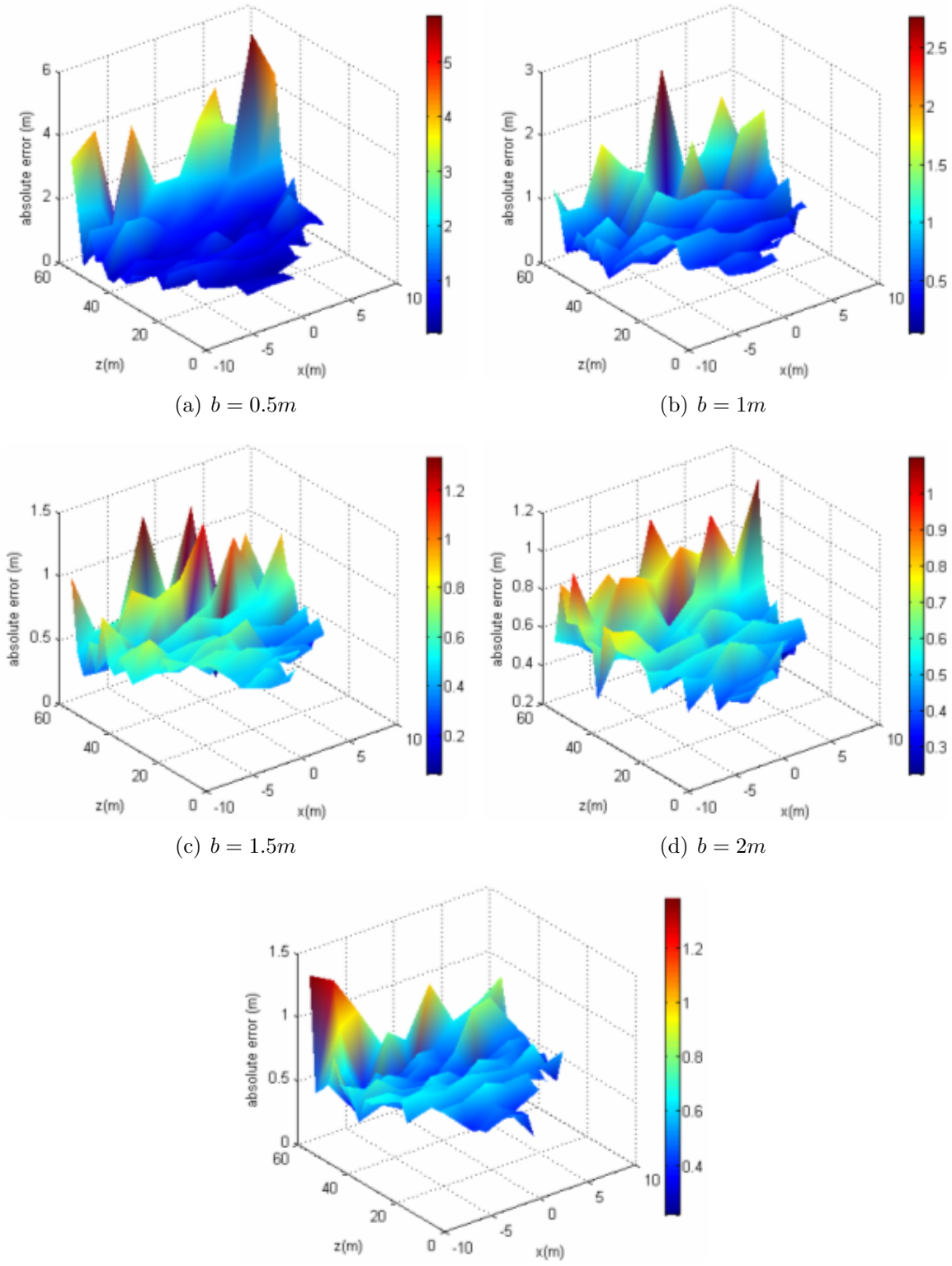
(b) $b = 1m$

(c) $b = 1.5m$

(d) $b = 2m$

(e) $b = 2.5m$

Figure D.3: Plot of distance error with respect to position in the WRF (FOV = 75°)



(a) $b = 0.5m$

(b) $b = 1m$

(c) $b = 1.5m$

(d) $b = 2m$

(e) $b = 2.5m$

Figure D.4: Plot of distance error with respect to position in the WRF (FOV = 90°)

Appendix E

Testing of the Overall System with Synthetic Images

E.1 Scenarios used to Test the Generic Obstacle Detection Capability of the System

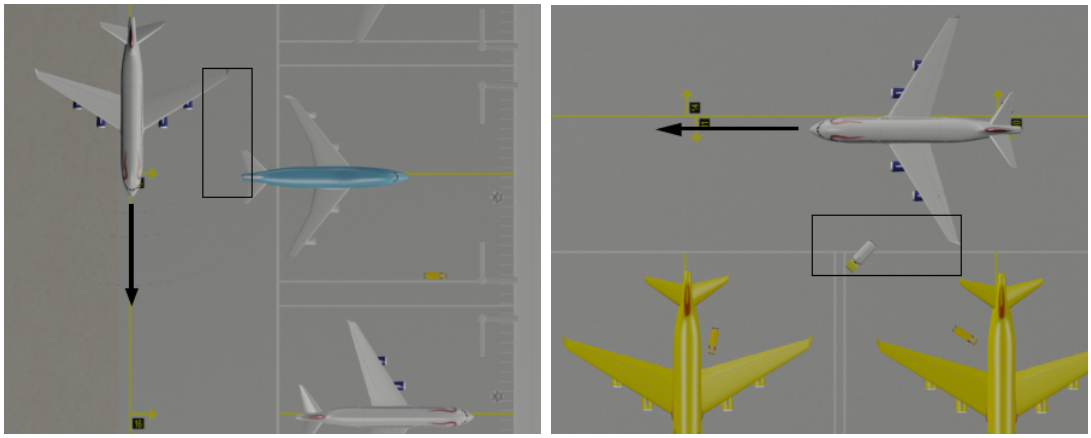
Table E.1 provides a description of the scenarios that were simulated in order to assess the ability of the system to detect generic obstacles, particularly aircraft extremities. Each of the scenarios is also presented in Figures E.1 and E.2.

E.2 Scenarios used to Test the Generic Obstacle Tracking Capability of the System

Table E.2 provides a description of the scenarios that were simulated in order to assess the ability of the system to track generic obstacles, particularly aircraft extremities. Each of the scenarios is also presented in Figure E.3. In Tracking Scenarios 1 and 3, the tracked obstacle is outside the FOV of the stereo vision system at the beginning of the scenario. In the rest of the tracking scenarios, the tracked obstacle is well within the FOV of the stereo cameras at the beginning of the scenario. Each of the tracking scenarios ends when the tracked obstacle leaves the FOV of the stereo vision system.

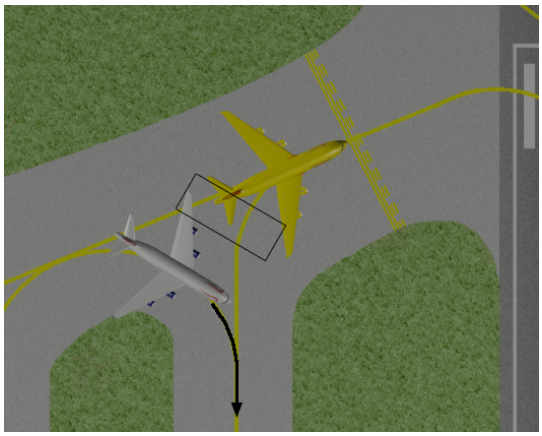
Table E.1: Description of the scenarios that were simulated in order to test the generic obstacle detection capability of the system

#	Scenario description	Obstacle types
1	The ownship taxis on the ramp at 15kts. A B747 is parked incorrectly on a stand to the left of the ownship, with its tail section partly outside the gate (Figure E.1(a)).	Aircraft, terminal building, catering truck, light poles
2	The ownship taxis on the ramp at 15kts, a few metres to the left of the centreline. Two A380s are parked at gates situated to the left of the ownship and a fuel truck is incorrectly parked at one of the gates (Figure E.1(b)).	Aircraft, fuel truck, catering trucks
3	The ownship makes a right turn behind an A380 which is holding at a Runway Taxi-Holding Position (RTHP). There is insufficient clearance behind the A380 for the ownship to manoeuvre safely (Figure E.1(c)).	Aircraft
4	The ownship moves on a taxiway at 15kts. A stationary A380 is situated on a parallel taxiway, in front and to the left of the ownship. There is insufficient clearance between the taxiway centrelines and the wingtips of the ownship and the A380 overlap (Figure E.1(d)).	Aircraft
5	The ownship makes a right turn into the ramp area. At the same time, an A380 taxis towards the ownship at 15kts (Figure E.1(e)).	Aircraft, fuel trucks, catering trucks
6	The ownship moves on a taxiway at 15kts, a few metres to the left of the centreline. A stationary B747 is situated in front and to the left of the ownship, at a taxiway/taxiway intersection (Figure E.1(f)).	Aircraft
7	The ownship taxis on the ramp at 15kts. To the left of the ownship are a number of parked aircraft (Figure E.2(a)).	Aircraft, light poles, catering trucks, stairs
8	The ownship taxis on the ramp at 15kts and then turns left into a gate. A B747 is parked in another gate to the left of the ownship (Figure E.2(b)).	Aircraft, terminal building, catering trucks, light poles, fuel truck

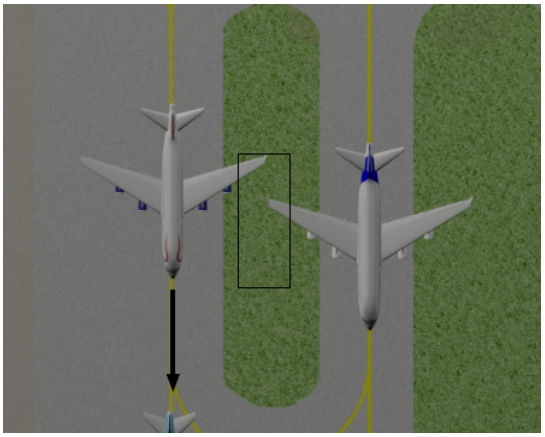


(a) Scenario 1

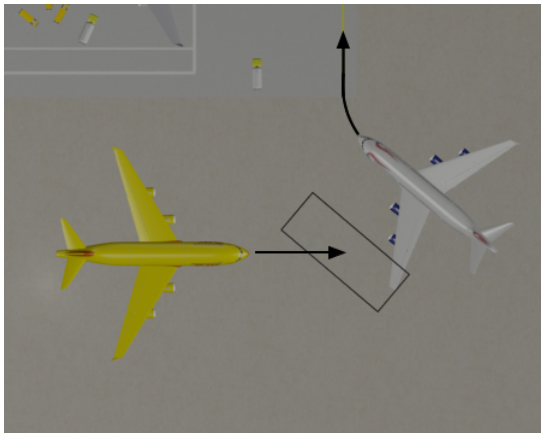
(b) Scenario 2



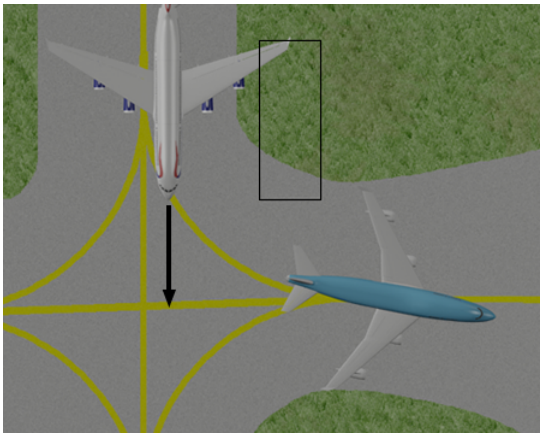
(c) Scenario 3



(d) Scenario 4

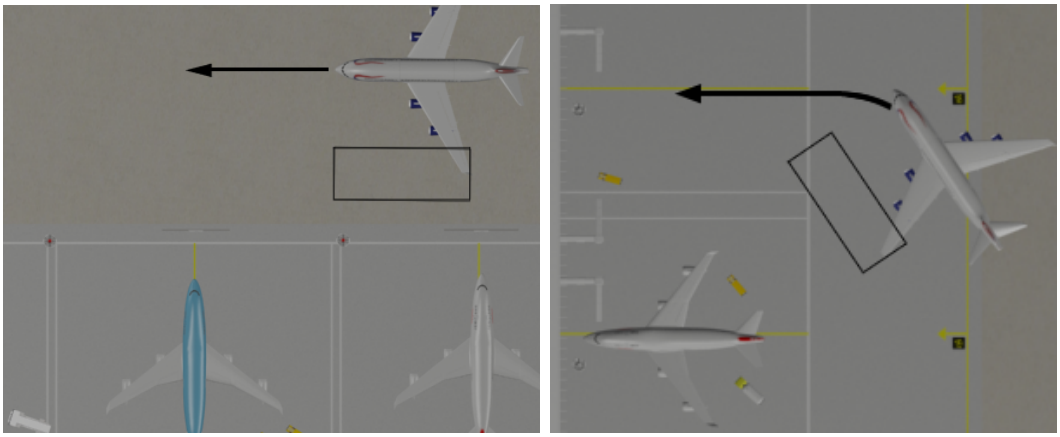


(e) Scenario 5



(f) Scenario 6

Figure E.1: Part 1 of the scenarios that were simulated in order to test the generic obstacle detection capability of the system

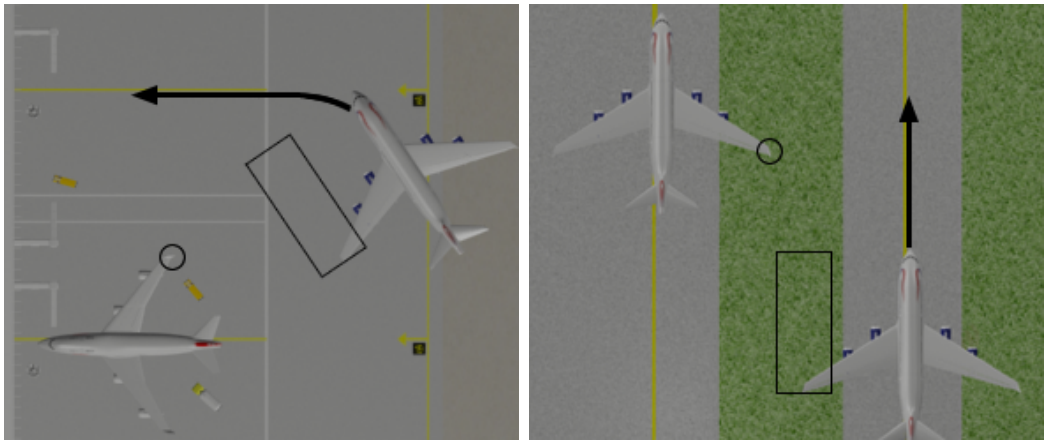


(a) Scenario 7 (b) Scenario 8

Figure E.2: Part 2 of the scenarios that were simulated in order to test the generic obstacle detection capability of the system

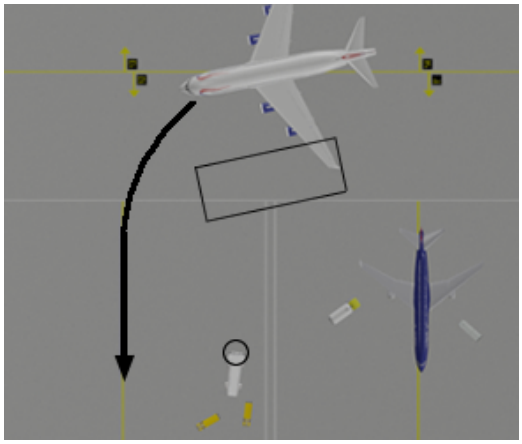
Table E.2: Description of the scenarios that were simulated in order to test the generic obstacle tracking capability of the system

#	Scenario description	Obstacle types	Tracked obstacle
1	The ownship taxis on the ramp at 15kts and then turns left into a gate. A B747 is parked in another gate to the left of the ownship (Figure E.3(a)).	Aircraft, terminal building, catering trucks, light poles, fuel truck	Right wingtip of B747
2	The ownship moves on a taxiway at a speed of 15kts. A stationary A380 is located in front and to the left of the ownship, on a parallel taxiway (Figure E.3(b)).	Aircraft	Right wingtip of A380
3	The ownship taxis on the ramp at 15kts and then turns left into a gate. A B757 is parked in another gate to the left of the ownship (Figure E.3(c)).	Aircraft, fuel truck, light poles, catering trucks, airport bus, passenger stairs	Passenger stairs
4	The ownship moves on a taxiway at a speed of 15kts. A stationary A380 is located in front and to the left of the ownship, at a taxiway/taxiway intersection (Figure E.3(d)).	Aircraft	Nose cone of A380
5	The ownship taxis on the ramp at a speed of 15kts. To the left of the ownship are a number of parked aircraft, including a B757 (Figure E.3(e)).	Aircraft, fuel truck, airport bus, catering trucks, light poles	Horizontal stabiliser of B757
6	Same as Scenario 2 but the ownship speed is 25kts.	Aircraft	Right wingtip of A380

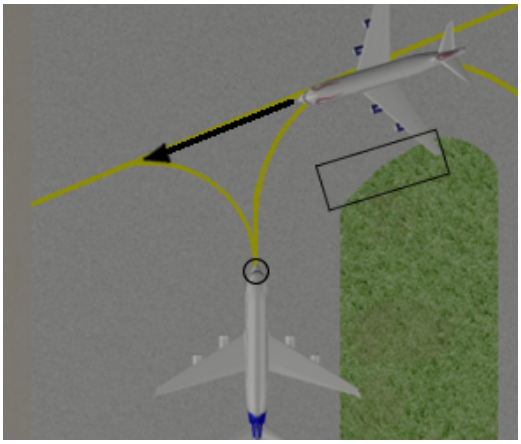


(a) Scenario 1

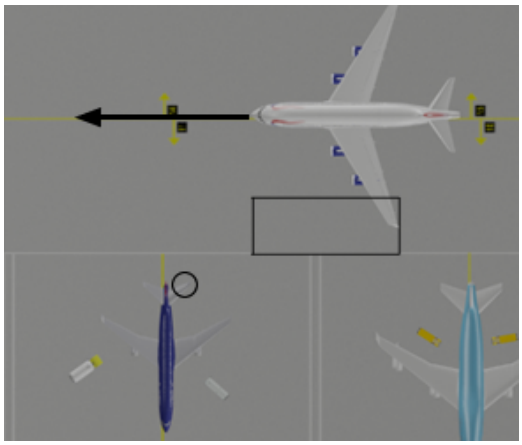
(b) Scenario 2



(c) Scenario 3



(d) Scenario 4



(e) Scenario 5

Figure E.3: The scenarios that were simulated in order to test the generic obstacle tracking capability of the system (The tracked obstacle in each scenario is enclosed by a black circle)

Appendix F

The Stereo Cameras

F.1 The FireWire Standard

The standard used for image transmission was IEEE-1394a, also known as *FireWire 400*. FireWire 400 is a serial bus standard that supports a maximum half-duplex data transfer rate of 400Mbit/s.[1] The FireWire bus is attached to the Peripheral Component Interconnect (PCI) bus of the host system via an Open Host Controller Interface (OHCI) compliant interface card. A single bus can support up to 64 node addresses.

FireWire supports two data transfer protocols: asynchronous and isochronous. Asynchronous transmission is used whenever the integrity of the transmitted data has to be guaranteed. To ensure integrity, this protocol makes use of acknowledgement packets, error checking, and data retransmission. On the other hand, isochronous transmission is used when the rate of data transfer is more important than the integrity of the data. Hence, this mode does not employ acknowledgement packets or retransmissions and is the mode used for video capture. Data is fragmented into packets, which are transmitted at regular $125\mu s$ intervals. The size of the payload is determined such that a complete set of data (such as a single frame) is guaranteed transfer across the FireWire bus within a specific time interval, thus guaranteeing a certain frame rate.

The FireWire bus can support multiple isochronous and/or asynchronous

[1]The actual data transfer rate is 393.216Mbit/s.

transmissions by sharing the bandwidth between different devices. This makes it possible to have multiple cameras streaming video concurrently. Images are transmitted from the FireWire bus onto the PCI bus and loaded into memory via Direct Memory Access (DMA). In order to support video capture, the IEEE 1394-based Digital Camera Specification (DCAM) was designed. This standard defines a set of functions and capabilities for FireWire video cameras, along with a set of register-based controls to interface cameras to host systems. The standard covers camera features such as video format, frame rate, external triggering and shutter controls, as well as manufacturer-specific advanced camera functions.

F.2 Cameras and Lenses

Each of the cameras used for the airfield experiments was a Flea monochrome camera manufactured by Point Grey Research. The main camera specifications are given in Table F.1. When two (or more) Flea cameras are connected to the same bus and run at the same frame rate, they automatically synchronise in hardware. This synchronisation process is described in detail in [106].

Custom image resolutions (below the maximum resolution) are obtained at the camera level by pixel binning. Pixel binning is the process of combining the individual charges of a square region of nxn pixels into a single larger charge or 'superpixel'. The area and light sensitivity of the superpixel are equal to the total area and total light sensitivity of the individual pixels, respectively. The entire superpixel is read as a single unit, as opposed to reading nxn pixels individually. Since each read event has a certain amount of read noise associated with it, one advantage of pixel binning is that the total amount of read noise is reduced, improving the SNR of the image.

For a resolution of 512x384 pixels, 2x2 pixel binning is used. This means that each superpixel in the lower resolution image effectively has four times the size and sensitivity of each pixel in the maximum resolution image. Although the resolution is reduced with pixel binning, the CCD sensor itself is unaffected. This means that the physical size of the image plane is not changed and the same 3D scene is captured as

with the maximum resolution.

Each of the lenses attached to the cameras was a varifocal lens manufactured by Edmund Optics. The lens specifications are listed in Table F.2. A photo of the camera and lens assembly is shown in Figure F.1.

Table F.1: Flea camera specifications

Sensor type	Sony 1/3" CCD
Maximum resolution	1024 x 768 pixels
Pixel size	$4.65\mu m$ x $4.65\mu m$
A/D converter	Analog Devices AD9849 A/D
Video output signal	8 bits per pixel / 12 bits per pixel digital data
Interfaces	6-pin IEEE-1394 for camera control and video data transmission; 4 general purpose digital input/output pins
Voltage requirements	8-32V
Power consumption	$< 3W$
Standard frame rates	1.875, 3.75, 7.5, 15, 30fps
Custom image modes	Format 7, Modes 0 (ROI), 1 and 2 (pixel binning)
Gain	Automatic/Manual modes at 0.035dB resolution (0 to 24dB)
Shutter	Automatic/Manual/Extended Shutter modes ($20\mu s$ to $66ms$ @ $15Hz$)
SNR	50dB or better at minimum gain
Trigger modes	DCAM v1.31 Trigger Modes 0, 1 and 3
Dimensions	30mm x 31mm x 29mm
Mass	46g
Lens adapter	C- or CS-mount lens
Camera specification	DCAM v1.31
Operating temperature	Commercial grade electronics rated from 0°C - 45°C

F.3 Stereo Image Sequence Capture

The flowchart shown in Figure F.2 presents the method that was implemented to acquire and process a stream of N frames from each of the stereo cameras running at the same frame rate.

The ability of the system to run at a particular frame rate depends on a number

Table F.2: Lens specifications

Focal length	$3.5 - 8.0$mm
Mount	CS
Max CCD Format	1/3”
Aperture f/# (C=closed)	F1.4 - 16C (Manual control)
Horizontal angular FOV	77.6° - 35.4°
Min working distance	0.4m
Dimensions (mm)	34.0 Dia x 43.5 L

Figure F.1: Camera and lens setup

of factors, mainly: image processing time (this must be less than the time interval between two consecutive frames), FireWire and PCI bus bandwidth, buffer size, and processing power. If the frame rate is higher than the system can handle, images will be missed or dropped. Images go missing when the buffer is full. Any new images will then overwrite the oldest images (unless they are locked). Dropped images are images that do not reach the memory buffers at all and are therefore lost in transit between the camera and main memory. This mainly occurs when the PCI bus or FireWire bus is saturated. Missing or dropped frames can be detected by comparing the sequence numbers of sequential frames.

Each frame captured by a Flea camera is tagged with a timestamp. Camera synchronisation is monitored by checking the difference between the timestamp of the frame captured by each camera and the timestamp of the frame captured by the first (reference) camera. If the time difference is greater than 1 cycle count ($125\mu s$), the cameras are considered to be out of synchronisation.[2] The timestamp can also be

[2]In other words, the cameras are synchronised to within $\pm 125\mu s$ of each other.

used to verify the frame rate.

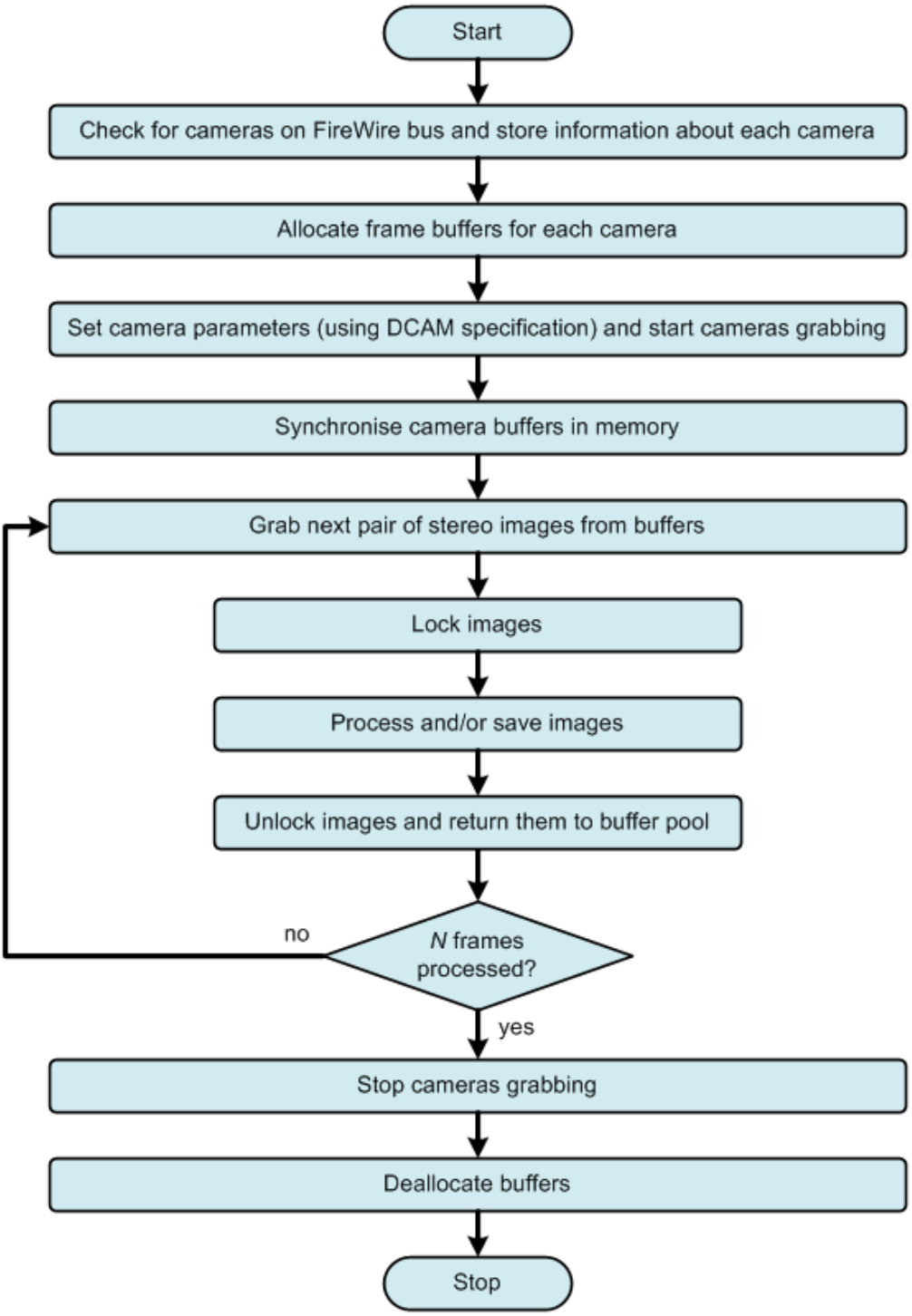


Figure F.2: Capturing a stereo image sequence

Appendix G

Testing of the Overall System with Real Images

Table G.1 provides a description of the image sequences that were captured at Cranfield Airport in order to assess the ability of the system to detect and track generic obstacles. A single frame from each of the image sequences is also presented in Figures G.1 and G.2.

(a) Sequence 1 (b) Sequence 2

(c) Sequence 3 (d) Sequence 4

(e) Sequence 5 (f) Sequence 6

Figure G.1: Part 1 of the image sequences that were captured to test the obstacle detection and tracking capabilities of the system

Table G.1: Description of the image sequences that were captured at Cranfield Airport

#	Image sequence description	Obstacle types
1	The test vehicle is driven on a taxiway towards a Lightning aircraft that is parked at the edge of the taxiway, to the left of the vehicle (Figure G.1(a)).	Aircraft
2	The test vehicle is initially stationary on one side of a taxiway. Then it is driven towards a stationary light aircraft that is situated ahead, on the other side of the taxiway (Figure G.1(b)).	Aircraft, buildings, trees
3	The test vehicle is driven on the ramp towards a stationary Jetstream aircraft and, as it approaches the aircraft, it is turned to the right (Figure G.1(c)).	Aircraft, trolley, trees
4	The test vehicle is parked at the edge of a taxiway and two vehicles pass in front of it (Figure G.1(d)).	Fire engines, van
5	The test vehicle is driven on the grass past a number of light aircraft and a fuel truck that are parked on the left (Figure G.1(e)).	Aircraft, fuel truck
6	The test vehicle is driven on the grass past a number of light aircraft that are parked on the left (Figure G.1(f)).	Aircraft
7	The test vehicle is first driven on a taxiway and is then turned left towards a number of buildings, before being stopped near a number of parked vehicles (Figure G.2(a)).	Buildings, vehicles
8	The test vehicle is driven on the ramp past two hangars and a number of parked aircraft and vehicles (Figure G.2(b)).	Hangars, aircraft, vehicles

(a) Sequence 7 (b) Sequence 8

Figure G.2: Part 2 of the image sequences that were captured to test the obstacle detection and tracking capabilities of the system