Automated Taxiing for Unmanned Aircraft Systems

By

William Eaton

A Doctoral Thesis

Submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy of Loughborough University

June 2017

I would like to dedicate this thesis to my loving parents …

Acknowledgements

This work was supported by the U.K. Engineering and Physical Sciences Research Council (EPSRC) Autonomous and Intelligent Systems programme under the grant number EP/J011525/1 with BAE Systems as the leading industrial partner. The work greatly benefits from the data set collected from an airfield provided by BAE Systems and technical advice provided by the technical officer Rod Buchanan.

In addition, I would like to acknowledge that this research was made possible by the support from the team at Loughborough University Centre for Autonomous Systems (LUCAS), led by Prof. Wen-Hua Chen. The Bayesian Network was implemented with assistance from Dr Matthew Coombes and the simulation interface between X-Plane and Matlab was created by Dr Owen McAree, both former colleagues at LUCAS.

Abstract

Over the last few years, the concept of civil Unmanned Aircraft System(s) (UAS) has been realised, with small UASs commonly used in industries such as law enforcement, agriculture and mapping. With increased development in other areas, such as logistics and advertisement, the size and range of civil UAS is likely to grow. Taken to the logical conclusion, it is likely that large scale UAS will be operating in civil airspace within the next decade.

Although the airborne operations of civil UAS have already gathered much research attention, work is also required to determine how UAS will function when on the ground. Motivated by the assumption that large UAS will share ground facilities with manned aircraft, this thesis describes the preliminary development of an Automated Taxiing System (ATS) for UAS operating at civil aerodromes.

To allow the ATS to function on the majority of UAS without the need for additional hardware, a visual sensing approach has been chosen, with the majority of work focusing on monocular image processing techniques. The purpose of the computer vision system is to provide direct sensor data which can be used to validate the vehicle's position, in addition to detecting potential collision risks. As aerospace regulations require the most robust and reliable algorithms for control, any methods which are not fully definable or explainable will not be suitable for real-world use. Therefore, non-deterministic methods and algorithms with hidden components (such as Artificial Neural Network (ANN)) have not been used. Instead, the visual sensing is achieved through a semantic segmentation, with separate segmentation and classification stages. Segmentation is performed using superpixels and reachability clustering to divide the image into single content clusters. Each cluster is then classified using multiple types of image data, probabilistically fused within a Bayesian network.

The data set for testing has been provided by BAE Systems, allowing the system to be trained and tested on real-world aerodrome data. The system has demonstrated good performance on this limited dataset, accurately detecting both collision risks and terrain features for use in navigation.

This thesis has made specific contributions to knowledge:

1. A deterministic machine vision system for semantic segmentation of outdoor scenes, using a Bayesian Network.
2. A novel method of graphical reachability, intended for use in combining superpixels into larger regions without sudden changes in colour.
3. The specification of Normalised Relative Luminance (NRL) and its relationship with distance for surface marking extraction.
4. The representative probability calculation for texton classification data, specifically for converting a Binary Decision Tree (BDT) based Support Vector Machine (SVM) classification into a probabilistic output.

Contents

List of Figures

List of Tables

List of Acronyms

AATVS Automated Aircraft Towing Vehicle System

ADS-B Automated Dependant Surveillance - Broadcast

AMM Airport Moving Map

ANN Artificial Neural Network

APU Auxiliary Power Unit

A-SMGCS Advanced Surface Movement Guidance and Control System

ASTRAEA Autonomous Systems Technology Related Airborne Evaluation and Assessment

ATC Air Traffic Control

ATS Automated Taxiing System

AVS Attribute-Value System

BDT Binary Decision Tree

BN Bayesian Network

BVT Bias–Variance Tradeoff

CCD Charge-Coupled Device

CLS Classifier Led Segmentation

CPA Closest Point of Approach

CPD Conditional Probability Distribution

CPDLC Controller-Pilot Data Link Communication

CRF Conditional Random Field

CRT Cathode Ray Tube

DAG Directed Acyclic Graph

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DGPS Differential Global Positioning System

EGTS Electric Green Taxiing System

FAA Federal Aviation Administration

FFT Fast Fourier Transform

GA General Aviation

GASR Group of Airport Safety Regulators

GBAS Ground-Based Augmentation System

GNSS Global Navigation Satellite System

GPS Global Positioning System

GTC Ground Traffic Control

HSL Hue-Saturation-Lightness

HSV Hue-Saturation-Value

IAIP Integrated Aeronautical Information Package

ICAO International Civil Aviation Organization

ILS Instrument Landing System

IMU Inertial Measurement Unit

INS Inertial Navigation System

IPM Inverse Perspective Mapping

ISTART Intelligence, Surveillance, Target Acquisition, and Reconnaissance

ITU International Telecommunication Union

JAA Joint Aviation Authorities

JPD Joint Probability Distribution

JPDO Joint Planning and Development Office

KNN k-Nearest Neighbours

LBP Local Binary Pattern

LIDAR Laser Imaging, Detection and Ranging

LVAS Low Visibility Assistance System

MFA Multiple-Frame Averaging

MFD Multiple Function Display

MLE Maximum Likelihood Estimation

MPI Motion Predicted Imagery

MR8 Maximum Response Filters

MRF Markov Random Field

MUAS Micro Unmanned Aircraft System

NAS National Airspace System

NGRM NextGen Unmanned Aircraft Systems Research and Development Roadmap

NLR National Aerospace Laboratory of the Netherlands (Nationaal Lucht- en Ruimtevaartlaboratorium)

NRL Normalised Relative Luminance

NTSB National Transportation Safety Board

OANS Onboard Airport Navigation System

OURRG	Oxford University Robotics Research Group
PDF	Probability Density Function
PGM	Probabilistic Graphical Model
PMV	Pairwise Majority Voting
PT	Proxy Technologies
RAF	Royal Air Force
RAG	Region Adjacency Graph
RBF	Radial Basis Function
RbDN	Region-based Deformable Net
RGB	Red-Green-Blue
ROI	Region Of Interest
RPV	Remotely Piloted Vehicle
RT	Radio Transmission
RTCA	Radio Technical Commission for Aeronautics
SatComms	Satellite Communications
SDC	Self Driving Car
SLIC	Simple Linear Iterative Clustering
SMR	Surface Movement Radars
SURF	Speeded-Up Robust Features
SVM	Support Vector Machine
TAP	Terminal Area Path
TCAS	Traffic Collision Avoidance System
TIS-B	Traffic Information Service – Broadcast

UAS	Unmanned Aircraft System(s)
UAV	Unmanned Aerial Vehicle
UDMS	Universal Distributed Management System
UGOMS	Unmanned Aircraft System Ground Operations Management System
UK	United Kingdom
USA	United States of America
VAS	Visual Acquisition Subsystem
VHF	Very High Frequency

Chapter 1

Introduction

1.1 Overview

Over the last few decades Unmanned Aircraft System(s) (UAS) have advanced significantly, primarily due to military development. As UAS are typically safer and cheaper than conventional aircraft, many roles which previously required a human pilot are now predominantly performed by an unmanned vehicle. Widespread military use of UAS has already created a large industry, with an estimated global worth of $11.3 billion in 2016 [122]. Future uses are also expected to include emerging civil applications, such as agricultural monitoring, aerial inspection and search and rescue. With significant growth predicted over the next ten years, a global market of $140 billion is forecast for 2026, of which 23% is expected to be civil [40].

However, despite such predictions there are still many barriers preventing widespread civil unmanned aviation. As nearly all contemporary UAS are flown by human pilots on the ground (operating as Remotely Piloted Vehicles (RPVs)), most lack the situational awareness and decision making abilities required to function autonomously. Although unmanned military aircraft have operated for decades, the vast majority of flights have occurred in controlled airspace, where each aircraft follows a flight plan and separation is performed by Air Traffic Control (ATC). These predictable interactions with other air users are highly beneficial, as the limited and delayed information available via radio link can make collision avoidance difficult for remote pilots. By comparison, aircraft operating within National Airspace Systems (NASs) commonly operate without flight plans and are responsible for self-separation. As the current capabilities of UAS are not sufficient to guarantee safety, most countries currently restrict UAS operations (both military and civil) to segregated airspace [2].

For future civil UAS to be viable, they must be capable of operating in the complex NAS environment. The biggest barrier currently preventing UAS integration is the difficulties in interacting with other airspace users. As civil airspace is “mixed user”, there will always be a small percentage of non-cooperating aircraft with which UAS will need to safely navigate around. These aircraft may be operating without flight-plans (such as recreational aircraft), without radios (such as sail-planes) or even without methods of direct control (such as hot air balloons). To account for the delay in human pilot reaction, or even complete communication loss, any UAS must be able to respond to these aircraft in the same manner as a human pilot; usually without knowledge of their intentions and with little warning of their presence. National regulators, such as the Federal Aviation Administration (FAA), are responsible for the current restrictions, and are also working with manufacturers and operators to help introduce UAS which are certified for civil use.

> *The FAA is committed to the safe and efficient integration of UAS into the NAS. However, as safety is our top priority, UAS integration must be accomplished without reducing existing capacity, decreasing safety, impacting current operators, or placing other airspace users or persons and property on the ground at increased risk.*
>
> *Michael P. Huerta - FAA Administrator [37]*

Although the majority of research pertaining to UAS operations relates to airborne activities, as the above quote states ’persons and property on the ground’ are also potentially at risk. Although much effort has already been spent on ensuring safe operation whilst airborne, comparatively little research attention has been focused on ground operations, despite similar requirements to interact with other users. In response, the specific focus of this thesis is to investigate and implement a method of controlling UAS whilst they are on the ground, i.e. automated taxiing.

1.2 Outline

This thesis details the development of an ATS for UAS operating at civil aerodromes:

Chapter 2 - Background and Context:

This chapter provides an extensive review of the background and context of the project. Due to the highly specific niche this system is intended to fulfil, the existing procedures, technologies and regulations are described in detail, explaining the reasons for the limited existing research within this area.

Chapter 3 - System Level Study:

A literature review is performed at a system level, in which the requirements for an autonomous taxiing system are identified, and the intended components of the automated taxiing system are proposed. Due to the diverse nature of the requirements, specific focus on elements is reserved for the respective chapters.

Chapter 4 - Review of Machine Vision Methods:

An investigation into methods of machine vision, specifically investigating methods of detecting known and unknown objects in cluttered outdoor environments. Various methods of visual data acquisition are reviewed and the chosen approach is established.

Chapter 5 - Review of Image Segmentation Methods:

A brief comparison of potential methods of integrating segmentation with classification is undertaken, with the decision made to implement both as separate stages. Various methods of image segmentation are also compared, in terms of their applicability to outdoor aerodrome scenes.

Chapter 6 - Implementation of Image Segmentation and Data Extraction:

Based on the review in the previous chapter, the exact implementation of the image segmentation process is defined, along with initial results. Following image segmentation, the image is divided into distinct clusters awaiting classification. As the classifier is entirely dependent on the provided data, this chapter also details the various types of information which can be extracted from an image, and which have been used for this work.

Chapter 7 - Classification:

A probabilistic Bayesian framework is established in which various types of data can be probabilistically fused. Methods of discretising and converting data into a probabilistic form are also described. Using the extracted data the classifier is trained and elements of the image can now be classified.

Chapter 8 - Semantic Segmentation Case Study:

Results are presented for the complete image classification system, applied to a real world aerodrome data set obtained at Walney Island Airport in the UK.

Chapter 9 - Depth Exaction and Collision Risk Localisation:

Methods of depth extraction from monocular images are discussed, in addition to the issues with frame-by-frame classification. Methods of temporal smoothing and localising potential collision risks are established.

Chapter 10 - Conclusions:

The final chapter concludes the thesis by providing an overview of the research, research achievements and the contribution made. It concludes by summarising the aim and objectives achieved in this research; including recommendations for future work to be carried out in this research project.

1.3 Publications

This work has resulted in one journal paper and three conference papers:

Matthew Coombes, William Eaton, and Wen-Hua Chen. Machine vision for uas ground operations. *Journal of Intelligent & Robotic Systems*, pages 1–20, 2017. ISSN 1573-0409. doi: 10.1007/s10846-017-0542-5. URL http://dx.doi.org/10.1007/s10846-017-0542-5

Will Eaton. and Wen-Hua Chen. Image segmentation for automated taxiing of unmanned aircraft. In *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*, pages 1–8, June 2015. doi: 10.1109/ICUAS.2015.7152268

M. Coombes, W. Eaton, and W. H. Chen. Unmanned ground operations using semantic image segmentation through a bayesian network. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 868–877, June 2016. doi: 10.1109/ICUAS.2016.7502572

M. Coombes, W. Eaton, and W. H. Chen. Colour based semantic image segmentation and classification for unmanned ground operations. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 858–867, June 2016. doi: 10.1109/ICUAS.2016.7502570

Chapter 2

Background and Context

The literature review for this thesis is divided into two parts; this chapter which is intended to provide context for the work, and the next chapter which provides more detail in the form of a system level study. This division is intended to establish the context of automated civil UAS taxiing, before reviewing the technical requirements.

Context for future UAS ground operations (i.e. the regulations and environment) must be established in order to define the requirements for an ATS. However, in accordance with regulatory restrictions on the weight of UAS in most countries' NAS, the vast majority of current civil UAS are too small to require conventional take-off and landing. For the few civil UAS which do require a runway, additional regulations also require separation on the ground, with manned and unmanned aircraft operating at different times to avoid incident [13]. Although smaller and less busy aerodromes could continue to facilitate UAS at less busy times, as UAS become increasingly common this will become impractical.

With this segregation in place, there are no current UAS surface operations at civil aerodromes from which to draw context. Instead, the civil aerodrome environment will be discussed by reviewing current surface operations for manned aircraft. In addition, technologies for automating taxiing for manned aircraft are also reviewed, identifying their advantages and shortcomings.

For context on UAS specific surface operations, military UAS are currently the best example, as they can be very large in size and commonly require runway landings. However, operating outside of the NAS exempts both airborne and ground movements from civil regulations, with operating procedures at military aerodromes different from that at a civil aerodrome. Therefore, although technologies and methods developed for military UAS are reviewed in order to identify potentially useful techniques, this chapter will also shed light on why certain methodologies are not suitable for this work.

2.1 Civil Surface Operations

Commercial aviation (the primary use of civil aerodromes) is commonly cited as the safest form of transportation. Despite being the fastest method of transportation, flying is statistically over 100 times safer than covering the same distance in a car [92], even though vehicles move at much lower speeds on the ground. This is primarily due to the differences in operating environment; compared to the busy surface environment for cars there is a significantly lower chance of colliding with anything whilst airborne.

Accordingly, it is not surprising that the risk of collision for aircraft greatly increases when they are on the ground. Research by Boeing states that aside from landing, ground operations are statistically the most likely ‘flight phase’ in which accidents occur [91]. This is corroborated by data from the National Transportation Safety Board (NTSB), which shows that during 2011 37% percent of the aviation accidents that occurred in the US took place during ground operations [1]. Although the overall incident rate is very low, over 10% of all fatal aviation accidents occur on the ground despite taxiing representing less than 1% of an average flight time [81]. This large proportion of accidents re-affirms that aerodromes are dangerous environments and the introduction of UAS must be done without increasing the potential risk.

Although recent regulations have specified how airport operators should aim to integrate UAS into their current operations [38], this information is typically aimed at existing small UAS operating from local aerodromes. For larger commercial airports, there has been some investigation into UAS terminal region operations using ADS-B [117], but few airports are willing to accommodate UAS (In fact, within Europe only a single civilian airport is permitted to allow UAS to land [3]). As large scale civil UAS operations are still prohibited in most NASs, this section will focus on surface operations for conventional manned aircraft.

2.1.1 Current Automation

Aerospace is typically considered an extremely high-technology industry, with automation and localisation technologies (such as autopilot and Global Positioning System (GPS)) becoming common on aircraft soon after the technologies were invented. Within the terminal area, automatic landing has been in development since the 1940’s [43], allowing aircraft to land without direct pilot input. Despite this, beyond the runway all civil aircraft currently taxi under manual pilot control [57].

The most likely reason for the current lack of research is the limited need for such a system at the current time. Manual human input is currently the most common method of control, for all forms of aircraft. As human pilots have controlled taxiing aircraft for decades, manual control has been demonstrated to be highly successful. As with any problem where a sufficient solution already exists, there is generally far less motivation to undertake research and therefore there has been little need for an automated solution. When incidents have caused concern about the safety of human-controlled taxiing, increased use of communications and rigorous operating procedures have been used instead. Although ground incidents still occur, current procedures have sufficiently mitigate most of the problems.

As a whole, the taxiing process is not the movement of a single aircraft, but instead the result of complex interactions between multiple vehicles and aerodrome users. Most large aerodromes host a wide range of aircraft, from small general aviation through large airliners. An implication of allowing such a range of aircraft is that the level of avionic equipment can differ widely. Although large civil airliners are equipped with extensive technology (such as the aforementioned autopilot and autolanding systems), they share taxiways with small General Aviation (GA) aircraft, which may operate with only a radio for communication. As light aircraft are often operated recreationally, the financial burden of upgrading prevents the uptake of more advanced systems. Therefore, the current civil user-base has been a major obstacle, preventing any form of generic automated taxiing system from being introduced for all aircraft. Instead, once an aircraft has landed the current procedure is for the pilot to operate the aircraft under manual control.

2.1.2 Dangers

Elaborating on the collision statistics in [82], the NTSB identifies pilot error as the main cause of ground operation accidents. Table 2.1 describes three sources of pilot error which occur during taxiing. Although automation would not immediately remove issues with handling and visibility, it is clear that the current manual approach can be greatly improved upon.

Handling	Aircraft are primarily designed to move well in the air and as such their ground handling is often a secondary consideration. During taxiing, most aircraft are very cumbersome and hard to manoeuvre when compared to exclusively ground based vehicles.
Visibility	Cockpit windows often provide limited visibility to the pilot. For large aircraft, the wing tips are usually not visible to the pilot if they remain seated, requiring the pilot to stand and change position in the cabin. This has lead to a large number of 'clipping' incidents where the wings collide with other vehicles or structures. In addition, in most aircraft the pilot is unable to see the ground immediately in front of the aircraft, due to the cabin arrangement. Any object low enough to be concealed becomes an undetected collision risk.
Attention	Counter-intuitively, the dangers of limited visibility are often compounded by the low speeds at which aircraft manoeuvre on the ground. With extremely long taxi-ways, pilots often become distracted whilst taxiing for prolonged periods of time and are therefore unprepared to manoeuvre around obstacles.

Table 2.1 Sources of pilot error

2.1.3 Regulations

In order to minimise the risk of accident, aerodromes are highly controlled environments with strict regulations governing all aspects of ground operations. These regulations provide rules about communications, aircraft movement and other instructions that pilots are expected to follow. For UAS to integrate safely into the civil aerodrome environment, their movements and actions must be predictable and in line with the other users. As such, the UAS should follow the same regulations as manned aircraft. As this research is being undertaken in the United Kingdom (UK), the system will be based around the rules applicable in the UK, which are listed in great detail in [105] and are outlined briefly below:

- Regardless of any given clearance, it is the duty of an aircraft commander to do all possible to avoid collision with other aircraft, vehicles or structures.
- Aircraft on the ground must give way to those taking off or landing, and to any vehicle towing an aircraft. Landing aircraft always have right of way.
- Two aircraft approaching head on must each turn right to avoid the other.
- When two aircraft are converging, the one which has the other on its right must give way, avoiding crossing ahead of the other unless passing well clear.
- An aircraft which is being overtaken by another has right of way and the overtaking aircraft must keep out of the way by turning left until past and well clear.
- Ground markings are defined by colour, with white markings signifying the runway whilst yellow signifies taxi markings.
- Information about how an aircraft should taxi is often conveyed using signals on paved runways and taxiways. Therefore the pilot must be able to see the ground whilst taxiing. On unpaved manoeuvring areas, such as grass, small flags are used to display the signals instead.
- An aircraft may only manoeuvre without the permission of Ground Traffic Control (GTC) whilst in maintenance zones. If the aircraft needs to enter the manoeuvring area of an aerodrome or taxi on the apron, it must first gain permission from the GTC.

- Although pedestrians should only be allowed in the areas used for aircraft if they have permission, it is still the pilot's responsibility to ensure that they do not collide.

As an UAS must obey the same regulations as manned aircraft, the following capability requirements must be met:

- Navigate through the aerodrome and self localise.
- Detection of other aircraft, vehicles, structures and pedestrians as well as any generic collision risk.
- Localise other aircraft relative to the UAS and recognise context through the position of other vehicles.
- Be able to detect and understand ground markings.
- Communicate with aerodrome authorities.

Of these requirements, all but the last are internal to the UAS and will be explored in greater detail in the next chapter. However, communication within the aerodrome is already a standardised procedure, and its application to UAS ground operations is reviewed below.

2.1.4 Communications

A significant aspect of civil is the delegation of authority from the pilot to the aerodrome Ground Traffic Control (GTC). During ground operations, the pilot remains responsible for ensuring the safety of the aircraft and direct aircraft control. However, the movement and overall navigation for each vehicle becomes the responsibility of the GTC at the aerodrome. In order to achieve this 'distributed authority', current regulations stipulate that manned aircraft must remain in constant verbal communication (via Radio Transmission (RT)) with the GTC. Before undertaking any activity, the pilot must first notify the GTC and ask for permission. The aircraft should then only move once given clearance, ensuring that a controlling authority has assessed that the action is viable and that the movement should not result in an incident.

In addition, it is usual for radio communications to be made during taxi, to provide additional information or changes to the how the aircraft should proceed. As such, the

GTC is responsible for over-seeing all ground based aerodrome activities, as well as instructing all aircraft on where they should be heading and what they should be expecting. To avoid confusion and to keep control of the airwaves, aircraft may not communicate with each other directly, but as all radio's are unencrypted, pilots can listen to each other's communications to stay informed. Although labour intensive (requiring the pilots to continuously listen and consider all broadcasts), this form of verbal communication has proven a reliable method of minimising risk. As manned aircraft have successfully taxied for many decades using this level of support, the type and amount of information provided by the GTC should be sufficient for UAS to operate as well. However, in order to adopt current practices for UAS, the methods by which communication can be automated must be considered.

Compared to manned aircraft, communication for UAS will always be more complicated. In order for the aircraft to be controlled at distance, control and communications systems are often highly integrated. The majority of data transmitted to UAS is interpreted by the systems on board and used to directly control the aircraft, whilst the data relayed back to those supervising the UAS is used to make the decisions about what the aircraft should do next. As such, the control of UAS can be considered distributed, with work being done both in and outside of the aircraft. Usually, the low level flight control and decision making is handled on board, whilst the more high level decisions are issued remotely.

Following legislation from the International Civil Aviation Organization (ICAO), all UAS must remain in contact with a ground supervisor at all times. Therefore, as permanent communication capability is a prerequisite of UAS operation, this thesis will assume that it is always present and functional. However, although the UAS requires a high integrity data communication to the ground, this is designed for the human operator and it is unlikely that communication with the aerodrome will use the same method. Instead, the ATS is more likely to make use of the same methods designed for manned aircraft communication when communicating with the GTC. Therefore, as almost all current aerodrome communications are achieved verbally, the biggest barrier for communication is the form of communication which is used.

With voice recognition and 'text-to-speech' software becoming more common in other fields, it is plausible that UAS could operate using conventional radio telephony. Although understanding the complexities of human speech used in daily conversation is extremely difficult, the method by which verbal communication is used within aviation is far more procedural. Due to previous aerodrome incidents (such as the Tenerife airport disaster), communications between the ATC, GTC and pilots is usually constrained to set phrases and responses. In this way, the ‘call and response’ methodology is similar to the manner in which computers natively communicate. Therefore this highly scripted method of voice communication could be fairly straight forward to implement. However, at this point in time even the most advanced voice recognition solutions still make mistakes. Combined with the reduction in audio quality created by low-fidelity Very High Frequency (VHF) radio systems, the risk of miscommunication is very large. As aerodrome safety is dependant upon all messages being understood, it is unlikely that aviation authorities will ever have enough confidence to allow a UAS to take actions based on verbal RT alone.

Fortunately, this dependence on verbal communication might be less important by the time UAS are in active civil usage. As the work undertaken in this thesis relates to an ATS in the conceptual stage, it is likely that any real-world applications will be more than a decade away. Therefore, it is important to consider what aerodromes will be like in the future. In the United States of America (USA), the Joint Planning and Development Office (JPDO) is a special initiative, aimed at predicting the future state of technology [1]. Specifically for civil aviation, the JPDO has published the NextGen Unmanned Aircraft Systems Research and Development Roadmap (NGRM) [55], which is the official publication for the intended development of aerospace within the United States. As part of the ‘roadmap’ , the NGRM makes several predictions about the coming changes to aerodrome operations by 2025, focusing on the changes coming to manned aircraft operations. In particular, future plans for manned aircraft include replacing a large proportion of verbal communication with data transmission.

Unlike today where only large airliners benefit from GTC data communications, the NGRM assumes that the majority of aircraft in 2025 will be fitted with communication systems such as Controller-Pilot Data Link Communication (CPDLC). This is due to the increased adoption of pilot aids for ground operations, commonly known known as surface guidance systems, such as the Advanced Surface Movement Guidance and Control System (A-SMGCS). Designed to assist pilots during taxi, the surface guidance systems will include Airport Moving Map (AMM) displays which are similar to GPS guidance systems used in cars. The onboard GPS sensor will indicate the the aircraft’s current location whilst data from an onboard database will be used to show the surrounding features of the aerodrome. For this to function, an accurate aerodrome map database will

[1]The JPDO is a multiagency initiative that includes the Department of Transportation, Department of Defense, Department of Commerce, Department of Homeland Security, Federal Aviation Administration, National Aeronautics and Space Administration and White House Office of Science and Technology Policy.

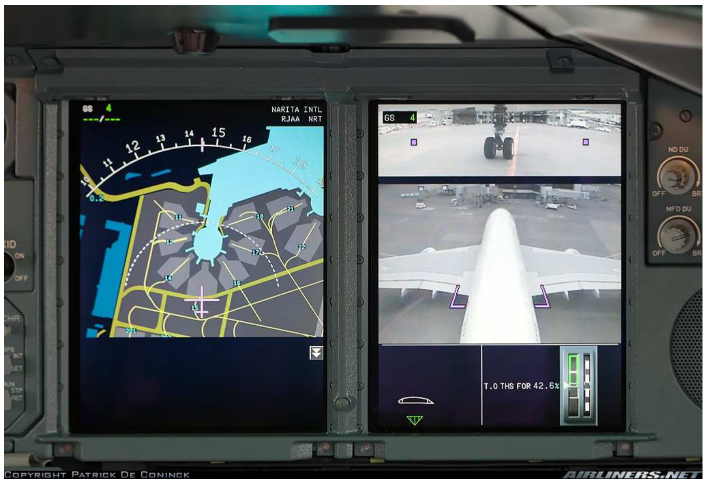


Fig. 2.1 A380 cockpit displays, showing OANS (left) and ETACS (right). ©Patrick De Coninck.

be required, showing all surface functionality, including situational awareness capabilities. The disadvantage of this system is that it will need to be kept up to date to ensure accuracy, however a digital version is easier to update than the traditional paper maps currently used by pilots.

Going beyond simply displaying the position of the aircraft, the AMM will also display a combination of data from the aircraft itself and data transmitted from the GTC. In order for information about the aerodrome to appear on the AMM, data transmissions are expected to replace many of the existing verbal transmissions used at aerodromes. Research into existing data-link technology used at aerodromes suggests that currently unused packets within broadcasts could provide enough bandwidth for the entire routing information to be relayed to incoming aircraft before they even land [69]. The introduction of data-links is also predicted to allow direct communication between taxiing aircraft, something which is current prohibited through RT during taxiing. Provided that all aircraft are suitably equipped, the GPS locations of aircraft can be shared via Automated Dependant Surveillance - Broadcast (ADS-B)/Traffic Information Service – Broadcast (TIS-B), allowing their positions to appear on the AMM display. Not only can this help increase the pilots situational awareness, but proposals also include automated systems capable of giving automatic collision risk alerts. For aerodromes that support CPDLC, facilitating UAS operation should require little modification to the proposed system for manned aircraft. In these cases, additional information provided by the aerodrome could greatly improve the situational awareness of an ATS and therefore it would be logical to for UAS to make use of this data where it is available.

However, the main downside to relying on CPDLC is that they are not always present at aerodromes. Whilst the NGRM calls for the introduction of these systems at large airports, many smaller facilities will not be quick to adopt CPDLC as they require extensive investment in training and ground infrastructure, and are only useful if aircraft operating from that aerodrome can use it. As such, an ATS for UAS cannot rely upon data-links being available. Of course, for aerodromes where CPDLC data is available, it provides such benefits that it may adversely affect the operation of the aircraft at the aerodrome if it were not to use it. Therefore integration with this kind of system must be implemented, but cannot be considered a dependency.

In addition to cases where data-links are not available, when data links are present verbal RT is still expected to be used. Despite the likely widespread adoption of aerodrome data-links, there is no expectation that all future aircraft will stop using RT. Instead, the NTSB identifies the continued need for RT for safety critical communications (such as permission to take off), where verbal confirmation from the pilot is essential. In addition, RT is still of great importance to aircraft which do not have access to data-link information. Obviously, non-co-operating aircraft are still expected to be operating in 2025 (for example light aircraft or sail-planes often function with the minimum of equipment). In these cases, traditional radio telephony will still be used between those pilots and the aerodrome.

As such, a method of integrating UAS with RT is still required. This is made simpler by the assumption that a human will remain in the loop. As there is no intention that future UAS will operate without human oversight, a human operator should always be available. As the ATS should handle the direct control of the aircraft, the responsibility of the human operator is much diminished. However, the purpose of an ATS is not to operate without a human, but instead to significantly reducing the workload of the remote unmanned aircraft pilot [42]. Therefore, the assumption has been made that the human operator will remain responsible for all verbal communications with GTC for the foreseeable future.

2.2 Autonomy and Automation

For this work, the intention is to produce a system capable of taxiing an UAS without relying on direct human input. The use of the word ‘unmanned’ within the acronym Unmanned Aircraft System(s) (UAS) is sometimes misinterpreted to mean that UAS operate without human control. In fact, the word ‘unmanned’ only indicates the lack of a human on board, with the majority of current UAS remotely controlled by human pilots on the ground. (For this reason, a common alternative to the acronym UAS is RPV).

The terms ’Automated’ and ’Autonomous’ are often used synonymously, despite having different meanings. Simply put, “automatic means that a system will do exactly as programmed” where as “autonomous means that a system has a choice to make free of outside influence” [23]. Based on these definitions, it would appear that the two concepts are direct opposites. However, as systems become more complex, the distinction between automation and automony becomes less clear.

Despite the human pilot, many elements of UAS flight control systems are highly automated. Early automation for aircraft was typically closed-loop control based directly on sensor feedback. More complex automation tasks, such as avoiding other aircraft, cannot be achieved without some form of decision making. Automated systems which incorporate decision making are termed ‘Intelligent Automation’.

Due to the potential risks of communication delay or even failure, UAS are usually capable of some form of decision making to ensure they can continue to operate without the human in the loop. Future UAS are also likely to operate with less direct input regardless of the communication quality. It has previously been identified that there is a large economic benefit for a single operator to command multiple UAS, during both ground and air operations [110]. As the authority to control the vehicle is seemingly moving from the human operator to the aircraft itself, a level of autonomy will be required, with data fusion and decision making explored in later chapters.

2.2.1 Autonomous Taxiing

In the section above, the concept of autonomy was considered with respect to the ability of the aircraft to act without input from the pilot. However, the pilot is not the only entity to have authority over the UAS. Although the term autonomous is often associated with artificial systems, the term “autonomous taxiing” has a specific meaning for both manned and unmanned aircraft alike.

Aerodromes are the most strictly controlled environment in which aircraft operate, with both the terminal area and ground operations overseen by the ATC (or GTC). Due to the close proximity of many aircraft, detailed instructions are provided and must be followed by all aircraft. Even in the unlikely event of communication loss with the pilot, the UAS is not free to make its own high level decisions but should instead endeavour to follow the instructions of the GTC.

As aerodromes become increasingly busy, the need to rely on verbal instructions is seen as a limit on the aerodrome efficiency. As such, proposals for increasing the efficiency of future aerodromes include the concept of self-control for manned aircraft, in which well equipped aircraft determine their own routing and collision avoidance, based on ‘peer-to-peer’ communication with other suitably equipped vehicles. As each vehicle would be making decisions independently, this concept is commonly referred to as *autonomous taxiing*.

Although the potential inclusion of autonomous taxiing was considered, autonomous taxiing is a separate area of research beyond the scope of this thesis. In addition, as routing information will need to be digitised for interpretation, the route provided by the GTC could simply be replaced by an onboard system, should autonomous taxiing be viable in the future. Whether the routing plan comes from an external source (i.e. the GTC) or from an additional ’autonomous taxiing system’, the ability to actually act upon these instructions and follow the route remains the focus of this work. As such, it should be made clear that the intention is to provide *automated* taxiing, and does not include any specific work for *autonomous* taxiing.

2.2.2 Communication Failure

Although general autonomous movement is not required, the ATS will require such capabilities in the event of communications failure. If communication to the UAS is interrupted, some form of autonomy is required in order for the aircraft to maintain safe operation. As verbal communications and data-links make use of different transmission methods, there are three forms of communication loss that must be considered for UAS:

- Data Communication Loss Between GTC and UAS
- Verbal Communication Loss Between Remote Pilot and GTC
- Data Communication Loss Between Remote Pilot and UAS

Of these three potential failure states, the only one in which a UAS could continue taxiing is in the case of data-link loss between the UAS and the GTC. As the availability of such data is not guaranteed, the intention is to produce a system which can function either with or without such data. As such, if data link is lost, verbal RT can be considered a backup system already in place. Provided the connection loss only affects the connection to the aerodrome, the human overseer is still connected to the UAS and can still verbally communicate with the GTC, following whatever procedure is most appropriate.

By comparison, regardless of whether an aircraft is manned or unmanned, loss of verbal communication with the GTC can have serious consequences. In the unusual event that verbal RT transmission is lost but data links remain, it may be feasible that instructions can be relayed via this method. However, as other aerodrome users may not have access to the same data, there is no method of verifying the intentions of other aerodrome uses. Therefore, provided the human operator still has control over the UAS, the remote pilot can take whatever action is appropriate to make the aircraft safe, before waiting for communication to be restored.

The third possible mode of communication failure is data loss between the human operator and the UAS. Unlike communication loss with the GTC, this form of failure removes the ability to depend on a human operator. If GTC communications remain, the aircraft could simply continue to follow instructions. Although it is beyond the scope of this work, a simple verbal declaration by the UAS over RT would be sufficient to inform the aerodrome and other users that an error has occurred, potentially requiring new instructions. Without the human operator available, it is most likely that the UAS will simply be instructed to stop where safe by the GTC via data link.

Finally, total communication loss must also be considered. If the communication is lost at an aerodrome without data links, or if communication loss extends to all systems, the UAS must act upon its current situational awareness to mitigate dangers to itself and others. Typically, when communication loss occurs in the air, UAS have relatively little chance of actually causing a collision. Therefore, the standard procedure is for the aircraft to loiter in a safe area autonomously until communications are regained, or to attempt to return to home base if it is within range. By contrast on the ground, the aircraft has no need to keep moving and can sit perfectly still. This would seem to be the safest option, but the UAS may have lost communication in an area which is dangerous. For example, should communication loss occur whilst on the runway, the UAS should be able to leave the runway to allow other aerodrome users to land. Instead, the safest option in the event of communications failure would be to have the UAS navigate (or stay) in the nearest area in which an incursion with the runway cannot occur, such as on a grass verge. For UAS with an aerodrome database, the nearest safe area should be known and the system GPS can be used to navigate to it. Alternatively, should communication failure have rendered all external systems unavailable including GPS, the UAS would need to rely on its own sensing capabilities to recognise a suitable safe area. The aircraft should then remain stationary until assistance arrives. As such, although autonomous taxiing is not explored, the need to autonomous functionality is recognised and the ATS will be designed with such capability in mind.

2.3 Current Unmanned Ground Operations

This section will cover the current state of UAS taxiing, outlining operating procedures and technologies in active use. As stated above, UAS are currently forbidden from operating out of unsegregated aerodromes and there are no well documented examples of existing civil UAS taxiing which can be investigated. Instead, all current examples of large UAS taxiing procedures will have to be taken from military operators. (In this work, the term ‘large’ is used to define any unmanned aircraft which is sufficient in size to function much like a conventional aircraft, i.e. requiring access to a runway for take-off and landing). The two main forms of control used by current military UAS are remote control and external recovery.

2.3.1 Remote Control

The most common control method for large military UAS is to be flown under remote manual control. For military unmanned aircraft (both combat and support roles), the rules of engagement require that a human operator is in control at all times. As such, all British military unmanned aircraft are Remotely Piloted Vehicles (RPVs) [19] during operations. With the functionality already in place, the simplest solution for ground operations is continue to rely on the same mission pilot, who guides the aircraft using the same conventional controls as used whilst in flight.

Although this method is straightforward to implement, there are many drawbacks. As the pilot is controlling the aircraft based on feedback from a screen, situational awareness is limited to the view from the on board cameras. As such, visibility is likely to be worse than a pilot would encounter on a conventional aircraft, with the 2D image also affecting the pilots ability to estimate both scale and range.

In addition, the method of remote control communication can also adversely effect the pilots ability to control the aircraft. Many military UAS are controlled via Satellite Communications (SatComms), which reliably allow the remote pilot to communicate with the UAS from anywhere in the world. However, this will inevitably introduce some delay between when instructions are sent and when feedback is received. Although airborne manoeuvres are usually quite slow (for non-combat aircraft), responsive feedback is required during taxiing. Any delay can make manual control of fine activities (such as navigating a corner) extremely difficult.

To overcome this problem, the pilot must be closer to the aircraft in order to minimise the delay. Therefore, military UAS ground operations commonly make use of a separate operator based at the aerodrome. As these *'taxi specific'* operators are based locally, the aforementioned delay problem is greatly alleviated. However, a multi-user communication set-up requiring hand-over of authority increases operational complexity, whilst also increasing the likelihood of communication failure.

A practical consideration regarding the use of remote control is the available bandwidth. Whilst controlling an UAS is fairly simple, the amount of data captured by the on-board sensors can be immense, with most military UAS making use of extensive ground based transmitters or satellite communications to relay information. As this is highly expensive, it is more likely that civil UAS will use a less powerful localised transmission method (Whilst ADS-B is fine in the air, it is too powerful to use on the ground). For multiple aircraft operating in the same area, the available bandwidth would rapidly be used up.

By far the largest concern with using any remote system is the potential for communication failure. For example, a recently landed UAS which becomes stranded on the runway due to communications problems would represent a large safety risk. The obvious way of overcoming this issue is to give the aircraft a level of autonomy, to allow them to navigate to safety during communication failure. However, if a requirement of any UAS system would be the ability for the aircraft to direct itself in times of difficulty, it would make more sense to abandon the remote control aspect and instead have the UAS systems navigate and control itself.

Compared to military aerodromes, civil aerodromes have a much wider variety of users and depend greatly on consistency for efficient operation. The reduction in efficiency than comes with remote control is a major reason why current UAS operating procedures are unsuitable. With projects such as ASTRAEA working to bring UAS into civilian airspace, the need for automated operations has increased. In addition to interacting with civil air traffic, a key goal of these programmes is to reduce the operators workload, with the intention of multiple UAS being controlled at once. Therefore, requiring a human pilot to taxi UAS individually is a far less attractive prospect. As any unmanned aircraft operating within such an environment should be able to follow the same regulations as manned aircraft, but cannot rely upon a human pilot, automated taxiing is required.

2.3.2 External Recovery

Although current military UAS commonly use remote control when on the ground, alternative ground operating procedures are also in use. Often, the techniques used are a reflection of the capabilities and size of the UAS being used. Despite similar requirements, many taxiing solutions for military UAS are not compatible with civil use. As both the aerodrome and aircraft share the same operator, military aerodromes often allow special dispensations for unmanned aircraft. This has resulted in some UAS being operated in unconventional ways.

For example, one of the key benefits of UAS is their size compared to conventional manned aircraft. As unmanned aircraft do not carry a human pilot or the associated equipment required to sustain human life, UAS are generally far smaller than their manned counterparts. This allows UAS to be more efficient and cheaper to operate (as well as more difficult to detect/shoot down) when compared to manned aircraft capable of similar roles. In order to maximise this benefit, military UAS are commonly equipped with only essential equipment, removing any apparatus surplus to mission requirements to further reduce weight. Without the need to consider pilot safety, features which would be considered essential to manned aircraft can be removed. For example, to save weight during flight neither the *Thales Watchkeeper WK450* nor the *USMC RQ-7B 'Shadow'* are equipped with wheel brakes.

As neither vehicle is able to decelerate after touchdown, the landing procedure is far from conventional, requiring an arrestor gear system deployed to catch a pendant extended across the runway during landing. Not only does this approach take far more time than conventional landing, but it would also represent an enormous safety risk in a civil environment if done in the presence of other users. As such methods of launch and retrieval necessitate allocating an entire runway to UAS activity, the procedures are completely incompatible with civil aerodrome operations.

Beyond landing, self-propelled taxiing is also not possible for either aircraft due to the safety implications of not being able to stop. Instead, movement on the ground requires human intervention, with both UAS being towed into position by a support vehicle before launch. As this practise is not dissimilar to the use of tugs for airliners it could be argued that such a technique could be deployed at civil aerodromes. As this methodology is already being promoted as a potential method of introducing *autonomous* taxiing for manned aircraft [77], discussion of potential external recovery methods for civil aerodromes will be continued below, in section 2.4.1.

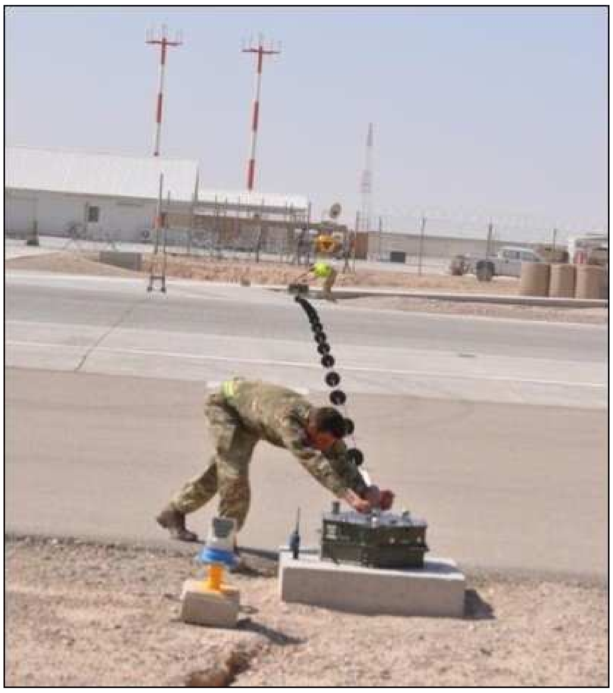


Fig. 2.2 Watchkeeper WK450 UAS Arrestor hook being deployed across runway. [104]

2.3.3 Radionavigation and Automated Way point Following

The most functional examples of current ATS operate using radionavigation systems, such as GPS. Despite the complexities involved with other aspects of producing an ATS, the act of moving the aircraft is fairly straightforward. As aerodromes are designed to make aircraft movement easy (i.e. flat, paved terrain with minimal obstacles), navigation can be achieved by using high accuracy Differential Global Positioning System (DGPS) to manoeuvre the aircraft to way points on an aerodrome map. This form of position-based auto-taxiing has been used on the GlobalHawk UAS since 2002 [68] and has proven that automated UAS taxiing is possible on external position data alone. As the methods used by the military UAS such as the GlobalHawk are unpublished (and potentially classified), an in depth review of the techniques used is not possible. However, similar systems produced by competitor companies have published methods allowing techniques to be investigated. The Universal Distributed Management System (UDMS) created by Proxy Technologies (PT) [78] is such a system.

According to PT, the majority of UAS incidents relating to medium or large UAS have occurred due to operator error during taxiing. As such, the UDMS was created to provide an automated solution that eliminates this risk. As simple waypoint following is not a significant achievement, the primary purpose of the UDMS is to provide an interface through which a single operator can control multiple UAS simultaneously. Unlike traditional ground controls that simulate a cockpit environment, the UDMS provides an overview of all UAS activities using a map display similar to AMM. After receiving any relevant routing information from the GTC, the human operator simply selects the destination and any intermediate points using a simplistic mouse driven interface. The precise route is calculated by the system and defined as a series of GPS waypoints for the UAS to follow.

After calculating a route, the system submits the route to the human overseer for approval and asks for permission to begin taxiing. If the route is deemed acceptable, the system navigates to the runway hold line. In the event of an emergency, the human operator can take control, adjusting speed, heading or even stopping the vehicle and taking manual control. As such, the UDMS provides an example machine-human interface for automated UAS. As the UDMS has been tested under real-world conditions, it demonstrates that GPS data alone can be used to taxi an aircraft. The procedural method of alerting the pilot at certain stages of the taxi process also demonstrates how a shared-autonomy system could function. By having major decisions validated, issues can be identified early on and pre-emptively resolved, rather than waiting for the overseer to react when the UAS get into a unwanted situation.

Provided that map data is known and accurate, external position data is sufficient to taxi under supervision. However, relying on a single source of data makes such systems highly vulnerable to any GPS error (caused either though inaccuracy or malicious alteration). To ensure that a vehicle is following the correct course, additional sources of position data could be used to validate the GPS data. In addition, although GPS only methods can navigate around known static collision risks, these approaches still require human interaction for avoiding other types of object.

For the rest of the UDMS, little mention is made of how the system operates beyond its use of GPS for positioning. As the aircraft has no method of directly sensing the environment the system is unable to avoid obstacles and the human operator is required for collision avoidance. As such, this form of DGPS controlled auto-taxiing can only used at segregated aerodromes, in order to remove the risk of collision with other aircraft.

Collision avoidance of known vehicles could also be achieved using radionavigation data. Systems such as CPDLC now exist to facilitate the sharing of data between users of aerodromes and the GTC. From work such as Low Visibility Assistance System (LVAS), these systems have been shown to be capable of providing the basic information required for autonomous operation. However, although vehicles known to the aerodrome may be avoided if they are broadcasting their position, most other forms of mobile obstacle are not so well equipped.

In order to produce a robust ATS it is insufficient to rely on a single source of position data, especially one external to the aircraft. Instead, the aircraft needs to actively seek it's environment, not only to detect potential risks, but also to validate it's position. As external data cannot be relied upon to provide information on all collision risks, direct sensing capabilities are required.

2.4 Proposed Automated Taxiing Systems

The current state of automation within aircraft ground operations is highly limited, with both manned and unmanned aircraft lacking a suitable method of taxiing without human control. Although the techniques used in military ground operations are functional, they are unsuitable for civil use due to incompatibilities with the civil aerodrome environment. In addition, although the current use of manual control typical of civil aerodromes could continue albeit through remote control, the dependence on data links for safety critical movement is not suitable for environments with none-cooperating users. From the current systems in use, it is clear that development is required to allow civil UAS to taxi.

Although this work is motivated by the lack of any current system in operation, it is not the only work intending to improve aerodrome surface operations. For civil ground operations, a large focus area for research is improving efficiency. As the levels of traffic at aerodromes have increased, aircraft are spending more time taxiing to and from the runway. Often this is countered by physically expanding the airport faculties, building new runways, terminals and taxiways. However this is both unpopular with the aerodromes neighbours and highly expensive. Instead, advances in technology have been adopted in an attempt to increase pilot's situational awareness in an effort to increase efficiency. Several systems have already been proposed for use on current manned aircraft, although the level of automation varies between systems. Due to the difficulties in bringing such a system to fruition, most have not yet progressed beyond the conceptual stage. However, the intentions and techniques that these systems intend to use can still be explored. Therefore, this section examines several proposed systems which are intended to automate taxiing in some way.

2.4.1 External Recovery

All surface movement for aircraft can be divided into two categories based on the method of drive; self-propelled or externally drawn. Although the term “taxiing” is most commonly applied to the act of an aircraft moving through the aerodrome under it’s own power, the term is equally applicable to any form of ground movement, even if the driving force is provided by an additional vehicle. As stated in section 2.3.2, a common method of recovering military UAS is to dispatch a ground vehicle specifically designed to operate at that airfield. As military airfields are typically single user (i.e. the ATC, aircraft and ground vehicles are all under the same chain of command) this allows manual retrieval of aircraft in a way which could not be replicated directly within civil conditions.

However, although the exact methods used at military aerodromes may be unsuitable, the use of an external recovery vehicle may have potential. Whilst the majority of civil taxiing is currently achieved under an aircraft’s own power, external assistance vehicles (i.e. ‘tugs’) are already in use at civil aerodromes . At large aerodromes, tugs are commonly used to help aircraft manoeuvre in ways that cannot be achieved with its engine set-up. As the majority of airliners cannot reverse, the most common use of tugs is backing aircraft from the gate onto the taxiway. Traditionally, tugs have only been used far away from the main runway. However, more recently several different organisations have undertaken research into using custom towing vehicles throughout the entire aerodrome.

An early example of such a system was the Automated Aircraft Towing Vehicle System (AATVS) [64], proposed in 2001. The AATVS assigns an unmanned tug to each taxing aircraft with the intention of using the tugs driving force, rather than the aircraft engines whilst on the ground. The main difference between this proposed system and traditional aircraft tugs lies in the fact that the tugs are unmanned, being able to drive to and attach onto the aircraft via remote control. Despite being unmanned, the level of automation is quite low; once the aircraft is undertow, the ground vehicle is controlled by the pilot in the cockpit. This enables the aircraft to taxi under the same authority as it would when using its own engines, only with a great improvement in fuel efficiency. As aircraft are specifically designed to be most efficient in flight, at low speeds on the ground all aircraft engines are far from their optimal design point and often run very inefficiently. The use of a custom towing vehicle completely removes this inefficiency, as the fuel supply used on the ground is entirely separate to the fuel required for the mission.

Extending this principle to using a tug to retrieve UAS would have additional benefits. Unlike manned aircraft, UAS operating on the ground will require ground specific equipment, such as sensors, processing units and communications equipment, additional to the equipment used in the air. By transferring this equipment onto a ground vehicle, the overall weight of the aircraft can be reduced. Therefore, in addition to the fuel saving made by not using the aircraft's own propulsion systems on the ground, this allows the aircraft to also be more efficient in the air.

Since AATVS was originally conceived, alternative systems which also make use of tugs have been proposed. Perhaps the most well developed system is TaxiBot [88], which has already undergone real world testing at Frankfurt Airport. As with the AATVS, the primary aim of Taxibot is to save fuel by not using the aircraft engines to taxi. However, the system differs as it intends to include higher levels of automation. Although current testing is being performed under remote control, the eventual aim of the project is to fully automate the movement of the tug whilst it is not attached to an aircraft, enabling it to navigate to aircraft autonomously. To achieve this, the system will already be capable of route planning and collision avoidance to an acceptable level. Therefore, extending the system to function with UAS would probably not require a great deal more development.

Automating the movement of the system whilst towing an aircraft could be considered the final level of automation, and is the primary goal of another system intended for automating aerodrome activities. As with the AATVS and TaxiBot systems, the anyTRACS system [11] attempts to improve ground operations through the use of automated tugs, However, compared to the other systems, anyTRACS is designed to integrate with the datalinks currently under development for aerodromes. Rather than rely on any form of human control, the intention is that all taxiing can be performed based only on instructions from GTC. Whilst still in the conceptual phase, the system promises increased safety and efficiency whist also reducing cost and fuel consumption.

This increased autonomy for ground vehicles offers additional advantages when they are considered in terms of their integration into the greater aerodrome environment. As the towing vehicles are stationed at the aerodrome, the level of customisation specific to that aerodrome can be far higher than a generic system designed to work at any airfield. This includes a more accurate 'knowledge' of the airports layout, and ensures compatibility between the system and the aerodrome in terms of communications and facilities. As it likely that the ground towing vehicle will be directed from GTC, the overall situational awareness of the control tower should be much greater if all aircraft were to use the same system. In the event of any failure in the autonomous control system, the ground vehicle could also be operated remotely by a human operator, without the need to access the controls of the UAS.

Despite these advantages over traditional aircraft-propelled taxiing, the concept of using custom towing vehicles for all stages of taxiing has yet to gather much interest. This is perhaps due to the drawbacks which come with such a system, the most inhibiting of which is most likely the extensive infrastructure required. For large commercial airports that already operate a fleet of tug vehicles, using tugs for all taxiing or even upgrading to autonomous versions might not require a huge investment relative to the airports financial income. However, many small airfields operate with a minimum of facilities, making such a system financially impossible. This would restrict the system to aerodromes that could afford it, limiting where aircraft dependant on this system could land. Not only would this impact operations in terms of practicality, it would also affect safety as it limits the ability of any dependant UAS to divert to other aerodromes.

For the aerodromes that can afford the infrastructure, there is also the practical implications of having many extra ground vehicles. To avoid leaving stationary aircraft in dangerous positions, such as on the runway after landing, the tug vehicles would need to move quickly and accurately. However, an increased number of moving vehicles will generally increase the risk of collision. Whilst this could be mitigated by driving the ground vehicles more slowly, the efficiency of the aerodrome would suffer which is generally undesirable.

Other complications of using autonomous tugs include the need to operate a sufficient number to account for mechanical failures. If the aerodrome expects to move many aircraft at a time, a great number would required and therefore this solution would be very expensive. More vehicles also increases the risk of mechanical failure during operation, as a fault in either the aircraft or the tug could stop the aircraft from taxiing. Again this could impact both efficiency and safety across the entire aerodrome. Finally, the interface between the aircraft and the tug would need to be established and universal. If not, separate systems would be required, further complicating the situation. The system would also need to be compatible with all aircraft sizes without putting undue effort into changing the design of the aircraft. Otherwise, the potential exists that the system could influence the physical design of UAS in a way which hinders the aircraft during flight.

Whilst self propelled aircraft are less efficient on the ground and potentially need to carry extra weight, they are less dependant on the aerodromes facilities. By operating more like current manned aircraft, UAS could integrate into current aerodrome operations instead of having current operations change to include UAS. Efforts to improve self propulsion are also on-going, with aircraft movement no longer limited to using the main engines. A joint project known as the Electric Green Taxiing System (EGTS) is in development by Airbus , in conjunction with Honeywell and Safran [51]. The EGTS is currently being trialled on Airbus Jet liners and aims to remove the need for main engine use during taxi. Instead, the electrical power generated by the aircraft's Auxiliary Power Unit (APU) is used to drive an electric motor embedded in the nose wheel. This allows the aircraft to be driven much like a car, propelling the aircraft using the wheels. In addition to being more fuel efficient, the aircraft is also more manoeuvrable and is able to reverse, a task that is usually only accomplished using tugs.

Despite the increased weight of the motor being carried during flight, the fuel savings during taxiing reduce the total fuel used greater than that lost carrying the motor. Should future UAS be equipped with an electrically driven nose wheel, the fuel saving element of the additional ground vehicles is no-longer a decisive advantage. Therefore, whilst a separate vehicle for taxiing is still a viable alternative for large airports, the infrastructure necessary and the disruption to current aerodrome operations puts it outside the scope of this project, and it is assumed that the aircraft will be operating under its own propulsive power.

2.4.2 Terminal Area Path Protocol

As the use of tugs has not gathered much momentum, alternative research has focussed on solutions which do not require additional ground vehicles. For manned aircraft, probably the most well defined proposal for automated taxiing is the proposed expansion of the Terminal Area Path (TAP) protocol so as to also encompass ground activities [69]. However, in order to make use of TAP on the ground, the proposal suggests a method of automating taxiing in direct opposition to the NGRM from the JPDO [55]. As outlined in NGRM, the JPDO suggests that future aerodromes would benefit from the introduction of data-link communications as it would allow authority to be delegated to individual aircraft, allowing the GTC to focus on safety critical issues. By comparison, the proposed extension to the TAP protocol aims to use the same data-links to do the opposite, giving total authority to the GTC.

The current TAP protocol is a concept recently introduced by the Radio Technical Commission for Aeronautics (RTCA) and is intended to be broadcast as part of the Ground-Based Augmentation System (GBAS) located at aerodromes. In addition to rectifying GPS position data, GBAS was designed to allow for different sorts of information to be transmitted, based on a series of message ID's. Of the 256 total possible types of GBAS broadcast, only 8 are currently defined, of which TAP is the most recent. The primary purpose of TAP is to assist landing aircraft by providing a GPS defined descent path. Whilst a straight approach with a constant descent is typical of most aerodromes, certain runways require complicated manoeuvres before landing can take place. This can include varying the rate of descent, turning and even climbing depending on the aerodrome in question. Unlike the traditional glide slope, TAP provides a four dimensional path to follow, listing both spacial co-ordinates and the time at which the aircraft should reach each point, therefore dictating its speed. The TAP data consists of ‘legs’ (straight sections) and turn radii, allowing the aircraft to line up with the runway after following a complex path. Not only is this useful as a pilot aid, it can also be used completely autonomously and has shown promise for use in UAS landings.

The proposed extension to TAP suggests that the same protocol can be extended for use on the ground [69]. As the entire aerodrome is well within the GBAS area whilst under taxi, the proposed taxi-route could be provided to the aircraft using a GPS defined path, rather than the more abstract existing method which uses semantic instructions (such as ‘follow taxiway bravo’). As such, the co-ordinates could be used to directly control the aircraft, assuming DGPS accuracy to be sufficiently high. The role of the human pilot would be much like the proposed role for future civil UAS operator, visually confirming that the system is performing correctly and providing a means of communication between the aircraft and the GTC outside of the established data-link. The human pilot would also be responsible for ensuring that the upcoming taxiway is free of collision risks, and as such a similar sensing solution would be required for any UAS.

In addition to providing a highly accurate taxi-route to each aircraft, the advantages of TAP include the ability to always have up-to-date information on the position and movements of all aircraft, as the GTC would essentially be in control. As the local aerodrome GTC is responsible for broadcasting TAP, any changes to the runway layout can easily be reflected in the GPS coordinates provided. The time based coordinate system also allows for aircraft separation to occur on the ground automatically, without the need for vehicle to vehicle transmission, or any form of autonomous taxiing.

However, the use of ground-based TAP also has disadvantages. The biggest is that the use of TAP requires all aircraft to have the same functionality. Although it could be suggested that this might be the case several decades in the future, it is unlikely to be available on all aircraft by the time UAS are ready to operate from civil aerodromes. To avoid pushing the economic cost onto the other aerodrome users, it would be better for an ATS to be compatible with current taxiing methodologies and not require active participation from other users.

In addition, of some concern is the increased workload pushed onto the GTC. Although such an activity would mostly involve computation, rather than manual human effort, the requirements for controlling all aircraft within an aerodrome would represent a significant increase in required computational power. Currently, as individual aircraft are responsible for the majority of their own navigation, the GTC simply provides path planning based on the local knowledge of the operators. In extreme cases for large aerodromes, more efficient routing can be introduced by the use of relatively simple path finding algorithms (such as time-dependent A star searches), to select the shortest path. This can then be transmitted to the aircraft as a simply list of taxiways to follow. Comparing this to TAP, using exact GPS coordinates requires a far higher level of accuracy in terms of the transmission. The need for real-time position monitoring and control feedback over time would likely result in an increase in data that must be transmitted, despite the reduction in RT. At busy aerodromes this could result in increased delays between communications due to the other aircraft requiring updated TAP information.

Finally, although TAP could be used to improve the routing information for aircraft, it cannot be considered a full ATS. As the GTC cannot directly monitor the entire aerodrome, all aircraft must also continue to assess whether the TAP route is safe and independently decide on whether the instructions should be followed. As such, although off board planning reduces the computational burden on the aircraft, it does little to actually reduce the overall computation required.

Removing computation from the aircraft also introduces potential problems during communication system failure. Although the control of the aircraft is performed on board, communication failure prevents the aircraft from receiving the route it needs to follow. For manned aircraft, the human pilot can simply return to aerodrome maps and navigate manually. However, any UAS specifically designed to function with TAP may be entirely unable to function without the TAP communications. If additional capability were added specifically to allow UAS to continue in these situations, the majority of development towards a system which does not depend on external data would have been completed. Therefore, a better approach would seem to increase the capabilities aboard the UAS, rather than rely on even more external control.

By keeping the autonomy internal to the UAS, the system will generally have to carry more equipment in the form of sensors and processing hardware. However, it has numerous additional benefits over a distribute approach. By designing the system to interact with other aircraft and the GTC like a human pilot, the system should be able to work anywhere, without any specialised equipment on the ground. In addition, the need for high bandwidth can be mostly negated. As much work has already gone into designing autonomous ground vehicles, ideas which are common can be extracted and applied here, simplifying the overall design process required.

2.4.3 Low Visibility Assistance System

In complete contrast to the proposed use of the TAP protocol as outlined above, the LVAS developed by National Aerospace Laboratory of the Netherlands (Nationaal Lucht- en Ruimtevaartlaboratorium) (NLR) has been developed in line with the predictions for future aerodromes laid outlined in the NGRM [18]. As the name suggests, the system has been conceived for use on manned aircraft when inclement or foggy weather interrupts airport operations. When weather conditions worsen, the visibility for GTC staff in the the control tower decreases. Without being able to see the aircraft on the ground, operators must instead confirm the location of aircraft via radio telephony, a process which takes far longer than looking out of the window. In addition, taxiing aircraft must closely monitor radio transmissions in order to be aware of other aircraft that might be concealed by the fog.

The purpose of the LVAS is to provide greater levels of information directly to the aircrew. This is achieved using a combination of A-SMGCS, AMM and CPDLC. By using all of these systems together, the pilot's situational awareness can be greatly improved.As the LVAS cannot control an aircraft directly it is only a pilot aid, rather than an ATS. However, the system architecture required for LVAS is comparable to an UAS operating purely on data-links and without voice communication capabilities. As such, many of the techniques used to create this system are essentially what would be required for a full ATS.

LVAS is included here as it is the most complete demonstration of the improvements in situational awareness which can be introduced through the use of data links. After it was developed, the capabilities of LVAS were tested through simulation, using a virtual airliner controlled by experienced human pilots [18]. Simulating low-visibility conditions and a busy aerodrome, the pilots were tasked to taxi through the aerodrome without direct verbal RT communication with the GTC, instead relying only on information obtained through data-link communication via LVAS. To complete this kind of task, LVAS incorporates many different systems.

As direct contact with the GTC was avoided, routing information was not provided. Instead, LVAS uses path planning algorithms to calculate the most efficient route, using the Integrated Aeronautical Information Package (IAIP) database to understand the airports layout. The map data is also used to aid in interacting with other aerodrome users. Using the data obtained via CPDLC, the positions of known aircraft were used as part of the path planning process to attempt to avoid entering conflicts. If the aircraft gets with close proximity of others, Closest Point of Approach (CPA) algorithms are used to predict potential conflicts. CPA operates by defining a safe zone around the aircraft and then alerting the pilot if any other aircraft is predicted to intersect this area. To allow for minimal clearance at busy aerodromes, accurate size data would required for all aircraft. However to simply ensure safety when dealing with aircraft of various sizes, a sufficiently large safe zone was used instead. The CPA system then warns the crew to varying levels depending on the remaining time till potential collision. Where possible, the system also gives advisory information based on the type of conflict which is occurring:

- Crossing
  - Occurs where aircraft are attempting to cross paths.
  - Can be resolved by obeying the rules on who has right of way.
- In trail
  - Occurs where a following aircraft catches up to the leader
  - Should primarily be resolved through speed adjustment. Where there is sufficient room overtaking is viable, whereas for broken down aircraft, re-routing is a required.
- Head-on
  - Occurs when aircraft travel towards each other on the same taxi way.
  - If the taxiway is not wide enough to allow aircraft to pass this conflict cannot be solved as most aircraft cannot reverse. Following one-way rules in planning and communication with the GTC should prevent this from occurring.

Both the route planning and CPA collision advisory systems were shown to be of great benefit to pilots, providing an efficient route whilst also preventing any risk of collision with known aircraft. However, a current limitation of LVAS is that all guidance and collision risk abilities are based on data provided through CPDLC. As such, only aircraft broadcasting their own position were known to the system, requiring the pilot to continue to visual identify any other collision risks.

As non-cooperating aircraft are likely to continue to exist for many decades, future civil UAS will also need a method of detecting unknown collision risks without relying on data-links. However, a large advantage of UAS is that a single system could be used to combine all the relevant data. During the aforementioned simulation, the visually detected aircraft were avoided by one system (the human pilot) whilst the prior-known aircraft were avoided by another LVAS. For an UAS which uses both communicated data and sensor based aircraft detection, all collision risks could be tracked using a single system, with a singular avoidance method (such as CPA) used for all risks.

In scenarios where the GTC did communicate verbally, the paper concludes that using the LVAS, pilots can operate in low visibility conditions nearly as efficiently as in good weather. As such, LVAS has shown the benefits of increased situational awareness through data links and an autonomous taxiing system using these features is a viable option. The remaining challenges involve integrating a sensing method that is just as effective at identifying collision risks as a human pilot. Once this is resolved, the situational awareness element of an ATS system could be considered complete, although such a system would remain dependant on high level of external data.

2.4.4 Airbus Automation: OANS and 'Trains'

The previous systems are pilots aids which simply provide data. Before moving to full automation, a possible 'stepping point' are 'shared-authority' systems, with both the human pilot and an automated system sharing some responsibility for control. As such systems require a high level of avionics, current proposals for shared authority systems are designed to build upon the most modern of avionics for large airliners. Focussing specific on Airbus, the following proposal is intended to become part of the Onboard Airport Navigation System (OANS) currently installed on the A380 and A350, which essentially combines a live aerodrome map with other data, much like AMM.

The proposed system from Airbus [96] is designed to help alleviate aerodrome congestion without modifying the aerodrome layout. Delays at aerodromes often occur while aircraft are waiting to taxi. As current pilot situational awareness is limited to what can be observed from the cockpit, large safety margins and extended periods of observation are used by pilots to avoid collisions. This delay is further extended as pilots often perform additional tasks whilst taxiing. When following other aircraft, pilots often allow a larger separation distance than is required, in order to provide adequate reaction time should the lead aircraft do something unexpected. As such, Airbus states that there is usually sufficient room for more aircraft, but due to inefficient taxiing procedure the taxiways are often artificially saturated.

During current aerodrome operations, a common instruction given by GTC is to follow another aircraft. For manned aircraft, pilots currently perform this action visually, recognising the aircraft by its type and registration and then following it. By automating the act of following another aircraft, both the pilot's workload and the distance between the aircraft can be reduced, allowing more aircraft to taxi simultaneously. The proposed system functions by grouping aircraft into 'trains', with multiple aircraft following a leader. Within each train, the lead aircraft remains under pilot control, with the speed, heading and position of the leader broadcast to a central control system before being relayed to other aircraft. The system has two proposed level of operation:

- For the most basic level, the separation distance and closing speed of following aircraft are monitored by the central system. By altering throttle settings, a safe separation distance is automatically maintained from the preceding aircraft, reducing the pilots workload and the overall length of the train.

- At the higher level, it is proposed that the following aircraft would also steer themselves, following the same route as the lead aircraft. It is assumed that the lead aircraft followed a safe route and therefore subsequent members of the train should be safe to follow that route as well. To stop scenarios where large aircraft attempt to fit through small gaps, aircraft larger than the current leader will become the leaders of a new train.

Additional elements of the system include the flexibility to allow aircraft to leave and join, as well as the ability to include non-cooperating aircraft; provided the aerodrome could accurately track the movement of a non-cooperating aircraft (something which is available at some large aerodromes) other aircraft could follow behind it. The main drawback to implementing 'trains' is the immense large of infrastructure required, both at the aerodrome and on board each aircraft. In addition, as all aircraft receive control commands from external sources, there are also safety concerns over data loss and security vulnerabilities.

For unmanned aircraft, the concept of having the UAS follow a lead aircraft is a different method of achieving autonomous taxiing. If an ATS were to replicate the abilities of a human pilot in order to operate at a civil aerodrome, the ability to detect and track other aircraft would be essential. If speed and position data for specific aircraft were made available via CPDLC, the UAS could follow a lead aircraft much more easily, as suggested in this system. This technique also has the obvious drawback of requiring another vehicle to be both present and heading towards the same destination. As this is not guaranteed, this method can only be used in certain conditions. However, where it can be used it offers a far simpler method than having the aircraft attempt to perform all actions of a pilot by itself. Therefore, this proposal to use lead vehicles opens a new method of autonomous UAS taxiing. Rather than having the UAS perform most of the calculations, following a lead aircraft could reduce most of the difficulties. However, the obvious shortcomings of this approach would be if there were no other aircraft to follow, or if the UAS itself were to be 'leading' a train.

2.4.5 UGOMS

As the previous examples have alluded to, despite ongoing efforts to automate taxiing for both manned and unmanned aircraft, there are very few examples of a complete system. From a review of literature, only one publicised example of a system designed to completely automate military UAS taxiing has been found. The Unmanned Aircraft System Ground Operations Management System (UGOMS) is currently the most advanced automated taxiing available, and represents the cumulative developments of several different parties over a number of years. Despite being the only notable example, the system is well formed and is capable of performing most of the functionality required for automated taxiing [42] [32].

Citing the impact of delayed communications on remote taxiing as the primary motivator, UGOMS is intended to be a taxiing system which works without direct operator input. In addition to automated taxiing, the system also includes many of the requirements for autonomous taxiing, in the case of external communications failure and GPS loss. These capabilities are provided through direct sensing, which is achieved using a computer vision system. The methods used when designing a machine vision system are dictated by what data is required. For example, a key function of UGOMS is to enable military UAS to continue taxiing in the event of GPS denial [32]. Without GPS data, machine vision is used to determine the aircraft's position through visual comparison of surrounding terrain to known map data.

As UGOMS was designed for military use, many elements of the system are incompatible with civil usage. However, as this work is similar in both hardware and scope, UGOMS provides examples of collision risk detection technologies which have been previously implemented. As such, UGOMS will be examined further in Section 4.3.2.

2.5 Ground Infrastructure

From the section above, it is clear that many of the proposed improvements to aircraft ground operations rely on the introduction of additional equipment at aerodromes: either ground based navigation aids or external recovery vehicles. For UAS, the reliance on external equipment is nothing new. Following the rest of the aviation industry, this work uses the term Unmanned Aircraft System(s) (UAS) rather than the somewhat older term Unmanned Aerial Vehicle (UAV), in order to convey that the system is comprised of more than the aircraft itself. As all UAS are dependant on observers and operators outside of the aircraft platform, the hardware available on the ground can be just as important as the airborne systems. Similarly, additional ground infrastructure has been used to improve capability at aerodromes for decades, with technologies such as 'Autolanding' being common at most large airports around the world.

It could be argued that if UAS are only expected to operate from certain aerodromes, an ATS could be tailored specifically for these locations. Systems such as A-SMGCS currently rely on aircraft communicating their position via data-link. However, several aerodromes are also incorporating Surface Movement Radars (SMR) and machine vision systems within A-SMGCS in order to track the movements of all vehicles on the ground [29]. If this information was provided to UAS, the need for active collision risk detection is greatly diminished as the GTC would be expected to provide the information to the aircraft.

However, there are also disadvantages to relying on systems external to the aircraft. As with other ground based systems, the financial requirement for the installation and upkeep of ground equipment would likely be the responsibility of the aerodrome. When civil UAS are first introduced, it is unlikely that all airports will immediately allow UAS access. Smaller aerodromes with less traffic are likely to offer UAS access first, but may lack the budget required for the equipment.

In addition, all aircraft operating in civil airspace must follow established regulations. If UAS can only operate from facilities with ground based equipment, the established principles of diverting to another aerodrome would only be possible if the other aerodrome were also suitably equipped. Furthermore, it would also be unwise to design UAS which are dependant on specific systems which might never be adopted. Therefore, aerodrome based equipment has not been considered, and the challenge of UAS taxiing must be met by a capability improvement on UAS themselves. This work will be undertaken with the current systems and procedures in place at civil aerodromes.

2.6 Conclusion

This chapter has provided an overview of the current state of UAS ground operations, with the intention of providing context for the rest of the work. By reviewing the basic regulations that all aircraft must obey within a civil environment, it is easy to conclude that the methods currently used by UAS are unsuitable for use at civil aerodromes. As present-day UAS are segregated from manned aircraft whilst on the ground, many modern UAS do not use standard operating procedure; instead using retrieval methods which are unconventional and incompatible with the aforementioned regulations. As such, civil UAS will need to use different methods. The inability to operate in non-segregated aerodromes represents a large barrier to bringing UAS into the civil airspace, with automated taxiing and aerodrome operations already identified as a research gap within Europe [36].

Focussing on ground operations within civil aerodromes, present operating procedures continue to rely on manual control, with very little automation used during taxiing. As a result, in contrast to earlier efforts to bring airborne automation to UAS, there are no existing ground based systems to simply copy from manned aircraft. In addition, although the potential benefits of increased automation has resulted in several potential systems being proposed for civil UAS taxiing, most are unsuitable due to there fundamental concepts involving a human pilot. Due to civil regulations requiring a pilot to communicate and make decisions, many of these systems do not represent entire ATS but instead are more highly advanced pilot aids and could not be simply adopted by UAS.

Ignoring the incompatible operating procedures, the most 'feature complete' ATS are currently designed for military UAS. Many such systems can already perform automated taxiing, relying on satellite based navigation solutions. However, the complexity of the civil aerodrome environment precludes a simple automation system based on position alone. Although military operators can function with only external observers for collision avoidance, such a system would not be suitable for civil aerodrome use. Despite rapid changes in technology, the interaction with legacy systems and none-cooperating users limits the ability to rely on a single system to govern the entire aerodrome.

Moreover, relying on radio-navigation as the sole provider of location data is potentially dangerous in the event of communications failure (or delay). As DGPS offers no method of position confirmation, even static collision risks cannot be avoided if external-localisation systems are incorrect. Furthermore, a shared-autonomy approach (such as GPS driven taxiing with human collision risk detection) is also insufficient as any approach which relies on external control can be considered unsuitable.

As conventional aircraft could also benefit from automated taxiing, the potential exists to create a universal system which could be used by all aircraft. However, the large variety of civil aerodrome users again makes this difficult. As many civil aircraft operate with minimal equipment, implementing any system which relies on all aircraft and aerodromes upgrading to said system is infeasible. Instead, in order for UAS to operate alongside safety, it is more practical for the capability of pilots to be replicated. For the system to operate in any aerodrome layout necessitates both flexibility and adaptability, with a generic approach capable of responding to any problem, regardless of local equipment levels. The inability to rely on systems external to the aircraft indicates the strong need for an ATS on board a UAS. The transition to automated taxiing will require a system with many of the capabilities currently provided by the human pilot. Rather than relying on ground based infrastructure, an ATS contined entirely onboard the UAS is the most suitable option.

2.6.1 Research Gap

As UAS already require many sensors to operate, a range of equipment can be considered available as standard (i.e. DGPS, compass, airspeed sensors). Of these sensors, only Global Navigation Satellite System (GNSS) (such as DGPS) is designed to provide position information. As previously stated, depending solely on external position data is dangerous as there is no form of validation to confirm this information to be correct. In addition, amongst typical avionics there are no sensors capable of detecting obstacles for collision risk detection. These problems could be overcome by including highly specialised sensors, such as Laser Imaging, Detection and Ranging (LIDAR), in addition to ground based localisation tools designed to track the movement of any UAS.

However, the barriers which prevent civil UAS ground operations are not only technical, but also economic. The inclusion of new equipment not only increases the upfront cost of purchase, but also represents additional payload which will continue to increase operating costs throughout the lifespan of the UAS. To work around this problem, the simplest solution is to only make use of the equipment available on board an UAS.

Furthermore, to prevent UAS from becoming limited to only using certain aerodromes, the solution should be capable of undertaking automated taxing without any additional ground equipment.

Aside from conventional avionics, the only other form of sensor commonly carried by UAS are cameras; included to provide additional information to the remote pilot. Although not original designed to be used as a sensor, images are extremely data rich and a forward facing camera is likely capable of performing both collision avoidance and localisation.

In addition, using the camera to extract information has additional benefits. Beyond confirming localisation and providing collision risk detection, a camera can be used to extract a wider variety of information. In 2004, the Joint Aviation Authorities (JAA) stated that a large risk with UAS in civil aerodromes was the inability to convey visual information in the same manner as a pilot. For civil taxiing, visual information such as waiting lights on runway entrances and surface markings convey much of the required information. The use of surface markings at aerodromes is so important that the Group of Airport Safety Regulators (GASR) performed a study to determine the impact of UAS who cannot follow the lines [103].

Therefore, this work aims to improve the ground operating capabilities of UAS solely through the inclusion of visual data in addition to existing UAS avionic systems. The research question is whether it is possible to replicate (or exceed) the capabilities of human pilots in aerodrome operations, through mimicking the visual capabilities of human pilots with a single forward facing camera.

Chapter 3

System Level Study

The goal of this work is to produce an Automated Taxiing System (ATS). Although the term ATS infers a single system solution, many elements are required to function together in order to ensure safe aerodrome traversal. Therefore, rather than a single algorithm or technique, the ATS should consist of multiple subsystems working together to achieve the final goal. This chapter will present a system level study of the ATS, briefly outlining the approach being taken, the required sub-systems and how they interact. As the creation of a full automated taxiing system is beyond the scope of a single thesis, this chapter also highlights elements which will not be explored, along with the methods used to work around their absence.

3.1 Outline of the problem

The problem that will be addressed in this thesis is how an UAS can navigate from one position in the aerodrome to another, without direct human control. If the aircraft was simply a large ground robot, established techniques from the field of mobile robotics could provide a solution; as all large UAS operate with external localisation aids (such as GPS) and all civil aerodromes should be well-mapped, such a task could be accomplished using a simple steering control strategy based on current position and compass heading. However, although such a method has been used to achieve military UAS taxiing (in which external human observers keep constant watch for collision risks) this approach would be far from appropriate for civil operations. For this work, the problem is not how to achieve the manoeuvre, but how to do so both safely and reliably.

This is a difficult problem for several reasons. Despite being far more constrained than other real-world environments, aerodromes are still uncontrolled, in that the surroundings and users change without input from the UAS operator. Although permanent fixtures are unlikely to change and can be avoided using only map data, the chance of colliding with other obstacles is too great to assume a clear path from start point to target destination. As such, additional sources of information will be required which convey the position of other collision risks.

Standard avionics are somewhat lacking in this area. Although well equipped aircraft can utilise technologies such as radar data and Traffic Collision Avoidance System (TCAS) whilst airborne, there is minimal support between aircraft on the ground. Where such support does exist (usually via ADS-B), only other similarly equipped aircraft can broadcast their own location, leaving all other potential collision risks as unknowns. In addition, dynamic objects (such as ground vehicles and pedestrians) and static objects (which are present but are not on the aerodrome map) will also need to be avoided.

Therefore, as external data sources and prior data cannot be relied upon, direct sensing of the environment is required. Again, standard avionics usually lack short range sensors, as they are of limited use whilst in flight. As any additional sensors will add weight to the aircraft, the viability of sensor types must also include their mass. As any increase in the payload mass of an aircraft mass decreases range and endurance, some sensor types may be ineligible for use due to their impact across the entire operational role of the aircraft. Which ever sensor is used, the chosen approach must be highly robust and able to handle both typical conditions in addition to unexpected objects.

Due to the safety implications of an aircraft moving through a civil aerodrome alongside conventional aircraft, the system must assert with high confidence that the route is safe. Irrespective of the sensing equipment used, automated taxiing requires more than basic feature extraction. After potential collision risks have been detected, interpretation and situational awareness will be required. Rather than relying directly on the instantaneous output of a sensor, the system must incorporate a re-observation and tracking methodology, which creates a form of situational awareness. This will require fusing data from all possible sources into a coherent system.

Finally, in addition to the requirement to detect and avoid collision, the position of the aircraft within the aerodrome must also be validated. Although localisation could be achieved through GPS alone, GPS accuracy is highly variable depending on many factors. GPS signals are very weak and as such are susceptible to signal segregation, localised interference or even malicious manipulation. Originally only accurate to within 100m, advances such as DGPS have increased accuracy to within 5m [76]. However, this is still sufficiently inaccurate for an aircraft to collide with a static structure, such as a building. If only a single data source were relied upon for position data, any inaccuracies in this data could have dangerous consequences. Although direct sensing could be used to prevent collision with static risks, other aerodrome users rely on aircraft maintaining correcting positioning for safety. In the most extreme, a UAS with inaccurate position data could commit a runway incursion, becoming a collision risk for landing aircraft.

Instead, to produce a robust and accurate solution, the navigation problem should be solved through the use of as many sources of data as possible. Therefore, in addition to GPS, the vehicle's own sensors will be used; with the intention to confirm or improve upon external position data. Although it is not expected that an UAS will ever need to taxi without any form of external assistance (as the majority of UAS cannot fly in GPS denied environments), the intention is that this work should allow for operation without GPS data, but will mainly seek to simply build upon it.

3.2 Research Scope

As both collision avoidance and localisation depend upon direct sensing, the research scope is already narrowed to identifying objects within the environment and determining their position relative to the UAS. In addition, the scope of this research is further narrowed by the following factors, which will be discussed below:

- UAS operational area within the aerodrome
- Regulatory restrictions
- Hardware available on board the UAS

3.2.1 Operational Area

In aviation, the term 'ground operations' is used to define the activities of aircraft which take place on the ground; including maintenance, loading and refuelling. As these activities are often aircraft specific, they are not simple to implement in a generic manner. As such, for this work the term Automated Taxiing System (ATS) has been chosen, as the purpose of the system is specifically related to taxiing rather than ground operations as a whole. Furthermore, the physical task of taxiing also extends slightly beyond the regulatory definition of ground operations. As any aircraft positioned on an active runway is still a risk to other aircraft attempting to land, ground operations do not technically include aircraft movements on the runway, which are instead considered to be part of the flight. Despite this technical distinction, the method of taxiing does not differ whether an aircraft is on or off an active runway. As such, this system aims to solve the taxiing problem regardless of where an aircraft is within an aerodrome.

3.2.2 Regulatory Restrictions

With the predicted introduction of civil UAS still many years away, it is likely that widespread use (and therefore the requirement for a full ATS) could still be decades in the future. This firmly suggests that the work undertaken here is early stage research, and as such should endeavour to investigate the most contemporary methods available. In recent years, data fusion and artificial decision making has improved dramatically through the use of machine learning techniques, such as ANN. Related applications, such as Self Driving Cars (SDCs), have already demonstrated the use of ANN for urban scene interpretation and response [8].

However, aerospace safety regulations require that all aspects of control algorithms are understood. As the internal functionality of the algorithms is typically hidden from human users, regulatory approval of machine learning algorithms (such as ANN) for aerospace use is unlikely. Similarly, non-deterministic techniques, where the output is not entirely dependant on the input, are also unlikely as the end-result cannot be guaranteed. The use of machine learning and non-deterministic systems has already been identified as a potential barrier to entry for new technology. From [116]:

"*Existing adaptive/ non-deterministic algorithms have not been widely applied to safety-critical civil aviation applications in part because of the lack of a mature process for designing, implementing, and testing such algorithms.*'

As such, these kind of methods are difficult to propose for use on an UAS when there is no certainty that they could ever be used. Although at this early stage it may seem unwise to limit the work based on unknown certification requirements, a grounded approach stands the best chance of producing a viable technique that could see real world use. As such, this work will endeavour to use algorithms that are deterministic and fully explainable. For this reason, techniques which use artificial learning have not been investigated.

A key element in the use of deterministic methods is that the system can be considered "automated". When given the same input data, a deterministic system will always respond in the same way. As such although the system can make make decision outside of the pilots input, it cannot ever make decision outside the scope of intended functions. This form of highly adaptive but deterministic automation is known as specialised intelligence (designed for solving a particular problem). No form of general intelligence (i.e. artificial intelligence) will be explored in this work.

3.2.3 Aircraft Equipment

As the research required to bring UAS to civil airspace is expected to take many years, the hardware and capabilities of future civil UAS are unknown. While it is possible to make predictions based on current research, this may result in solutions that rely on expected developments which never come to fruition. Therefore, it has been decided to work under the assumption that any automated taxiing system will only make use of current aircraft hardware.

Although it is not implicitly stated in the aim section, as an Automated Taxiing System (ATS) is essentially a piece of software, the system should be generic enough to be deployed onto any unmanned aircraft. This can be achieved by keeping the research as *platform independent* as possible; avoiding the use of hardware which may be unavailable on certain UAS aircraft and instead using only systems which are common to the majority of UAS. Conversely, assumptions should also be made about the minimum level of equipment. Current manned aircraft are produced by a variety of different manufacturers and carry a range of equipment on board. Despite it being likely that future UAS will emulate the same pattern, unmanned aircraft are slightly more constrained as the lack of a human pilot necessitates a minimum equipment level. In order to navigate while airborne, all UAS must be equipped with a wide array of avionics, including Inertial Navigation System (INS), airspeed sensors and a GPS receiver. Therefore, this equipment is considered available for this work. In addition, unlike some current military UAS, the ATS will be designed to only work with aircraft capable of safe self-propelled taxiing.

3.3 Direct Sensing Equipment

When considering the equipment available to an aircraft, most of the hardware required to achieve automated taxiing functionality is already present (e.g. undercarriage, lighting, propulsion and communications equipment can all be assumed to be present on any operating large-scale UAS). Despite this, all current taxiing methods (both manned and unmanned) rely on human operators to provide functionality that is difficult to replicate. Aside from verbal communication (which is discussed below in section 3.5), this is usually in the form of collision risk detection and position confirmation, both done through visual inspection. In order for the ATS to function with human oversight rather than direct input, it must replicate the human capability to observe and understand the surrounding environment. As such, a mixture of sensors and interpretative algorithms are required.

Assuming that future UAS will not be overly dissimilar to current aircraft, a wide variety of sensing equipment is already present on-board. However, current sensors are typically designed for airborne use and are not suitable for use on the ground. The majority of sensors used by current military UAS are either related to mission activities (commonly referred to as Intelligence, Surveillance, Target Acquisition, and Reconnaissance (ISTART) equipment) or the operation of the platform itself [89]. To conserve weight and volume, sensors are often employed in more than one activity at once. For example, cameras used for surveillance are often also used by the pilots when commanding the aircraft remotely.

A potential approach would be to include sensors specifically for ground operations. Active sensors (such as LIDAR) are being introduced for similar purposes in self-driving cars, due to their output being easily used for collision risk detection and even localisation. However, active sensors are typically energy intensive and add additional weight which impacts a vehicles performance. In addition, as LIDAR operates on line of sight, the sensor is typically positioned as high and far from the rest of the vehicle as possible in order to prevent self obstruction. As the size of the LIDAR unit is usually proportional to its detection range, larger units are able to use higher power and sense further; with LIDAR systems intended for outdoor use often being bulky and heavy, consisting of a rotating tower that protrudes from the vehicle. Although smaller LIDAR systems are available which could fit within even a small UAS airframe, the detection range of small LIDAR is often limited, such that any detected collision risk would already be very close to the UAS.

Therefore, although the output from a LIDAR sensing would provide data highly appropriate for automated taxiing, it is unlikely that a small UAS will sacrifice flight performance for a system that can only be used on the ground and hampers the aircraft aerodynamically. As the time an aircraft spends on the ground is only a secondary consideration, and as certification issues are generally prohibitive to new technologies, it is unlikely that UAS will be fitted with this kind of sensing equipment, purely to improve their ground capability.

3.3.1 Cameras and Machine Vision

As active sensors are unlikely to be available, a different form of sensor must be selected to use with the ATS. The alternative to active sensors are passive sensors, which rely on external stimuli to gather information. As with active sensors, passive sensors depend on specific mediums to function (i.e. microphones depend on sound, radiometers depend on radiowaves etc). However, as the sensors only take in data, rather than output a signals, they are usually far smaller, lighter and less power consumptive than active sensors. The downside to passive sensors is that they require far more interpretation of the received data. For active sensors, the difference between the outgoing and returning signal is used to determine information about the environment. For passive sensors, the incoming data is entirely outside of the sensors control and must be examined in full.

For remote sensing, the most common type of passive sensor are cameras. With the only output from a camera being a standard two-dimensional image, a processing system must be associated with each camera for interpretation, leading to the term ‘machine vision’ (or computer vision). Although originally used in controlled conditions, machine vision has proven to be extremely reliable and is now being used in increasingly varied environments, with recent production road vehicles (such as Tesla Motors) fitted with a camera and using machine vision techniques to provide ‘autopilot’ functionality [61], a large step towards fully functional SDC.

An advantage of machine vision is that images are extremely data-rich. Therefore, different techniques can extract significantly different data from the same image. As the principle of machine vision is to mimic human visual interpretation, theoretically any visual detection task performed by a pilot can be replicated. This suggests that a camera-based system could be used to directly replace human interpretation.

Visual sensing has additional benefits when used for aerodrome operations, due to large amounts of data being transmitted visually to pilots. As stated in Section 2.1.3, aerodrome regulations require information about how an aircraft should taxi to be conveyed using signals on paved runways and taxiways. Ground markings are defined by colour, with white markings used on runways and yellow markings used for taxiing. In addition to the centre line, taxiway ‘Stop Bars’ (where an aircraft should hold and wait for GTC clearance) are also denoted by surface markings, indicating how visual detection is required.

Unlike other sensor types, visual cameras are already in extensive use on unmanned aircraft. As current UAS are operated as RPV, cameras are used to relay images back to the human operators to interpret. Therefore, the majority of UAS are already suitably equipped. As computer processors become ever more powerful yet smaller in physical size, on board interpretation (i.e. machine vision) may not even require any change to the UAS. The main hardware concern with using machine vision is that the cameras must be positioned to provide good visibility during taxiing. As on-board cameras for UAS are primarily designed for airborne observation, the main cameras are often under slung and downwards pointing. As such, an additional forward mounted camera may be required, although any UAS which has previously been taxied remotely is likely to already have such a camera.

For manned aircraft, situational awareness is being improved by using camera systems mounted in areas which are hard for the pilot to see, such as in the tail, undercarriage and wing tips. With the increasing simplicity of digital cameras and the increased size of airliners operating at aerodromes, the NTSB has recently recommended that all large aircraft should be fitted with cameras on wing tips to help pilots see potential collision threats [50].

The main disadvantage of monocular cameras is the absence of depth data. For automated taxiing, the position of the aircraft relative to other objects is important. Sensors which provide depth data directly allow collision risks to be detected and avoided without any need to classify the information. In contrast, when compared to active sensors (such as LIDAR) monocular cameras require significantly more processing of their output.

Alternatively, stereoscopic cameras can be used. With wing tip collision representing one of the most common incidents in all of aviation, there has already been an extensive investigation into the use of computer vision to prevent collision [44]. Using wing-tip mounted cameras in a stereoscopic camera set-up, it has been shown that the position of objects captured by both cameras can be estimated. As the range of stereoscopic depth estimation is dependant on the separation between the cameras, the wing tips of aircraft could provide a method of long range distance estimation. As such, implementing a similar stereoscopic system on a UAS could provide the information required for collision risk detection.

However, requiring that all aircraft are fitted with wing tip cameras would place a hardware restriction on the system. Instead, to ensure compatibility with as many different UAS types as possible, it would be better to assume the more common configuration of a single nose mounted camera for taxiing. During movement, each aircraft is only responsible for maintaining it's own position and avoiding collisions with other vehicles in front of it. As such, a forwards facing camera should be sufficient, provided that other aircraft behind the UAS are equally capable of avoiding collision.

Therefore due to their small size, low weight and ease of installation, cameras will be assumed to present on all UAS and will be considered the main sensor in the investigation. It should be noted that when relying purely on passive visual cameras, such a system would only be useful in good lighting conditions during the day. Although machine vision can be applied used with infra-red cameras (and therefore could be used at night) for the sake of compatibility this has not been considered. In addition, in order to capture more information, high resolution cameras will be assumed. As high resolution images contain more information, this increases the processing required, reducing the ability for the system to operate in real-time. However, as this system is designed for future UAS, an increase in computational power can be expected and real-time operation will not be a main focus of this work.

3.3.2 Test Equipment

As part of the ongoing efforts to bring UAS into civil airspace, many nations have created organisations to assess the viability of civil UAS. Prior to its premature finish in 2015, the Autonomous Systems Technology Related Airborne Evaluation and Assessment (ASTRAEA) programme was the main project within the UK [12], with the aim to *"enable the routine use of UAS in all classes of airspace without the need for restrictive or specialised conditions of operation"*. Although the final goal of ASTRAEA was to demonstrate functioning civil UAS, all ground based movements were expected to be controlled by remote human pilots and as such no work was undertaken for automated ground operations [53].

However, some benefits from ASTRAEA survive to be of use to this work. Safely testing technologies on full scale UAS requires access to segregated airspace and large financial investment. For ASTRAEA, established UAS manufacturers, such as BAe Systems and Thales, were expected to undertake much of the testing. This work has been undertaken in conjunction with BAE systems who have provided the practical test data. As a result, the has been tailored towards the test-platform, the BAE Jetstream surrogate UAS, shown in Figure 3.1. Matching this hardware, this work assumes that direct sensing capabilities are limited to a single forward facing monocular camera.

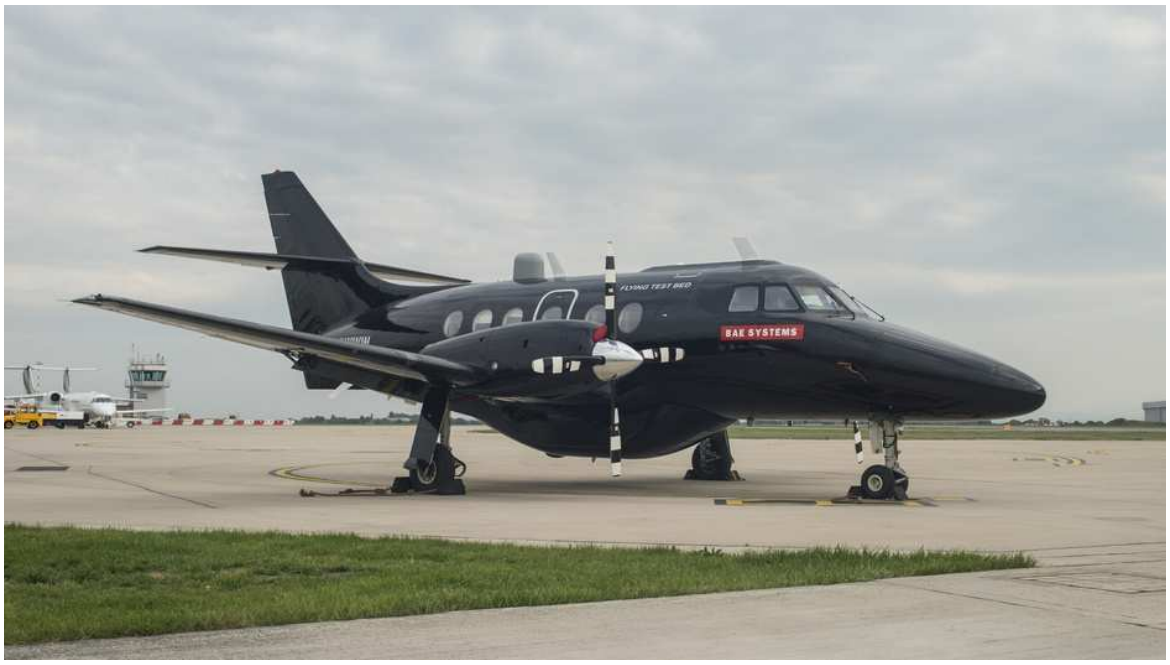


Fig. 3.1 BAE Jetstream Surrogate UAS

Although it is theoretically possible to produce a sufficiently capable machine vision system to replicate all elements of human vision, this would represent an enormous undertaking and far exceed the requirements of this project. Therefore, to prevent the system from becoming overcomplicated, it is essential to first determine what is required for the system to operate successfully, then restrict the computer vision algorithms to only extract this data. Suitable data must be reliable, accurate and pertinent to the UAS operation. In order to facilitate this, a separate literature review of machine vision techniques will be undertaken in Chapter 4.

3.4 Navigation

Accurate navigation (i.e. positioning the UAS correctly on the taxiway) is an essential part of aerodrome operations. If an aerodrome is compared to a typical urban environment as faced by SDC, roads are far busier and are far more dangerous (in terms of statistical life loss). However, the possible loss of life from a single collision at an aerodrome far exceeds the danger from the majority of situations encountered by road vehicles. Therefore an extremely high level of safety must be maintained.

Navigation of aircraft also must deal with dangers beyond physical contact. Whilst ground vehicles are generally driven by wheels, aircraft use propulsive methods such as propellers and turbofans. Not only are propellers dangerous directly, due to the high speed at which they rotate, but these engines all produce thrust that can be a risk to anything that is immediately behind the aircraft. To avoid danger, routes around aerodromes are often planned out to mitigate these effects. Aircraft which reach their destination but do so using an unorthodox route may represent a danger to other aerodrome users, if not themselves.

As all UAS require many different avionics systems to operate, GNSS ()such as GPS) are assumed to be available. However, the accuracy of these systems can be highly variable, primarily depending on the level of local augmentation available at the aerodrome. As such, the accuracy of these systems, and other methods of position estimation, are outlined below. Not included within this section is the use of inertial sensors. Whilst critical in ensuring that aircraft are stable and useful for dead-reckoning in the event of GPS failure, the tendency to drift makes inertial sensors less useful as a primary data source. As GPS is assumed available, inertial sensors will not be investigated for aerodrome navigation.

3.4.1 Map Data

Although knowing the exact location of the aircraft is essential, this data on its own is not useful. Navigation relates to the system being able to determine the location of the aircraft with respect to the rest of the aerodrome. Therefore, in order for the system to be capable of navigation, the system must also hold data on the layout of any aerodrome at which it might land. Fortunately, as aerodromes are already charted for human use, the majority of data is already publicly available in digital format. For example, ICAO publishes Aerodrome Charts which display the global co-ordinates of all stands, taxiway numbers, markings and hold points at an aerodrome. Due to the size and complexity of aerodromes, pilots routinely rely on aerodrome maps in order to navigate.

For general aviation, these maps are often printed and inspected manually by pilots. However, for better equipped aircraft aerodrome map data is often stored as part of the avionics, and shown on a Multiple Function Display (MFD) in the cockpit, forming the basis of the previously mentioned AMM. As map data for nearly all aerodromes is already available and digitised, migrating this data for UAS use should be fairly trivial. In fact, several existing examples of automated taxiing systems for UAS already make use of digitised map data (discussed in greater detail in Section 2.4).

3.4.2 GPS and Ground Based Augmentation

For current manned aircraft, the lack of automation requires the human pilot to maintain control. Although GPS data is available, it is currently only used as reference. As such, GPS accuracy needs only to be good enough to inform the pilot of the aircraft's general location, as maintaining the correct position on the taxiway remains a task to be performed visually by the pilot. Without the need for high accuracy, pure GPS (i.e. satellite signals only) is sufficient. Although pure GPS is only accurate to around 15m, taxiways and runways are typically large enough that this has little effect on the human interpretation of where the UAS is.

If the ATS is to function without a pilot, a far more precise position estimate must be available to the ATS. As accurate positioning is a requirement for numerous other applications, several types of augmentation which improve upon the raw GPS position are already available. The majority of accuracy improvement is achieved through the use of GBASs, where local ground based receivers with known locations compare their actual location with their GPS indicated position. Any discrepancy is then used to produce an error correction signal which is retransmitted to other users.

The most common form of GBAS augmentation is DGPS. Although it varies by nation, the system is usually comprised of a few stations throughout a country, which transmit the correction signal over many hundreds of miles. For example, the UK has 12 DGPS transmitter stations used to cover the entire country. For most users, including aircraft in flight, DGPS is easily sufficient for accurate navigation. However, certain applications require a far higher level of precision. In these cases, localised GBAS can be employed. Localised GBAS uses the same methodology as DGPS but operates over a much smaller area, ensuring that error correction is locally as accurate as possible. Using localised GBAS should allow GPS accuracy to increase to within 10 cm.

Although large airliners traditionally use systems such as Instrument Landing System (ILS) when landing, more modern aircraft can make use of GPS position as well. As such, localised GBAS is already installed at many major airports, enabling precision approach, landings and take-off. Although localised systems are intended to be the most precise at the runway, previous investigations have found that the accuracy improvement from local GBAS extends far enough to easily cover the largest of aerodromes [69]. Therefore, with this precise navigation system being available in these locations, it would be logical to use the system for more than just landing.

However, whilst small countries such as the UK have total DGPS coverage, larger nations often do not and as such at many aerodromes DGPS is not available. In addition, although GBAS is common at large aerodromes, many smaller facilities are not equipped with the system. As UAS are usually dependant on autolanding, it is likely that any aerodrome from which unmanned aircraft operate will have some form of GBAS. However, as other methods of autolanding do exist, the presence of high accuracy GPS cannot be guaranteed.

In addition to the possibility that high accuracy GPS won't be available, the suitability of relying solely on GPS for position data is also questionable. Even with accurate GPS data, the system is just as dependant on the accuracy of the aerodrome map. Without any form of direct sensing, any modification to the actual aerodrome must also be updated on the map for the system to be aware of the change. Despite aerodrome operating environment being strictly controlled, even small changes can have a large effect. For example, a simplification of a junction during taxiway repainting, would render the the map onboard the UAS outdated. With no method of directly sensing the change, the UAS would attempt to follow the route shown on the map. At minimum, this could be confusing for other airport users and at worst, this could lead to a collision.

Finally, even supposing that high accuracy GBAS were available at an aerodrome with an up to date map, the integrity of GPS data is not certain. With the possibility of GPS loss always being a factor, the minimal operational level of the system should incorporate a method of functioning without GPS data. In addition, malicious manipulation of GPS signals (i.e. 'spoofing') is easily achievable at close range. Whilst airborne, UAS are sufficiently fast and far from the ground to generally be safe from amateur interference. However, when on the ground this could easily be used to misdirect an aircraft which have no other reference against which to check. This scenario is unique to UAS that rely on GPS alone, as aircraft which have methods of validating GPS position (such as a human pilot), will detect this discrepancy and so are naturally less susceptible.

Therefore, for this project is has been decided that relying solely on GPS position data is unsuitable. Instead, GPS should only form the basis of the localisation system with additional sensing used to confirm this data. In order to allow this system to be deployed anywhere, when available GPS precision is assumed to only be as nominal GPS accuracy, of around 15 metres.

3.4.3 Machine Vision with Visual Landmarks

As machine vision has already been identified as the solution being used for collision risk detection, its potential for localisation should also be considered. For pure navigation, machine vision has been demonstrated to be successful in navigation for many years. As early as 1987, [119] demonstrated that a visual navigation system could be used on autonomous ground vehicles. Further extensions have meant that work undertaken in [59] demonstrates map generation by feature recognition in the environment, allowing autonomous vehicles to not only visually navigate an environment, but to also generate a map which then optimises there subsequent routing algorithms.

As such it is clear that the issues identified above could be resolved by augmenting GPS with visually determined localisation estimates. As this will involve more detailed study, previous examples of visual navigation systems will be assessed in a later section.

3.5 Communications

As stated in section 2.1.4, communication with the GTC is essential during civil aerodrome operations. When a manned aircraft is taxiing, control authority is distributed between the pilot and the GTC. Although the pilot remains in direct control, to ensure safety the GTC has the authority to command the pilots action; selecting the routes and timings which should be obeyed for the aircraft to reach its destination safely. This dual approach is most clearly seen in how separation between vehicles is maintained; with the pilot following any preceding aircraft at a safe distance, whilst the GTC ensures other aircraft can only move when safe.

With the potential inclusion of data-links in the future, the GTC could provide much of the required information directly to the UAS. Assuming that taxiing aircraft would reciprocate by declaring their position, it is theoretically possible that safe taxiing could occur without the need for direct sensing on-board each UAS. However, as the use of data links is only proposed, it cannot be considered part of standard aerodrome equipment and therefore cannot be relied upon for this ATS. Although the ATS should have the ability to include additional sources of information, the system is implemented under the assumption that a human will remain in the loop for RT communications.

Although RT information from the GTC will not always be directly available to the ATS, it is still required by the system in order to comply with GTC instructions. Therefore the assumption is made that the human operator will interpret verbal commands for the ATS. Where data-links are not available, the human overseer would be required to highlight the route that the UAS should follow. Although there are proposals to introduce autonomy for aircraft to assign their own routes in crossing the aerodrome, for an aircraft equipped with an aerodrome map, route planning is straight forward and is already considered highly mature. Therefore this has not been explored, and a route across the aerodrome is instead considered given.

As the required data entry would be significant, it is unlikely that even a skilled human overseer could interpret all verbal information for the system. Instead, only the most critical information is likely to be provided. Although the position of other aircraft would be beneficial, information relating to other moving aircraft continuously changes, requiring more effort to update the ATS than would be required to control the aircraft directly. Therefore, if RT is the only form of communication available, the position of all neighbouring aircraft will be assumed to be unknown.

Even if all aerodromes were already fitted with data-links, for the foreseeable future there will always be a certain percentage of aircraft which do not have the required equipment on board (i.e. non-cooperating). Therefore, the ATS cannot be expected to establish the position of other aircraft through communication and instead must detect them directly. In addition to other aircraft, aerodromes often use large numbers of ground support vehicles to assist in operation. Unless all of these were also exchanging position data, a UAS would be unaware of potential collision risks throughout the aerodrome. The decision to use collision risk detection was intended to help mitigate these risks.

In the event of an unexpected encounter (such as a collision risk across an taxiway) the requirement for an intelligent response from the aircraft has not been considered. In the interest of safety, if the aircraft encounters something it cannot comprehend or respond to in a routine manner, the intent is for the system to request manual assistance and for the UAS to stop. This is considered in keeping with current operations, where pilots would notify the tower and wait for instruction if the unexpected occurred.

3.6 Summary

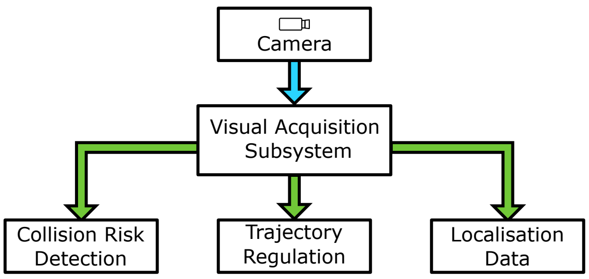


Fig. 3.2 An overview of the Visual Acquisition Subsystem (VAS), showing the data required.

For this research, it will be assumed that the aircraft is operating under its own power and is not making use of a towing vehicle. In addition, the ATS will be attempting to replicate the actions of a human pilot as closely as possible. This will restrict the basic sensing and processing capabilities to being on board the aircraft, as well as indicating that the system must be able to correctly interpret and respond to the conventional arrangement of a civil aerodrome. The ATS must be capable of performing all necessary actions with the minimal amount of support from the GTC. Under normal operating conditions, position data should be available from conventional sources, such as GPS. However if additional data is available it should also be capable of assimilating this data from sensors.

In terms of sensor choice, active sensors (specifically LIDAR) is very suitable for collision risk detection as all objects nearby will be detected without any need for additional interpretation. However, LIDAR was deemed unsuitable primarily due to the sparse environment limiting the effectiveness of LIDAR for navigation, as there will be few 'landmarks' to confirm position. In addition, as LIDAR is not already present on UAS it would have to be added. This is unlikely for several reasons. LIDAR is:

- 'Bulky', and must protrude from the vehicle, interfering with airborne capabilities.
- 'Heavy', adding weight to the aircraft which limits the payload capabilities.
- 'Power-hungry', requiring a suitably capable power delivery system onboard.
- 'Expensive' which could prohibit universal adoption.

Instead, visual sensing using a conventional monocular camera has been selected as most UAS are already fitted with a forward facing camera; making the system more generically applicable and requiring no-airframe alterations (The BAE Jetstream Surrogate UAS test-platform is also already fitted with a forward-facing visual camera). In addition:

- Visual sensing range is only limited by the resolution of the camera, with any large obstacle being clearly visible at range with most modern cameras.
- A visual camera should be capable of interpreting any visual data in the same manner as a human, allowing for both collision risk detection and localisation to be achieved using a single sensor.

- Data is often presented visually to pilots, through the use of lights, floor markings and sign age. For a UAS to make use of this information, a form of visual sensing is required.

As such, the use of machine vision to validate the externally given position will be explored. As GPS data is expected to be relatively accurate, the more critical use of machine vision is to detect potential collision risks. Therefore, for this work collision risk detection is considered the main goal with visual localisation a secondary target, both helping to provide the ATS with data.

Chapter 4

Review of Machine Vision Methods

The following chapter discusses the use of machine vision for automated taxiing, focusing on methods of extracting data from an image. As discussed in Chapter 2, there is very little literature specifically related to sensing techniques for aircraft on the ground. Instead, due to the similarities in the sensing requirements, literature concerning mobile robotics and Self Driving Cars (SDCs) is often more relevant and will be explored below.

This is done in line with the system development process, assessing the requirements and seeking solutions to the encountered problems. As collision risk detection is a major requirement which can only be accomplished using visual data, it is the initial point of research. Therefore, the first half of this chapter is a review of established methods of object detection. As it is established that there are too many types of risk to seek specific objects, object detection methods are determined to be unsuitable, with a more generic approach required for this application. Therefore the second half of this chapter focuses on the chosen approach of using terrain classification to identify surfaces, with the remaining regions of the image assumed to represent obstacles.

After assessing various potential methods of accomplishing the task, an appropriate image-segmentation approach is briefly outlined before the chosen implementation (along with the related contributions) is fully defined in the next chapter. As mentioned previously, this work aims to avoid problems with regulations by avoiding non-deterministic and deep-learning methods throughout the ATS. Therefore, only fully-definable vision processing solutions are reviewed.

4.1 Visual Acquisition System

In Chapter 3 it was established that an ATS would be comprised of multiple sub-systems, each responsible for a specific element of the overall task. Based on the system architecture described in the previous chapter, a Visual Acquisition Subsystem (VAS)is required to interpret the image captured by the forward facing camera. As conventional avionics are not designed for surface operations, the VAS provides the main source of information used for automated taxiing. Without alternative sensors for comparison, the success of the ATS is dependant on the accuracy of the data extracted from each image.

Despite this requirement, visual data extraction is significantly more complex when compared to other types of sensor. When using a three dimensional sensor (such as LIDAR), the depth data may require interpretation to determine what something is, but the presence of an obstacle is known from the basic sensor output. By comparison, although an image can contain a great amount of information, the system which captures the image is only aware of coloured pixels. As digital cameras were originally intended only to produce images for human interpretation, any useful information must be extracted in an additional step. Just as human perception is based on experience and understanding, the software inspecting the image is the key component in the abilities of visual sensors.

4.1.1 Task Priority

Shown in Figure 3.2, the VAS is responsible for extracting three types of information from the camera image. In each case, visual information can be used to:

- Determine the position within the aerodrome (Aerodrome localisation data).
- Ensure correct orientation on the current taxiway (Taxiway position regulation).
- Detect objects which pose a potential collision risk (Collision risk detection).

The inclusion of visual data is intended to improve localisation accuracy, but is not an essential requirement. As shown by many existing military UAS, taxiing is possible using GPS alone (as discussed in Chapter 2). When compared to other ground based systems which require collision avoidance, such as robots or SDC, collision avoidance for aircraft is simplified by the aerodrome environment. To make landing and taxiing easier, aerodromes are usually built on flat land, greatly reducing the chance of collision with the terrain. In addition, there are rarely any structures built along taxiways or runways, with any permanent static collision risks typically already marked on aerodrome maps. Therefore known collision risks can typically be avoided using accurate localisation alone.

Assuming that the UAS has highly accurate localisation and will follow a suitable course, the remaining potential collision risks will be unknown objects positioned on the taxiway (or runway) which have the potential to impact with the UAS. As collision risk detection is essential and can only be accomplished through machine vision, it is the most critical requirement and therefore the initial point of research.

4.1.2 Methods of Detecting Collision Risks

When attempting to detect objects, the choice of sensor dictates the entire approach. Active sensors (such as LIDAR) provide range and bearing estimates to any nearby surface, allowing any object which extends from the ground by even a few centimetres to be detected. Processing the captured pointcloud (such as segmenting by velocity and location) allows both static and mobile objects to be isolated, providing generic collision risk detection without the need to specifically identify any objects.

Although this information would be highly useful for automated taxiing, this work is being undertaken with only a single forward facing camera. Without any form of depth data, any captured image requires significant interpretation in order to yield useful results. If working solely within the image plane, it is extremely difficult to detect the presence of obstacles without first identifying what the obstacle is. If compared to active sensors, the process is essentially reversed, beginning by detecting objects within the image plane, then estimating depth only after an object is identified. Therefore, identification of collision risks (and non-collision risks) is a key part of a working system. The chosen approach must be capable of:

- Identifying objects within the image plane which could potentially collide with the UAS.
- Estimating the position of each object relative to the UAS.

There are many methods to detect objects within an image, yet all methods can be broadly divided into two categories; known object detection, where the object being sought is of an expected type, or generic object detection where any object can be detected. Both approaches have advantages and disadvantages and so both were considered, beginning with known object detection.

4.2 Known Object Detection

'Known Object Detection' methods attempt to detect the presence of specific objects within an image, comparing prior knowledge of the expected objects to what is visible within the captured image. For collision risk detection, this is only viable for objects whose presence can be predicted and where the number of potential risk types is fairly low. As aircraft ground operations are potentially very dangerous, aerodromes are highly restricted environments, effectively eliminating many types of risk that are found in other settings. Aside from static obstacles, the most likely collision risks are other aircraft, ground support vehicles and pedestrians. Assuming that these three types of object will make up most of the risks within an aerodrome, known object detection could be a viable solution.

Although limiting the detection capabilities to only certain objects is disadvantageous, there are numerous benefits of using known object detection. As the sought objects are well defined, comparison to known examples makes detection more robust and typically resilient to false detection. Furthermore, the additional information about the known objects can be used without the need for further data extraction. Although identifying the presence of an object suggests that there is a potential risk, the object is not actually a known risk until it has been determined to intersect with the intended path of the UAS, requiring both it's position and motion relative to the UAS to be established. Despite generic 'depth extraction' methods being viable for any object, if the actual size of an object is known it can simply be compared to its scale within the image to quickly estimate the distance. Furthermore, the orientation of a vehicle can be used to estimate its heading, even when the vehicle is not in motion. With this additional information being highly useful, methods of known object detection were investigated for use in aerodrome conditions.

4.2.1 Orthogonal Templates and Feature Descriptors

'Known Object Detection' methods function by comparing known information about an object to what is present within an image. Owing to the heavy use of machine vision in manufacturing, many methods were original designed for inspecting objects on a flat surface, with the camera arranged orthogonally. Orthogonal methods work on the assumption that sought objects will be consistently orientated towards the camera in the same way (i.e. the same face will always be visible). As orthogonal object detection methods are still widely used, they were chosen as the starting point of this investigation.

Despite orthogonality being difficult to enforce between vehicles, orthogonal approaches have been used by SDCs systems for identifying other vehicles on the road. As cars tend to remain in lane, the vehicle ahead of the camera will present a fairly consistent view which can be identified through orthogonal template matching. Therefore the potential exists to use a similar approach to determine the presence of a leading aircraft ahead of the UAS on the taxiway.

The most basic object detection methods use raw pixel data directly, with the pixels in example images compared to pixels within the captured image. Although correlation based matching allows for the object to vary in scale and rotation within the image and even global changes in illumination [107], raw pixel methods can only detect objects if they are extremely similar to the template. Direct pixel matching approaches are strongly affected by problems such as occlusion, clutter and local illumination changes. As UAS operate outdoors and in all weather conditions, pixel matching methods would be difficult to use in an aerodrome environment.

Rather than relying on raw pixel values, most approaches use 'feature descriptors', which establish trends in images which are easier to compare. This not only improves the result, but is also usually accompanied by a decrease in processing time. Furthermore, although individual features can be used for detection, the most common orthographic method is shape based matching, in which features are extracted for the template as a whole. The extracted features are stored relative to each other, such that an affine transform can be used to vary the rotation and scale of the template for comparison with the captured image. Figure 4.1 shows various feature descriptors which can be used in the orthogonal detection of a leading aircraft. As feature descriptors must be extracted from an existing image, the grey-scale image in Figure 4.1a was created using a 3D model to be a basis for example templates.

As the intention is to use object detection in outdoor conditions, illumination invariant features are key. Some of the most popular illumination invariant features are 'key-point' descriptors such as Speeded-Up Robust Features (SURF) [14], shown in Figure 4.1b. Although the methods of establishing key-points vary (often based on local maxima), the underlying principle is to only extract a limited number of key points from a template, using a method which can be reliably repeated. This allows a quick comparison between key points in another image to quickly determine if the sought object is present.

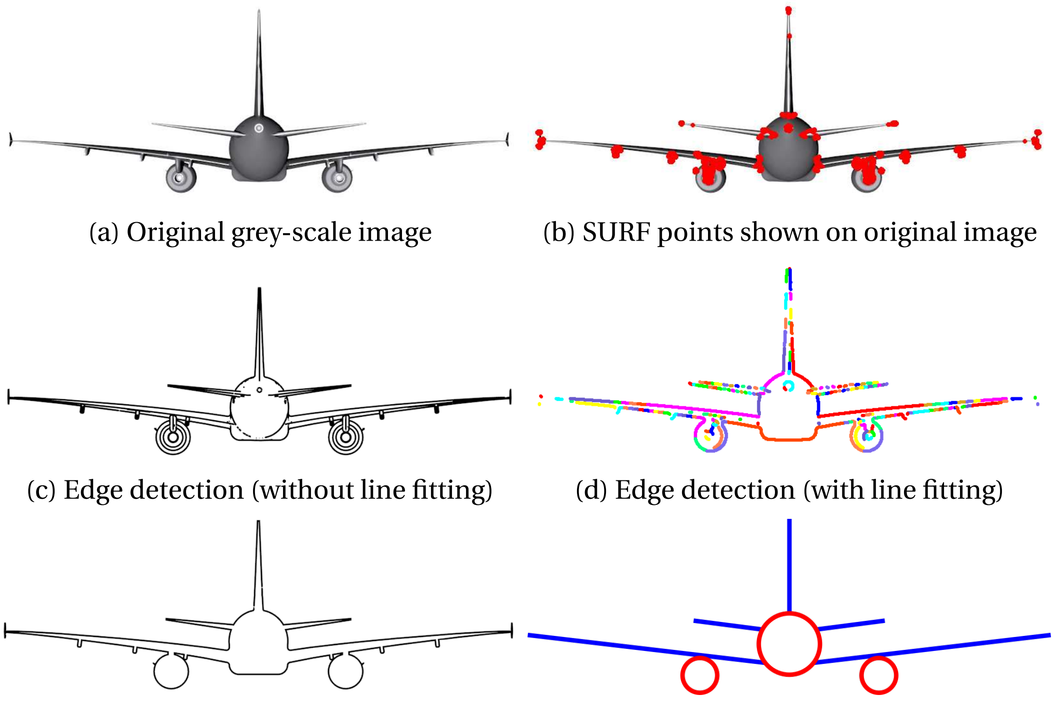

(a) Original grey-scale image
(b) SURF points shown on original image
(c) Edge detection (without line fitting)
(d) Edge detection (with line fitting)
(e) Shape outline for Fourier transform detection
(f) Simple component detection strategy

Fig. 4.1 Comparison of several feature descriptors applied to a template image, designed for use with orthogonal detection techniques.

Alternatively, other simplistic illumination invariant features are ‘edges’, defined as elements of the image where neighbouring pixels are highly dissimilar. Figure 4.1c shows a template created using pure edge detection. As changes in lighting are unlikely to make two highly different regions within an image look very similar, edges should persist under nearly all changes in illumination. This makes them a highly robust feature to use, and are easy to understand as they are also a critical part of human vision.

However, as using raw edge data is very similar to raw pixel data, the matching process is computationally expensive and a robust match is not always achieved. Instead, it is more common for the edges to be further processed. Figure 4.1d shows the raw edges grouped into straight lines and curves which are easy to mathematically define. Not only is it quicker to compare two straight lines based on location and orientation (rather than the individual pixels within an edge) but these techniques also allow some degree of variation tolerance during matching. Alternatively, if the outline of the shape is well preserved against the background, the entire outline may be used, as shown in Figure 4.1e. Many airborne detection systems identify flying objects using just the perimeter of a shape, where the Fast Fourier Transform (FFT) of the perimeter can be calculated to quickly compare against known examples to seek a match.

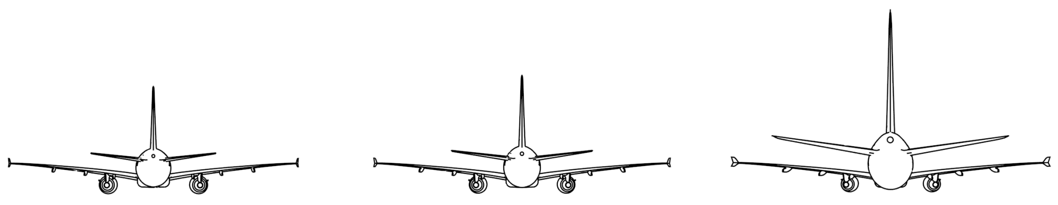
Regardless of the method used, orthogonal shape based matching is widely established. However, the main disadvantage of shape based methods is the inability to deal with any variance between the template and the captured image. As detection relies on maintaining consistent spatial relationship between every feature detected within the template, small changes can often produce a false negative result, even when the majority of the object is correctly identified.

A potential solution would be to move away from shape based matching and instead use component based matching. As with shape based matching, shapes are sought within the template. However these are typically far simpler primitive shapes which are detected independently, as shown in Figure 4.1f. The relationship between detected shapes is then used as the template for the overall object. Component based matching is commonly used for detecting objects with variable configuration but the same underlying components, or objects of a class which are similar but not precisely the same. As detection is based on individual parts of aircraft, such as engine nacelles and tail fins, it is possible to use a component based approach to detect a generic aircraft shape. However, if the exact dimensions of the aircraft are no longer known, the scale of the object within the image cannot be used to provide depth information (as aircraft of varying sizes use similar configurations).

The main benefit of orthogonal methods is the speed of detection. As the object is only sought in a single orientation (with only scale and rotation transformation applied to the template) an entire image can be searched very quickly. However, limiting detection to only objects known to be aligned with the camera (or objects which appear the same when approached from any direction) severely limits the ability to use the approach. As many differently sized aircraft share a similar profile, detecting an aircraft using this methods cannot reliably provide an estimate of distance. As such, not only is an alternative method of detection required for other objects, when orthogonal detection does detect an aircraft an additional method of depth extraction would also be required. As such, despite the speed and prior use of orthogonal matching in related applications, it is not considered a viable method for collision risk detection in aerodromes and has not been further pursued in this work.

4.2.2 Perspective Methods

A major limitation of orthogonal methods is the inability to deal with the effects of perspective. Even when an object is orthogonal to the camera, unless the object is entirely planar (i.e. flat) the effects of perspective can be seen as the camera moves closer, as shown in Fig 4.2. Due to the size of aircraft wings, the issues with perspective are far more pronounced for an aircraft than for a car, requiring significantly different templates as the range changes.



(a) 70m separation (b) 40m separation (c) 12m separation

Fig. 4.2 Comparison of the effects of perspective over distance. A 3D model of an Airbus A320 is shown at varying distance from the camera. Each image has be scaled such that the wingspan appears equivalent, to allow the changing shape to be observed more easily.

This simplest method to introduce perspective matching is to use the same techniques as in orthogonal matching but to extend the transformation beyond the affine. By linearly scaling the co-ordinates of a template in two dimensions simultaneously but unequally, straight lines remain straight but parallel lines stop being parallel, mimicking the effects of perspective. This is known as a ’Perspective transform’.

However, the obvious limitation with this method is the lack of depth data within the original template. Much like raising a piece of paper from the desk and rotating it in free space, the template is only representative of a 3D transformation of a 2D object, as shown in Figure 4.3. Although perspective deformable matching is widely used for 3D applications where the object has a flat face, more complex geometric shapes, such as aircraft, cannot be easily detected using the same method. Perspective deformation searching is suitable for finding the 3D poses of 2D objects within scenes, but is not suitable for locating full 3D objects. Instead, in order to ensure that the system can recognise an object from any angle and distance, data pertaining to every potential viewing position must be available. As the object is no longer considered a two-dimensional representation, this is commonly known as 3D matching.

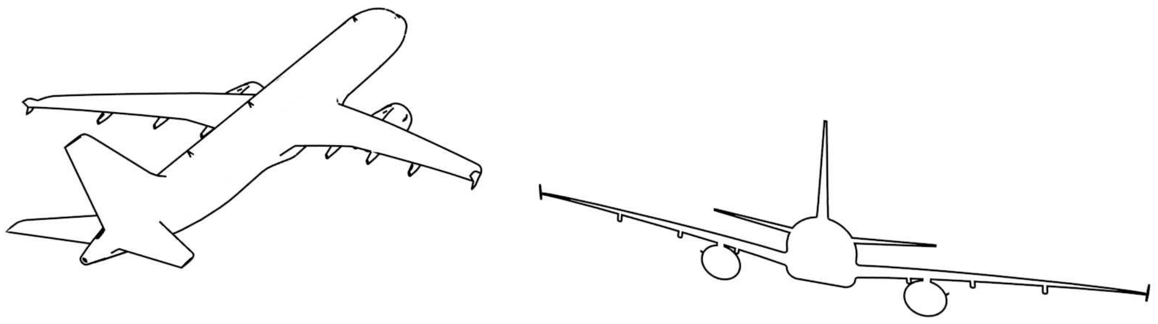


Fig. 4.3 Complex objects under perspective cannot be detected using orthogonal techniques. Even when the template undergoes the correct perspective transformation, the lack of depth within the template prevents enough alignment for a match to be found.

Despite the use of the term '3D' implying that a different technique will be used, the detection problem is essentially the same, with a known object to be found within a 2D image. As such, the same feature extraction methods as used for orthogonal detection are still valid. Instead, rather than a single template, additional templates are required for each potential viewpoint (orthogonal shape based matching is essentially an over constrained application of 3D shape based matching). Although a physical object could be photographed from many angles to generate the templates, a virtual camera orbiting a 3D computer generated model is often more practical. As the entire shape of the model is used, this is again commonly referred to as 'Shape based matching'. Using software designed to portray models realistically, the characteristics of the real-world camera are mimicked to introduce the effects of perspective. The distance and orientation of the object with respect to the camera are then altered in small increments, to generate many thousands of viewpoints.

To test this method of object detection applied to an aerodrome environment, the commercial MVTec Halcon vision processing software was used to generate the templates required to detect an A320 aircraft. Returning to the orthogonal methods above, the limitations on orientation were still appropriate, in that other aircraft are never likely to be seen from above or below during taxiing. Using this prior information simplified the search process; as the aircraft should lie on the ground plane and the height of the camera is known, the model only has to rotate in a single dimension, reducing the number of required templates. An example of a match is shown in Figure 4.4, demonstrating that this method is highly successful in detecting known objects. In addition, the relative position and orientation of the aircraft were both extracted, allowing them to be used in the wider ATS.

However, although the knowledge that the aircraft is on the ground does reduce the total number of templates required, the aircraft's heading and distance to the UAS are entirely unknown; still requiring many thousands of sample templates to represent the aircraft in varying poses relative to the camera. As each potential orientation and distance of object is a separate template, the matching process is essentially repeated many thousands of time. This makes the approach far more computationally intensive, taking several seconds to get a match when seeking the highest level of precision on a standard desktop computer. To speed up the search process, the features are extracted from each template and stored to eliminate the need to constantly sample the same templates. However this can require significant storage space, with the templates of the A320 generated to perform the matching in Fig 4.4 taking 140GB of storage to account for all potential orientations and distances. Furthermore, as the effects of perspective and scale are specific to the camera and lens, the produced templates are only compatible with the same hardware configuration.

When it is further considered that each additional object will also require the same number of as many templates, the total computational effort to detect all expected collision risks would multiple with the number of objects, likely making 3D shape based matching unsuitable for use in this task.

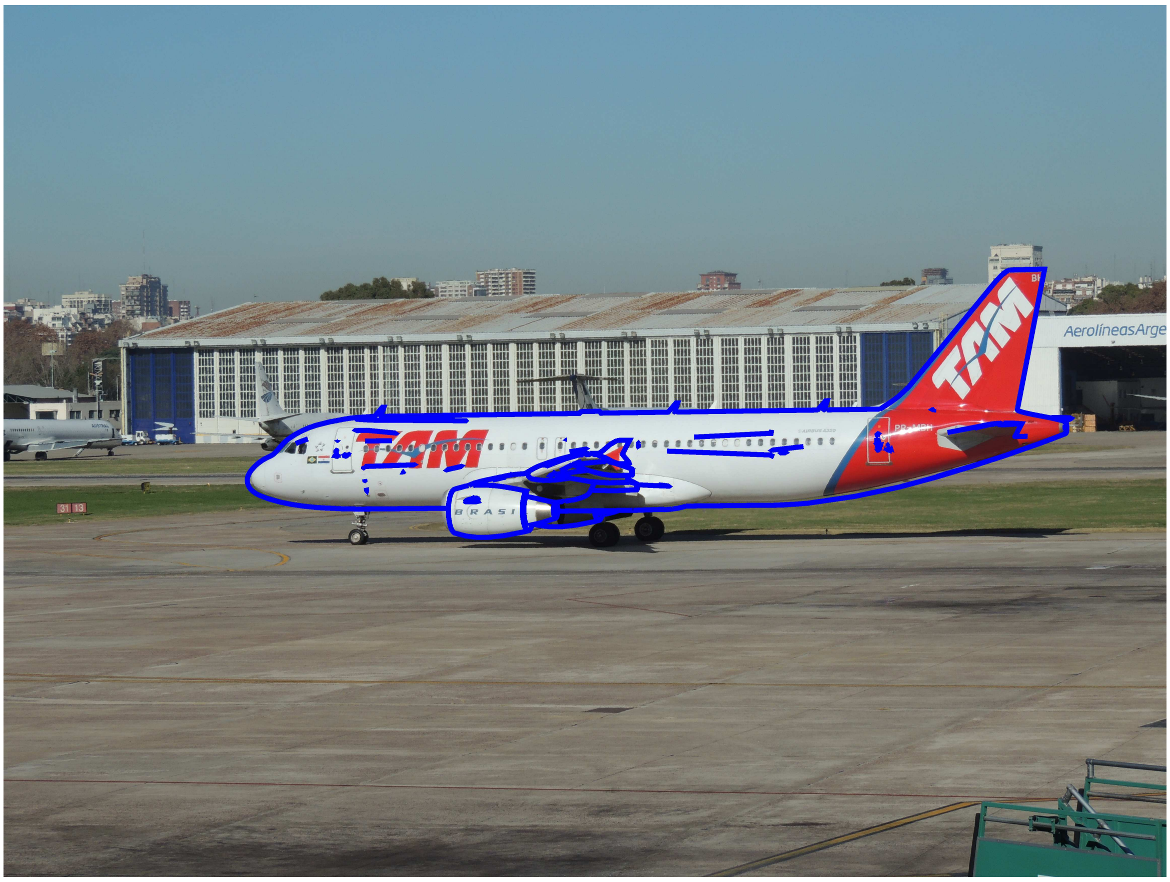


Fig. 4.4 Example of 3D shape based matching, using previous A320 model to detect real-world example. From the known camera parameters, the pose and orientation of the aircraft can be determined with respect to the camera. Object detection achieved here using MVTec Halcon.

4.2.3 Model Based Object Detection

As seeking out many specific objects is likely to be computationally intensive beyond the scope of what is achievable aboard an UAS, the alternative approach is to establish a ‘model’ for each class of objects, and use that to detect potential collision risks. Making use of the shape and contextual relationship between parts of objects, non-specific variants of classes can be identified. For example, four wheels in a rectangular configuration may be enough to identify a road vehicle (although the system would not be capable of distinguishing if the vehicle were a car or a van).

Similar to the orthogonal component matching approach, this can be achieved by breaking a 3D object down into simpler 3D ‘primitive shapes’ and then identifying the larger object based on the relative position of primitives within the captured image. However, although a component based approach is commonly used for orthogonal matching, performing this action with parts in any orientation or distance is far more complex. Determining the spatial relationship between elements requires a high-level understanding of the parts being extracted, in order to match other elements of the image. For example, most aircraft have two wings, but only one is visible when the aircraft is seen from the side. Any viable 3D component based approach would require interpretative logic capable of understanding that only one wing is visible if the fuselage is obscuring the other.

For a component based detection method to be viable, members of the class must not only look alike, but share similar physical characteristics. For example, using Fourier descriptions of outlines rather than individual components, work done in [21] shows that both road vehicles and aircraft can be detected using generic models of a class, rather than an exact 3D model, allowing generic detection which could be used for collision avoidance. For SDC this can be taken further; as the majority of collision risks will be other road vehicles on the road ahead and most road vehicles are similar in width, the scale of the vehicle in an image can be used to estimate the range between the camera and a proceeding vehicle, without any exact classification of the vehicle which has been detected.

Despite this prior work, this same process cannot be used for aircraft. As with road vehicles, many aircraft have the same configuration, with airliners being especially similar across different models and even different manufacturers. However, aircraft can differ significantly in terms of scale. Seeking wings, tail, fuselage and undercarriage both an Airbus A320 and an Airbus A380 could be detected using the same generic parts. However, the A380 is over twice as long as the A320. Without any approximate dimension within the aircraft that can be used to infer scale, the distance from the camera remains unknown. As the end result is recognition without depth, this approach cannot be considered viable and has not been pursued in this work.

Beyond methods which rely on physical models, deep learning methods are also capable of identifying generic examples of objects within images without the system having an exact example to match against. Work such as that undertaken in [31] explore the use of deep neural networks to recognise a wide range of object types, at any scale and within any form of image. As before the biggest limitation with this approach is that the specific type of object is not identified. As such the scale and therefore the position of the object is still unknown. Whilst this type of method has great potential in identifying aircraft and other entities around the aerodrome, it is not useful for determining a objects position. Therefore, whilst a similar approach may be applied to determine what an object is, it cannot be used to determine where it is. Finally, as deep learning methods contain elements which are hidden from the human operators, they are not suitable for use within aerospace due to issues surrounding certification.

4.2.4 Conclusions on Known Object Detection

From the section above, it is clear that ‘known object detection’ is an established area, with much research progress having been made in related fields. A large benefit of seeking only known objects is the ability to integrate prior knowledge into the system. Specifically for collision avoidance, if the actual dimensions of an object are known then the size of the object within the image can be used to estimate range. Additional information, such as the type of object, can also be used to increase situational awareness. For example, if a stationary vehicle is detected, the heading and therefore most likely trajectory can be determined, allowing collision risk detection to be included should the vehicle begin to move. Similar concepts have already been demonstrated on SDC, with road vehicle detection allowing for safe navigation, even in cluttered urban environments.

However, searching for specific types of objects has several significant disadvantages. Compared to the instant range and bearing given by active sensors such as LIDAR, one of the most noticeable issues is the long processing time. Searching for a particular object requires that every region of the image is checked not just once, but for every possible orientation and range that the object is likely to be. As collision risks can occur both at distance and nearby, this can significantly increase the computational time required. Furthermore, if multiple types of object are being sought, the processing requirement is essentially multiplied by the number of potential objects (after feature extraction). This is problematic due to many types of object the UAS could encounter. In addition to the hundreds of potential types of aircraft, there are countless support vehicles and other objects that would also need to be maintained in the database, potentially requiring many thousands of object types in total.

As object techniques have matured, model based detection methods now exist which can identify objects by class, rather than finding an exact match. Downsides of this approach include the loss of any prior knowledge about specific objects, in addition to potential decrease in robustness. In addition, unless the dimensions of the object are known, the range can only be estimated using an additional stage.

Finally, the most obvious disadvantage of only seeking known objects is that the system can only identify pre-selected objects. If an object is present in the image but was not anticipated, the system would not be able to recognise it. Despite the well-controlled environment, there is always the possibility of an unexpected object posing a collision risk. Although a complete library of aviation related equipment could be collated, any new types of object that are not part of the detection system would not be detected. Collision risk detection methods which can only detect objects of expected types are not capable of fulfilling the requirements of this project. Therefore, in order to detect any risk, the obstacle detection methodology must be as generic as possible.

As a generic risk detection method should also detect expected objects, there is no specific requirement to include a known object detection stage and work on a single generic risk detection method will be the focus of this work going forward. However, as identifying other aircraft and ground vehicles could provide additional information useful to an ATS, the ATS should be implemented such that it can be expanded out to include specific object detection at a later date.

4.3 Generic Object Detection Using Depth Estimation

‘Generic Object Detection’ methods are defined as being capable of detecting objects within an image, without prior knowledge of the object’s appearance. Unlike known object detection, where all possible techniques are fairly similar in their approach, the methods used to detect generic object differ significantly. Although some techniques may seek to detect distinct objects, other methods inspect individual pixels or small regions instead. As generic object detection techniques are highly varied, it is also possible to use separate techniques in combination to improve the final result.

The added requirement of seeking to avoid collision with any detected object introduces the need for depth extraction. As the identity of any detected object is unknown, features of that object cannot be used to estimate its position. Instead, a different method of estimating the distance to an object is required to determine if it poses any risk. Although depth data is not present within a single digital image, methods of recovering depth from 2D images are well documented. If the captured image can be converted to a depth map, objects can be found using similar techniques to point cloud based processing. Therefore, object detection and depth extraction are both required, but either can occur first.

As extracted depth directly removes the requirement to actual identify most of the object in each image, depth recovery methods were the initial focus of this work and are documented in this section. Although direct depth extraction is likely the best form of generic collision avoidance, the methods available to use with a single camera were found to be either too inaccurate or too dangerous to employ in an ATS. As such, direct depth estimation methods have not been used in this work and are only presented here for completeness.

4.3.1 Stereoscopic Depth Extraction Through Motion

Despite the lack of any range data within a two-dimensional image, some three-dimensional information is available in the form of the angle between each pixel and the optical axis of the camera. From a single observation, this bearing alone cannot be used to estimate depth. However, if the same point can be identified in multiple observations, the relative position of the camera can be used to calculate range. As such, the most common machine vision techniques which attempt to extract depth data use more than one image, commonly referred to as ‘stereo-vision’.

As the only difference between a monocular camera and a stereoscopic camera is an additional viewpoint, images taken by a moving camera can be used in a similar way, with multiple frames used to estimate depth. As this work intends to make use of a video feed, rather than a single image, it is possible to use techniques which seek to extract depth data from images by tracking the change in objects over time. Work undertaken as early as 1994 [129] demonstrated accurate depth estimation, provided the egomotion of the camera could be accurately tracked. For static objects viewed from a side-facing camera, the motion of the camera can be used in combination with the movement of pixels in an image to estimate their respective distance based on parallax. Pixels with similar distances can then be grouped into objects as appropriate.

However, for this work the only camera available is a single forward facing camera. As the motion of the vehicle is aligned with the optical axis, there will be minimal change in the location of pixels between frames. As conventional pixel tracking techniques (such as ‘optical flow’) rely on clear features and relatively large motion for accuracy, most objects would not be detected until the UAS were in close proximity. As such, Stereoscopic methods which rely on a single camera can be considered unsuitable for this task, as depth cannot be extracted at the required range.

4.3.2 Motion Predicted Imagery

The UGOMS system [32] (as aforementioned in Section 2.4.5) uses a similar camera arrangement, with a single forward facing camera intended to provide all relevant information for automated taxiing. The developers of UGOMS recognised the difficulties in known object detection and elected to use a generic object detection method for collision avoidance. This was achieved using a technique known as Motion Predicted Imagery (MPI).

MPI uses multiple image frames to detect risks without classification. However, rather than attempting to determine depth from the outset, a prior stage is introduced in which likely objects are first detected based on their change in appearance over time. Assuming that everything ahead of the vehicle should be flat ground, the pixels from previous frames are shifted to reflect their expected position in the current frame. This uses both the egomotion of the vehicle and the relative position of the camera with respect to the ground. As the aircraft approaches a three dimensional object, the effects of perspective will cause the object region to differ from the estimated image. Groups of pixels which show a large discrepancy are then grouped and assumed to indicate an obstacle. Once isolated, the movement of this region between images is used to estimate depth. As no prior knowledge is required, MPI can be considered a truly generic object detection technique. In addition, MPI is especially suited to aerodrome environments as its main requirement is flat terrain.

However, despite offering a truly generic solution, MPI has several issues which make it less suited for collision risk detection. As MPI warps the previous image assuming all pixels represent the ground, the disparity between detected and estimated position increases with the height of objects. As such, low objects, are extremely difficult to detect, despite small objects (such as runway lights) still posing a large risk to the UAS.

In addition, MPI is highly dependant on accurate ego-motion tracking of the camera. As aircraft undercarriages are often very flexible, vertical camera movement is common. Even slightly uneven terrain can produce unexpected camera movement, resulting in inaccurate results. The errors introduced by ego-motion and camera stability are exacerbated when tracking is performed on individual pixels. As each pixel is compared to the prediction independently using MPI, small discrepancies can grow into large errors. Simply shifting the image by a few pixels (such as would occur during a bounce) will result in the entire image being indicated as a collision risk. Although multiple frames are used to establish the presence of an object, if an error occurs over several frames a false collision risk could be detected. As false positives are highly undesirable, a method which can overcome these issues would provide a better method of risk detection.

4.3.3 Deformable Net Object Detection

A technique known as Region-based Deformable Net (RbDN) Analysis uses region based tracking to detect potential collision risks. Initially proposed in [97] and extended for vehicular use in [98], RbDN isolates regions of an image and tracks them over time. As the name suggests, RbDN begins by segmenting the image using a 'net', i.e. breaking an image into a series of small regions for analysis through region based image segmentation.

A difference between simple region based segmentation and the RbDN approach is that net is formed of polygons, rather than clusters of pixels. The net consists of individual vertices (or nodes) which are free to move. These nodes are then connected to form edges, which in turn form polygons. As the edges are restricted from crossing each other, each polygon represents a unique part of the image. The net used in RbDN is able to deform and adapt to the image below, with the end result being a series of polygons which reflect a simplified version of the original image. Although the net is adjusted to fit the underlying image, it remains defined by the position of the vertices and the edges which run between them. This allows a consistent net, which slowly updates to reflect the changes in the images over time. By tracking the change in polygon size relative to the motion of the camera, the range to each polygon can be calculated with some precision.

The work done in [98] applied the concepts developed in [97], by using RbDN on a ground based robot. In [98], a robot is used in an environment where externally provided position data (such as GNSS) is unavailable. Without GNSS the robot's position data is limited to mechanical odometer and compass heading readings. Over time, the accumulation of errors through integration leads to the robot losing orientation and becoming lost. As a method of countering this, RbDN is used to track the position of nearby objects within an image captured by an onboard camera. By tracking the objects over time, [98] shows it is possible to accurately determine their distance to the camera. By tracking these distances as the robot moves, the robot can effectively eliminate the issues of growing position errors by correcting the odometer data. Later work in [98] shows that it is also possible to estimate the distances to objects that are more than 100 m away, as long as the camera displacement is over 10 % of the expected distance to the objects.

However, although RbDN is capable of detecting generic objects, it also has disadvantages. One issue is that this method is only suitable for detecting stationary objects. If an object is moving at the same speed as the camera, its area will not change over time and it will either not be recognised or simply calculated to be very far away. In addition, certain forms of motion can result in false collision risk detections. For example, in outdoor conditions it is common for terrain to be partially obscured by buildings. If the camera were to move passed a building in front of a grassy area, as the building passes out of the image frame, the grass region will grow as it becomes more visible. Despite being static terrain, the increase in area would suggest a rapidly approaching object and trigger a false collision risk detection. As these issues are fundamental to the methods used in RbDN, it has not been selected as a suitable method for this work.

4.3.4 Conclusions on Object Detection Using Depth Extraction

If the distance between each pixel and the camera can be determined, objects can be identified by grouping points located close by in both the image and in free space, allowing collision risk detection without having to first identify the image contents. Drawing from previous work in related fields, three potential methods of achieving this have been presented. However, as stated above, each of the presented techniques has issues which make them unsuitable for this application.

Furthermore, in addition to the previously stated issues specific to each technique, the biggest limitation with depth based detection methods is not the methods themselves, but the hardware limitations specific to this work. As only a single monocular camera is assumed to be present, additional viewpoints will only be available if the camera moves.

Therefore, for any of the three techniques listed above, the UAS would need to be in motion in order to detect if there is an object in its path. As each process can require several frames before any potential risk is detected, there exists the risk of immediate collision, as soon as the UAS moves. As requiring that the UAS is taxiing in order to detect if there is an object in its path is potentially dangerous, even with human operator oversight, it would be better to implement a method which functions even when the aircraft is stationary. Primarily for this reason, in addition to the other reasons stated in the individual sections above, motion based depth estimation is not considered a suitable method of generic object detection for aerodrome conditions.

4.4 Generic Object Detection Using Terrain Classification

Returning to the concept of known object detection, the conclusion was drawn that it would be impossible to create a suitable database of all the potential objects (i.e. all objects) which could represent a collision risk. Furthermore, as an obstacle can be any size, shape or colour, no suitable model could encompass such variety for detection. For these reasons, known object detection was determined to be impractical for this application.

One method of overcoming this issue is to focus on identifying only the most relevant objects within the scene. For extremely simple environments, the process of detecting unknown objects can be considered ’anomaly detection’ in which regions of the image which are sufficiently different from the rest are viewed as objects. However, as the complexity of the image increases (with variable lighting, weather, occlusion, clutter and border interception) attempting to find outlying regions becoming increasingly difficult.

Although collision risks are highly variable, the aerodromes themselves are not. Despite some differences between individual aerodromes, most are typically uniform in appearance. As such, the most easily classifiable regions in a captured image will not be obstacles, but the terrain features of the surrounding environment; such as concrete, asphalt and grass. Therefore, this work proposes using the principles of anomaly detection to detect collision risks. By determining the parts of an image which do not represent collision risks, the presence of risks can be inferred from the remaining areas.

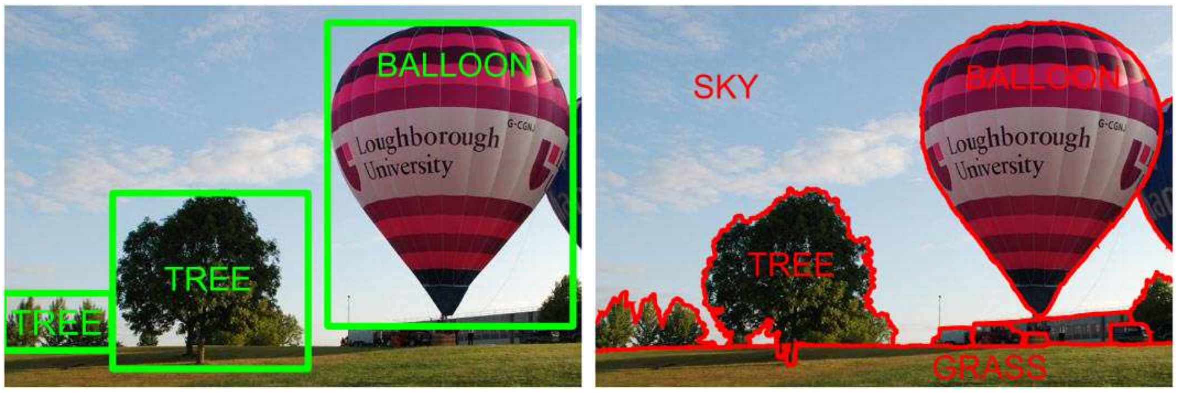


Fig. 4.5 Comparison of Object detection and Semantic Segmentation results

Image Segmentation

Whole image recognition tasks which seek to both divide up an image and classes the parts, are commonly known as 'semantic segmentation'. As with object detection, classification can be performed on any known region. Therefore, in addition to typical terrain surfaces, common objects that could pose a collision risk can be classified through comparison to known data, similar to object detection. The regions with a low confidence of matching any known class can be generically labelled as an 'unknown' collision risk. As generic risk detection is essential, this approach has been chosen.

Often considered an alternative to object detection, semantic segmentation differs in that every single pixel within an image must undergo classification. As such, the computational cost of semantic segmentation is high. However, accurate semantic segmentation isolates regions within the image to a pixel level, not only determining the class of objects but also performing accurate border extraction. This provides more precision which should be of benefit when estimating the position of potential collision obstacles.

For this work, semantic segmentation will be explored with the specific goal of accurate 'unknown' detection. As semantic segmentation is not novel, existing techniques can be utilised. However, many techniques cannot reject areas as unknowns. Of the methods which can include an unknown class, the focus remains on known classes, with the end results featuring minimal unknowns or inaccurate borders. As such this work will take a more novel approach, intending to introduce highly dependable 'unknown' recognition for collision risk detection. Furthermore, terrain classification can also be used for localisation. Rather than rely on discrete landmarks, an alternative is to use a more holistic approach, in which the entire ground area is compared to known map data.

Data Extraction, Horizon Detection and Depth Estimation

In order to classify regions of the image, data extraction will be required. Typically, data for classifying a region is taken from that region itself, such as colour or texture. Additional information, such as relative position, can also be used. Due to the decision to use terrain features to determine risks, determining the difference between sky and ground based elements of the image is essential. As such, the position of the horizon should be known, and horizon detection methods should be explored.

Horizon position is also useful for 'depth estimation' when working with unknown objects and a single camera image. By determining the position of the horizon within the image, and knowing the characteristics (including height) of the camera, the relative bearing of objects within the frame can then be established. By limiting the region of interest to below the horizon, unknown regions must always meet with the known terrain Assuming it can be seen, this is always the lowest point of the unknown region within the image. As the height of the camera is known, an approximate distance based on trigonometry can be extracted.

If the actual object does not make contact with the ground, this approach may indicate the range between the camera and the object to be greater than it actual is. However, as the position of the bottom object should give a better indication of range, the risk can be avoided by using the lowest pixel of connected objects to provide the shortest range estimate.

4.4.1 Conclusions on Object Detection Through Terrain Classification

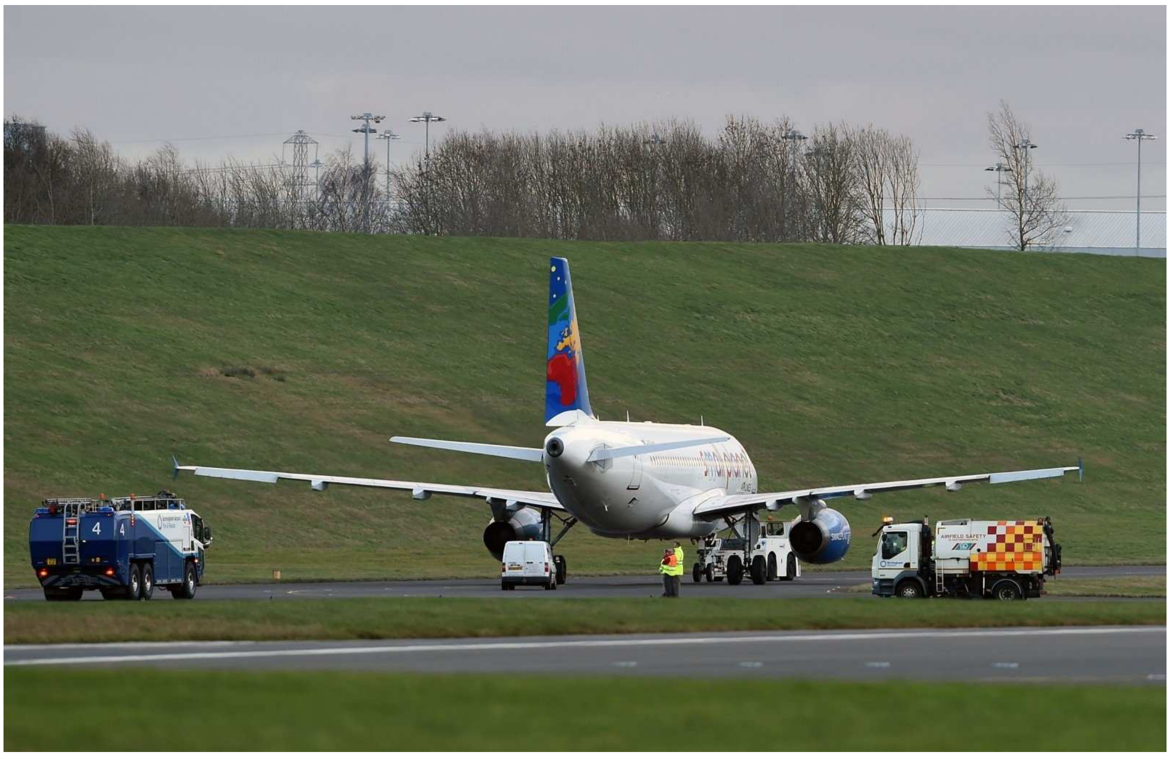


Fig. 4.6 Earthen barriers at Birmingham Airport (Reuters UK, 2016)

As all unknown regions are to be treated as collision risks, the largest potential issue is misclassifying an unknown object as a terrain feature. If this occurs, objects which look similar enough to terrain features would not be detected as collision risks. Although buildings are not typically made of asphalt, both buildings and taxiway surfaces are commonly made out of concrete. In addition, airports which are located close to urban populations often endeavour to minimise disruption to their neighbours by erecting structures to reduce noise and light pollution. Although some structures are made of artificial materials, many airports simply use large earthen barriers to absorb or reflect the sound (as shown in Figure 4.6). As these 'structures' share identical features as the ground (such as colour and texture), terrain classification techniques will likely not detect them as collision risks.

However, for this work collision risk detection is focused on the detection of non-static risks. Static collision risks, such as buildings or sound barriers, should be denoted on the aerodrome map. Provided that localisation is accurate, the UAS should be able to avoid these objects despite the possibility that they are not detected as risks.

4.5 Conclusions

The visual data available to the UAS has two main purposes: collision risk detection and localisation improvement. As GNSS systems, such as GPS are already standard on UAS, collision risk detection can be considered the more important requirement and therefore is given the higher priority. However, monocular cameras are difficult sensors to work with in uncontrolled environments. Despite being immensely data rich, they have little usable information in their native form. As no depth data is available without processing, depth extraction is a necessary part of collision risk detection.

Within an aerodrome environment, the size of moving obstacles can vary dramatically; ranging from small Foreign-Object-Debris (FOD) to large airliners. At the same time, these obstacles might be alone on an empty runway, or surrounded by other objects on a busy apron. Assuming that the UAS is limited to the same data currently available to human pilots, the size, type and location of all mobile risks start as unknowns.

Object detection techniques which attempt to find known objects are usually very processor intensive, needing to inspect the entire image for the object in any pose and at any distance. As more potential objects require detection, this process becomes more complicated, reducing the applicability for collision risk detection. More generic methods for detecting known objects seek a 'class' of object, rather than a specific version. However, without an exact match it is impossible to extract the pose data required for depth estimation, making the result far less useful for collision avoidance. As such, it can be considered impractical to attempt to detect risks on an individual basis.

Generic object detection methods commonly use the principles of stereoscopic depth extraction. For this work, without multiple cameras available, this requires that multiple images are captured from a single camera whilst moving. Several methods are available which can perform generic object detection using a moving camera. However, the need to move the camera (and therefore the aircraft) is dangerous and therefore has been identified as being unsuitable for use within an aerodrome environment.

Instead, the methodology which has been chosen for this work uses the alternative approach of identifying regions of the image which do not pose a risk. By interpreting the captured image in its entirety, regions which could pose a risk can be determined by identifying all the regions which do not. The underlying process, of dividing the image into visually distinct parts, is known as image segmentation. As smaller segments should ideally represent a single object or material type, the scene will be easier to interpret, with each region being labelled as the most likely class based on its contents. Where identification is not possible or confidence in identification is low, the segment is to be considered a collision risk by default.

Once collision risks have been found within an image, they can be extrapolated from the image plane into real-world coordinates by seeking their closest point within the image. As most UAS have fixed height landing gear, the height and pose of the camera can be used to calculate range and bearing estimates. As a small variation in angle can result in a large error in range estimation, visual horizon detection can be used to correct for any variation in camera position. The remaining elements within the image which are not considered collision risks are essentially the ground surface. As the segmentation should isolate grass from taxiway and taxiway lines from asphalt, extracting features to compare against the aerodrome map allows the same technique to be extended for localisation improvement as well.

Chapter 5

Review of Image Segmentation Methods

In Chapter 4, it was established that Image Segmentation would be used to interpret visual data for both collision risk detection and localisation improvement. As it would be impractical to identify all object types that could be potentially encountered, the task of image segmentation focuses on terrain classification; establishing which areas within the captured image represent ground features such as asphalt or grass. By determining which elements of the image do not represent a risk, this works seeks to achieve generic collision risk detection through the use of 'negative space', where regions which cannot be identified are considered to be collision risks by default. This chapter reviews potential methods of achieving this, identifying why many established techniques are not suitable for this application. The next chapter (Chapter 6) builds upon the final approach selected in this chapter, and establishes the actual implementation used.

5.1 Image Segmentation for Terrain Classification

In order to detect potential risks through the detection of terrain features, the entire image must be classified. As the end result will partition the image into multiple regions (each assigned either a known class or labelled as an unknown), the term 'Image segmentation' is commonly used. Despite the word 'segmentation' suggesting that the process is a task of division, segmentation is more commonly approached from the bottom up, clustering similar pixels together. For an individual image, the optimal segmentation result can be achieved manually, assigning the relevant class to each pixel within an image. Figure 5.1 shows the results of manual segmentation applied to an aerodrome image. For this scene, only 6 classes of data are required to categorise the entire image in a manner which could be used for collision avoidance and visual localisation.

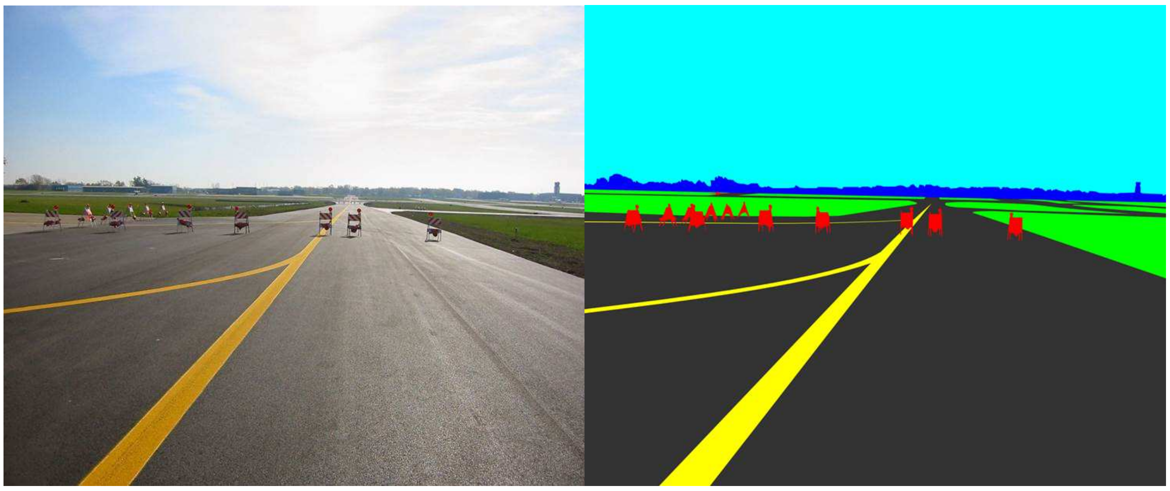

(a) Original Input Image (b) Manually Segmented Representation

Fig. 5.1 Example of manually-achieved ideal segmentation, using image of Waukegan National Airport, Illinois. [118]

For an automated approach, determining whether pixels are similar requires features to be extracted for comparison. Low-level features (such as colour) make clustering straightforward as each pixel already contains the required data. However, as a single object can have multiple colours, the resulting segmentation might not provide much benefit. Instead, higher order features are often more useful. For example, understanding the appearance of fur could allow dogs to be isolated within an image, regardless of the colour of the dog. Segmenting regions based on a understanding of its contents, rather than purely its visual appearance, is known as semantic segmentation.

A semantic approach to image understanding is highly beneficial for automated taxiing. For example, taxiway surfaces are often formed of multiple materials, with concrete and asphalt commonly used alongside each other. If image segmentation was achieved using colour data alone, the dark grey asphalt pixels and beige concrete pixels would be visually different, and therefore separated into different regions. However, if it was understood that both are part of the same taxiway surface, a better segmentation would be to combine them into a single region, as they serve the same purpose. An image where distinct regions are isolated and labelled in a way which reflects human interpretation is the goal of semantic segmentation.

The main objective for this chapter is to compare various image segmentation methods, in order to determine which approach produces the best segmentation result for aerodrome images. Ideally, the chosen approach will produce object specific segmentation, in which each object within the scene is isolated from it's surroundings. However, as this high level of accuracy is technically difficult, a subjective assessment of the quality of the output from each method will be required. If the ideal result cannot be obtained, the result will either be an under-segmentation or an over-segmentation. As an under-segmented image risks missing potential collision risks, an over-segmentation is preferable. If an image is over-segmented, each cluster should still only contain a single object type (i.e. class), allowing the scene to still be classified correctly.

Depending on the method employed, classification and segmentation can either be achieved together, or separately. For this work, the initial investigation is to determine whether to use a method which does both simultaneously, or to segment the image first and then classify afterwards. Regardless of the approach, to be viable the approach must:

- Avoid under-segmentation and ideally minimise over-segmentation.
- Preserve object borders from the underlying image.
- Avoid the use of deep-learning or non-deterministic methods.
- Not bias the classification process towards a particular class.

5.2 Classifier led Segmentation

Classifier Led Segmentation (CLS) is a term used to describe any image segmentation methodology where the final segmentation is based directly on classification results. By categorising pixels at a small scale (either individually or in small groups), larger regions within an image are formed where many neighbouring pixels share the same class. Without the need for a separate segmentation stage, data is only extracted from the image for classification, making CLS techniques highly efficient. CLS techniques can be broadly divided into two types; traditional CLS and modern CLS.

Traditional CLS techniques are some of the earliest forms of image segmentation, having been an area of active research for over 40 years [48]. These techniques rely heavily on small scale classification, comparing small sections of the image to known examples of other classes. As features are inspected at a very low level (typically using colour or some texture data) this limits the techniques to applications with consistently scaled features, such as for finding buildings and roads within satellite images [47].

Modern CLS techniques emerged from research for image categorisation for 'semantic retrieval'; enabling search engines to match images without requiring manual categorization of image contents. Early image categorisation algorithms often used image-wide content descriptions, which sought dominant visual descriptors throughout the image. This could produce a list of image contents that would allow classification, such as 'grass, animal and water'. However, with an ever increasing amount of images to categorise, more modern techniques (such as [90]) seek to establish a relationship between the elements of the image. For example, a dog on grass in front of water. As the position of unique regions with an image is determined with respect to each other, each region in the image must be isolated (i.e. segmented).

Over time, the two tasks have converged and techniques originally for one purpose are used for the other. For example some image segmentation methods [100] begin by examining the image as a whole to create an image level prior, which lists the most likely classes present within the image, based on dominant features. Further classification is then performed with the classifier biased towards these expected classes. Alternatively, methods such as [28] continue to use low-level features for segmentation into regions, but use high level categorization techniques to assign each region a final classification. From a brief review of current literature it is clear that the vast majority of modern semantic segmentation techniques commonly attempt to perform both image segmentation and classification simultaneously.

5.2.1 Prevalence of Machine Learning

As image segmentation methods have become more popular, there are already examples of using CLS for collision risk detection. Work undertaken in [8] describes a CLS method specifically designed for object recognition in urban environments, designed to detect and classify nearby objects for SDC collision risk detection.

Although this does indicate that modern CLS methods could be used for collision risk detection within aerospace, as with many modern CLS techniques, [8] uses an ANN to perform classification. Due to the complexity of fusing many different information types required for CLS, machine-learning approaches are commonly used. However, as the inner functionality of deep-learning methods cannot be easily defined/explained, they are difficult to certify or diagnose when unexpected behaviour occurs. As established in section 3.2.2, new technologies which include non-deterministic methods or deep learning have the additional regulatory barrier to overcome and are unlikely to gain regulatory approval for aerospace use.

Consequently, this work differs from the majority of modern semantic segmentation techniques as it only makes use of fully explainable methods for semantic segmentation. Although the majority of modern CLS approaches use deep learning, alternative CLS methods also exist, which are often examples of supervised learning. However, as the newest techniques predominantly use deep-learning, alternative techniques are often based on methods from traditional CLS and have therefore inherited many of the problems.

5.2.2 Known Class Biasing and Border Accuracy

The use of deep-learning methods notwithstanding, issues associated with CLS approaches impact how suitable they would be for collision risk detection. Issues specific to this work emerge from the decision to infer collision risks from unknown regions. Once a collision risk has be isolated within an image, the position of the segmented region borders are used to extrapolate the range and bearing to the potential risk. As such, accurate border extraction is critical. However, as the data within an unknown region cannot be used to help define the border, the limits of any potential collision risk can only be defined by the boundaries of surrounding known classes. Therefore, accurately segmenting 'unknown' regions within an image is just as important as defining known classes.

Despite the previous use of CLS for collision risk detection, precise region extraction is often difficult when using classification data to directly segment an image. Inaccurate region borders are partially introduced through the use of feature descriptors. Despite being data rich, images are comparatively information devoid, with a great number of pixels required to provide any significant feature useful for recognition. Although colour data is stored in each pixel, more complex data, such as texture, requires data from multiple pixels. To ensure pixels are sampled in a consistent manner, a feature descriptor is typically 'slid' around the image to sample pixels consistently. Regardless of the method used, the descriptor must be large enough to encapsulate a visually definable amount of texture, sometimes requiring several thousand surrounding pixels (for example, [112] suggests a 50x50 pixel sliding window which would process 2500 pixels each sample).

Prior to segmentation, a supervised learning stage is required to create a model for each class. This achieved by applying the feature descriptor to a training-specific labelled data-set. As the training set should be an accurate sample of each class, the model should reflect the training set accurately. However, as the training set is also limited, the model must be able to 'generalise', in order to correctly classify data outside of the training set. As these requirements are in opposition, a point must be established which allows the model to generalise but still closely adheres to the known examples. This problem is known as the Bias–Variance Tradeoff (BVT) and is a well known problem in supervised learning.

If the model adheres too closely to the training data, it increases the chances of over-fitting to trends in the training set, which are not representative of the class as a whole. During segmentation, this would result in many unknown regions which would produce many false-positive collision risk detections. In contrast, simplifying the model by enclosing more of the feature space may under-fit, resulting in a model which captures elements from beyond its class. When this is the case, individual pixels are often sufficiently close to multiple classes to be considered a match, with the class with the highest confidence claiming the pixel. As visual training sets typically require vast amounts of data for training, models with high-bias (i.e. under-fitting) are more common in image processing. As such, classifiers will tend to bias towards known classes, rather than unknowns.

When feature descriptors extract data from the centre of a region, there are minimal effects of under-fitting as the large sample size helps to provide enough data to achieve the correct classification. However, when the feature descriptor is applied near region boundaries, it can capture pixels from multiple regions at the same time. When the feature descriptor overlaps multiple classes simultaneously, if classifier confidence is higher for one class (which is always true when the other region is an unknown), fewer pixels are required to qualify as a known region. This results in inaccurate region borders, with the known class region ‘encroaching’ into the unknown region. Results from [100] and [46] show that segmentation based on texture feature descriptors often shifts regional boundaries from their original position. Where regions are small enough, this can result in them being entirely misclassified, with small regions becoming absorbed into larger neighbours. As even small objects can pose a risk, this form of misclassification could directly endanger the UAS. Despite the widespread use of CLS, as the goal of this system is to able to detect generic collision risks, relying on expected classes for segmentation was deemed inappropriate.

5.3 Untrained Segmentation

As semantic segmentation is a goal, not a process, there are many methods which can be used. Although multiple forms of CLS are available, the tendency to bias classification and shift edges is difficult to avoid when relying solely on classification data during segmentation. In an attempt to correct for these issues, some CLS techniques incorporate a secondary stage to re-adjust region borders. (Work undertaken in [101] and [127] makes use of edge detection to to correct region border shift by moving the border to a nearby edge).

To avoid introducing the same classifier-based issues, region borders are extracted from the original image using low-level image features; using either direct pixel values or neighbouring pixel relationships. Working with extremely small features ensures that the border-shift issues common with CLS do not occur. Accordingly, the most precise region boundaries can only be extracted by directly comparing values of individual pixels (i.e. brightness or colour) and therefore pixel-level border regions should be achievable.

Low level features are often very capable of extracting region borders. As such, it is entirely possible to carry out segmentation without using classification data. As directly comparing pixels does not require a classification model, there is no training stage and these forms of segmentation are often described as 'Untrained segmentation'. By separating segmentation and classification, an alternative approach is to perform segmentation first, using only low-level image features. After segmentation is complete, the image will be divided into many distinct regions (as with CLS), each of which will still require classification.

Provided that the untrained segmentation is capable of correctly segmenting a scene (each cluster should only contain visual similar regions of the original image), the contents of each cluster can be assumed to represent a single class. This allows every pixel within each cluster to be used during the subsequent classifications stage. Should any cluster be malformed so as to contain more than one class, at least partial misclassification is guaranteed as no further segmentation will be introduced. Instead, the entire cluster is assigned the most likely class (or as an unknown). Therefore, when relying on low level features alone, it is often better to err on the side of caution, and produce an over-segmentation such that each cluster is more likely to only represent a single object or part of one.

The main disadvantage of an initial over-segmentation is that it requires more classification stages (in accordance with the increased number of regions) and each region has less data to use. However, even if the initial segmentation is an over-segmentation, as each cluster should represent a single class, the classification stage can use the entirety of each cluster as sample data. If the segmentation is good, classification should be very accurate. Semantic understanding can then be introduced during the classification phase, where each cluster is categorised based on its contents. After classification, neighbouring regions of the same type can be merged, producing a result more compa-

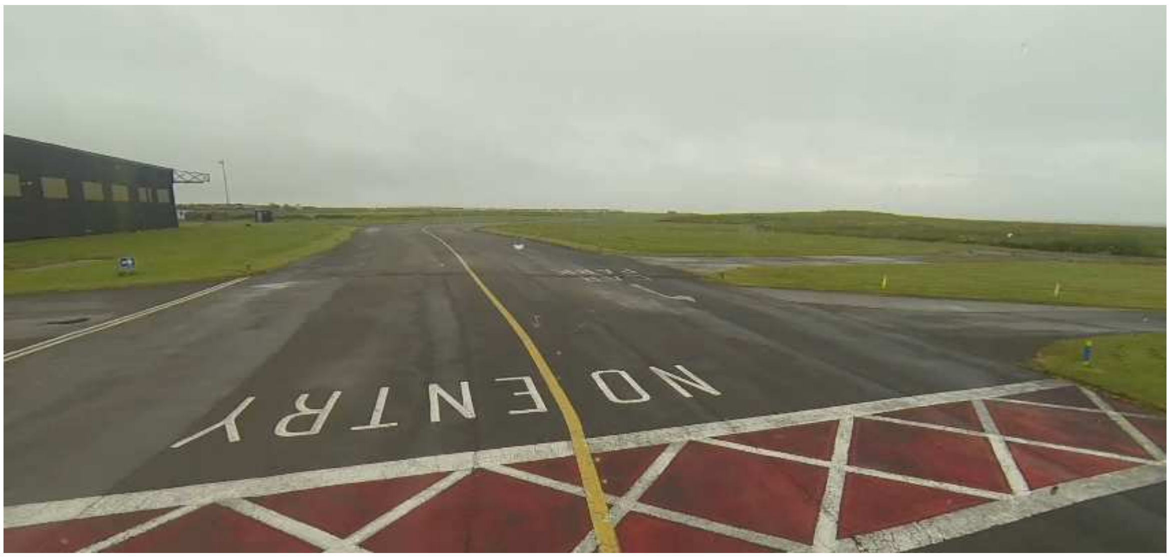


Fig. 5.2 Example image which will be used to display the output of different segmentation techniques.

rable to human segmentation. As the data used for segmentation is different to that used for classification, additional processing stages are required. This makes the process less efficient than CLS. However, the precision of edges and the ability to use well-established methods make this a more viable alternative.

As image segmentation has been an area of research for decades, untrained segmentation methods that produce pixel level accuracy are well established. Therefore, a review of several segmentation methods will follow, assessing the suitability of each technique to use in an aerodrome environment. Unlike CLS methods, which could be directly compared to examples of manual segmentation to determine accuracy, the overall success of untrained segmentation depends both on the segmentation stage and the subsequent classification stage. As such, a numeric representation of the performance of each method cannot be obtained. Instead, the perceived quality of the segmentation will be based on the perceived accuracy. Figure 5.2 shows an image captured at Walney Airport, which will be used to compare the results of multiple untrained segmentation techniques. Additional consideration will also be given to the difficulty of obtaining an accurate segmentation.

5.3.1 Edge based segmentation

Although it was stated at the beginning of this chapter that ‘segmentation is commonly approached from the bottom up, clustering similar pixels together’, an alternative approach is to directly seek the borders present within the image, i.e. to use edge detection. When distinct regions are present in an image, adjacency between regions usually creates a strong discontinuity in colour or brightness. Edge detection filters (such as the Canny filter [20]) are designed to produce a ’filter response’ image which highlights discrepancies between neighbouring pixels. To ensure only edges are detected, the filter further increases the response when other neighbours also have a strong response, maximising the chance of detecting a consistent edges whilst reducing the response of anomalous individual pixels.

Once the raw edge detection response has been acquired, the ‘strength’ of a detected edge is apparent by the values of the pixels. Depending on the application, weaker edges may either be useful or discarded. As such, a user defined threshold is applied, creating a binary image in which every pixel response is assumed to represent an edge. As the clearest edges in most images are found at a perimeter, edge detection is useful for defining the boundary between different objects. Edge detection is also highly efficient and can be accomplished very quickly.

Although the results of edge detection may appear segmented, none of the pixels within the image have been grouped. To accomplish this, areas enclosed on all sides by edges can be considered a single cluster. As this appears much like contours defining similar elevation on a map, the process is often referred to as ’contour’ detection. Contour detection is far more complex than edge detection, with multiple possible approaches. A possible approach involves:

- Morphological closing to ensure there are no gaps in the edges.
- Skeletonisation to determine junctions and connectivity between junctions.
- Loop detection, using the connectivity between junctions and treating the entire image a graph.
- Region reconstruction, combing the edges which form the smallest loops to create unique regions.

Despite the number of stages, ‘contour detection’ can be highly efficient. As the process is based on junctions and edges, the vast majority of an image is not used during the contouring stage. This allows for even a large image to be segmented quickly.

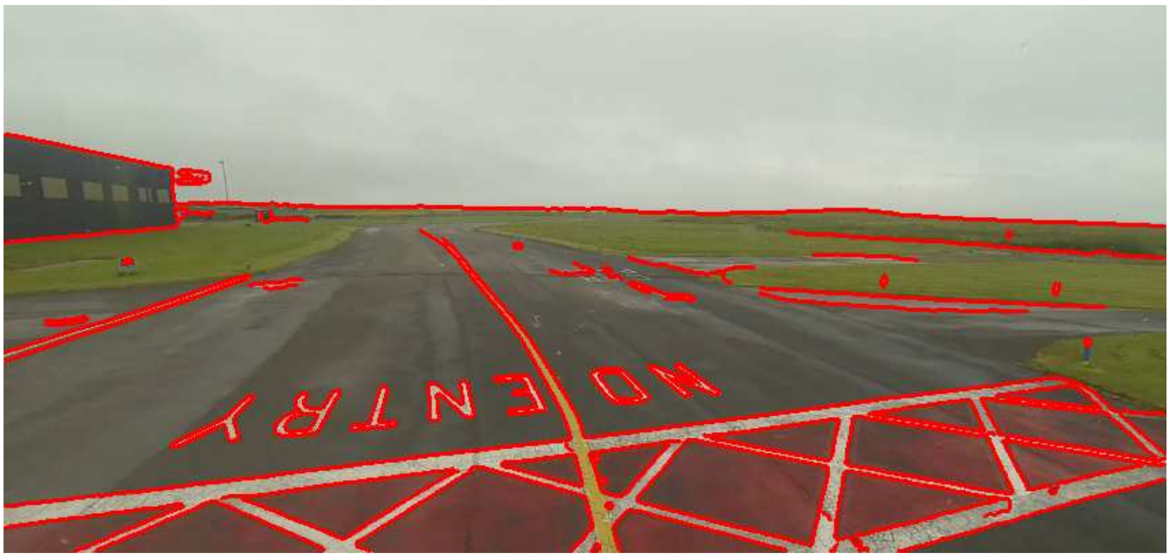


Fig. 5.3 Canny edge detection, seeking only ‘strong’ edges

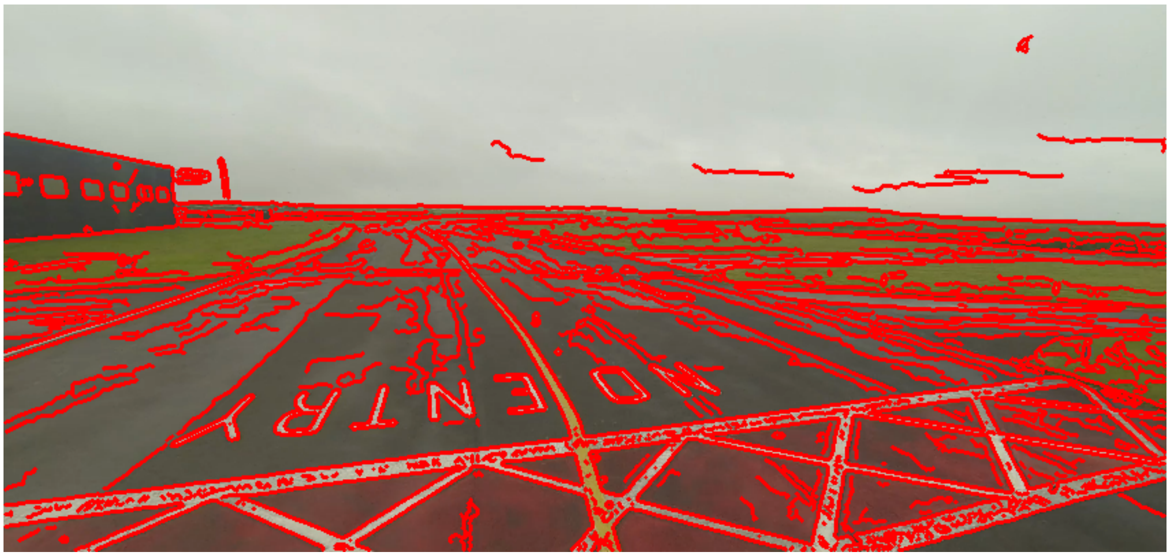


Fig. 5.4 Canny edge detection, including ‘weaker’ edges

Difficulties

The fundamental problem with edge based segmentation is consistency. In order to detect an object, it must be isolated as a region (or several smaller regions) within an image. As edge detection methods rely on strong discontinuities, this requires a continuous border around each object. However, in outdoor environments (such as aerodromes) strongly defined borders are not guaranteed. Over distance, effects such as haze and atmospheric scattering result in lines becoming less distinct. During testing, it was found that the clarity of the edge between taxiway and surroundings reduced towards the horizon. This resulted in continuous regions containing both taxiway asphalt and grass borders, as shown in Figure 5.3.

To properly define regions, weaker edges must also be captured. This is achieved by lowering the response threshold to include weak edges. As shown in Figure 5.4, lowering the threshold significantly does capture the border between runway and grass as an edge, all the way to the horizon. However, it also has the adverse affect of introducing many more edges within other parts of the image. In order to capture borders between each type of terrain, the edge detection method produced an over segmentation. Highly textured materials, such as asphalt or grass, result in many edges being detected despite not representing the border of a region. The small regions created are often too small to provide sufficient information for accurate classification, whilst taking substantial time to extract using the method outlined above.

Despite additional testing, consistently extracting edges was found to be difficult. Variability in lighting conditions, weather, camera motion and the neighbouring classes of objects combine to make edge extraction results difficult to use outdoors. As such, other untrained segmentation approaches will be considered which do not use edges, but instead group similar pixels together into clusters.

5.3.2 Colour/Intensity Quantisation

A more common method of forming regions within images is to ‘cluster’ pixels together, based on similarity. In the simplest implementation, regions can be formed by comparing the colour (or intensity) data of each pixel within the image, without including additional information, such as connectivity or cluster size. However, attempting to group pixels on their original colours is usually impractical. As modern cameras use 8-bit colour, each colour channel has 256 possible colour vales. With most camera using three channels (RGB, YCbCr etc), there are approximately 16 million possible colours per pixel.

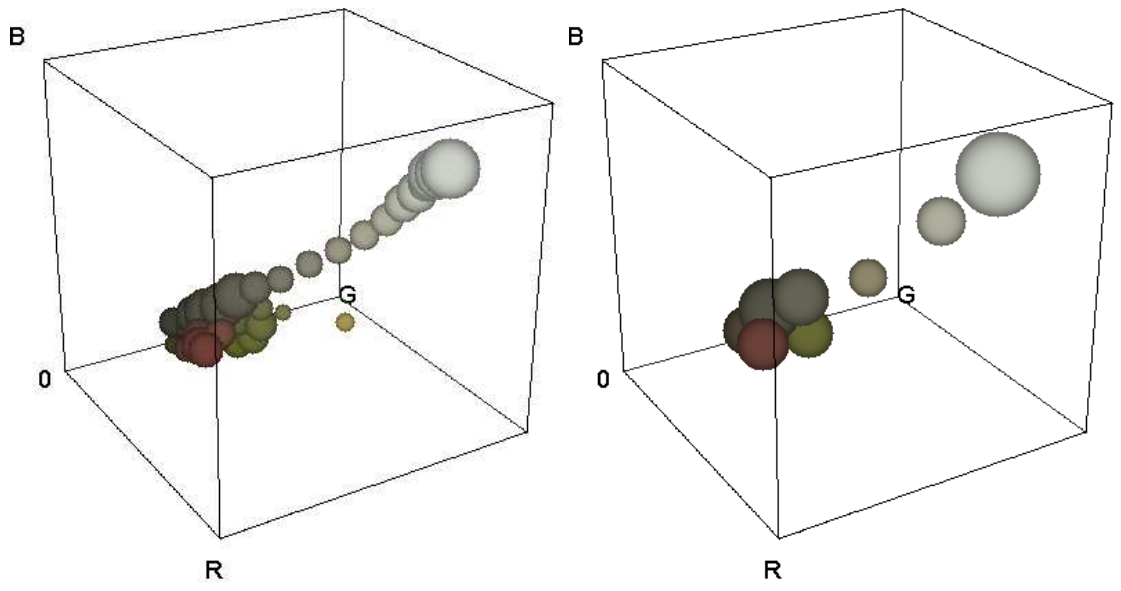
Therefore, as it unlikely that many pixels will share the exact same colour, the number of overall colours must be reduced.

Different methods of reducing the number of colours within an image has resulted in different terminology being used to describe very similar tasks. The term ’Thresholding’ was originally used, as it described the process of determining the threshold which would convert a grey-scale image into the optimal binary-image output. Techniques such as the Otsu method [83] were extremely successful and are still commonly used when attempting to separate image foreground and background. For colour images, the multi-Otsu method built upon the original by establishing multiple thresholds within colourspace, continuing to promote the use of the term ’Thresholding’. However, many other techniques do not use thresholds, such as K-means clustering (which has now been proven to be equivalent to Otsu [66]) and histogram peak detection. As such, the most suitable term to encapsulate all of these methods is ‘colour quantisation’. For this work, the quantisation method chosen for testing is Wu Quantisation [121], (which is a threshold based method, repeatedly pi-bartioning the RGB colourspace using variance minimisation). The Wu method was chosen as it is commonly used in computer graphics due its tendency to preserve the appearance of the image for human interpretation, despite a large reduction in the number of colours.

For simplistic images with distinct regions of solid colour, pure quantisation methods can efficiently segment an image into distinct regions. When isolated against a uniform background of a different colour, pixel based quantisation can be coupled with ‘connected component’ analysis to determine distinct objects in the foreground, making these methods widely used in industrial/manufacturing processes settings. However, when used on more complex scenes, such methods are rarely successful at achieving a good segmentation. As similar colours can be dispersed throughout an image, pixel groupings often lack spatial cohesion, leading to cluster being comprised of pixels scattered throughout the image, rather than a connected region. Outside of extremely simplistic conditions, colour quantisation alone creates poor segmentation results.

Figure 5.5 shows a 3D representation of the colourspace used by the example image in Figure 5.2. In Figure 5.5a the colourspace is reduced to 32 distinct colours, before being further reduced to just 8 in Figure 5.5b using Wu Quantisation.

Despite a reduction from 16 million to only 32 colours, the remaining colours are still similar enough so as not to form distinct areas within the image. Instead, where the original pixel colours are near equidistant to multiple quantised colours, pixels will alternate between quantised colours depending on which one is closest. As such, the results



(a) Colourspace quantised in 32 colours (b) Colourspace quantised in 8 colours

Fig. 5.5 3D RGB colour quantisation representations of the example image

of connected component analysis would generate thousands of small regions, many of which are simply individual pixels. Even where the majority of one area is a single colour, fragments of other clusters will be present within the larger region, making any task of separation very difficult.

A possible method to impose spatial constraints through quantisation alone would be to decrease the number of colours available. Further reducing the image to just 8 colours subjectively improves spatial clustering, as shown in Figure 5.6. However, all yellow and white taxiway markings have now been combined into a single cluster, which includes part of the sky as well as some asphalt. As such, quantisation alone is not very suited for untrained segmentation.

Some recent work attempts to improve segmentation through quantisation by representing the fragmented groupings as Gaussian ellipsoids [74]. This approach favours areas dense with a single colour while fragmented outliers are lost, presenting a highly efficient method of isolating regions of similar colour within the image. The resulting ellipsoids sufficiently convey the colours and locations of regions within the image such that they can be used to classify the type of road a car is currently travelling along (e.g. urban, rural or offroad). However, the applicability of this approach to automated taxiing is limited, especially considering the loss of shape data through the Gaussian conversion would make accurate tracking of collision risks or terrain features difficult.

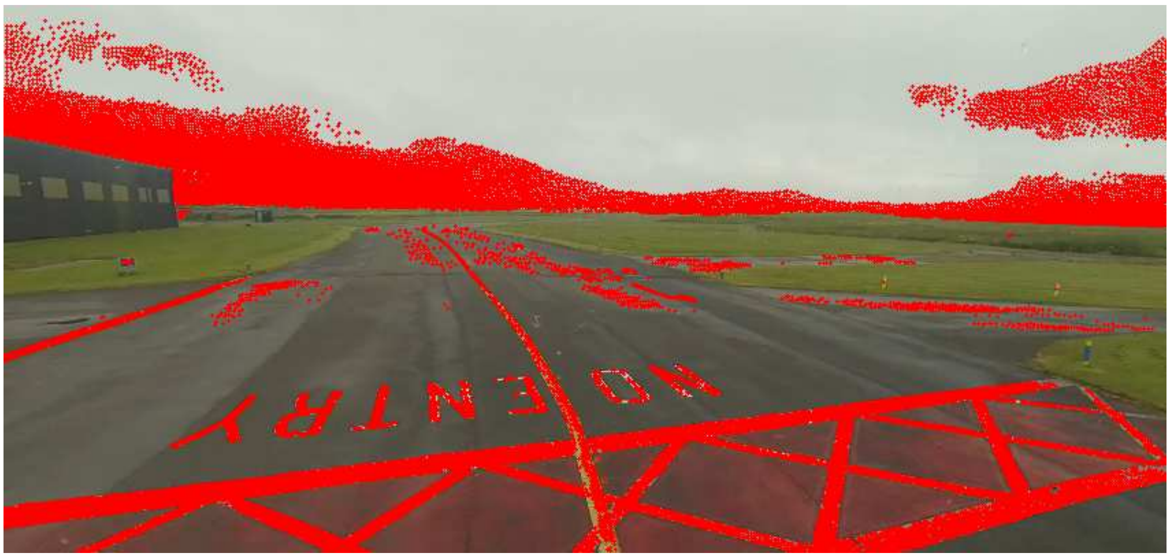


Fig. 5.6 A single 'cluster' produced by colour quantisation

5.3.3 Unsupervised Region-Growing

The use of image-wide features, such as the most common colours, is only appropriate for tasks which are not focused on low-level details, such as file-size reduction. Although some of the results of pure colour quantisation are acceptable, as the colours are chosen before the segmentation process, they cannot adapt to regions of the image. To define regions more precisely, techniques which work at local level are more appropriate. One method of localising clustering is to ensure that all pixels which make up a cluster are not only similar in colour, but are also connected together. This is commonly achieved through region growing algorithms, which segment images based on similarity between neighbouring pixels.

Specific implementations use slightly different methodologies, but region growing methods typically begin with a random selection of starting pixels, each of which is compared to their immediate neighbours. If the difference between the neighbours and the original pixel is less than a preselected threshold, clusters are formed consisting of the connected pixels. This process continues (either using colour distance to the starting pixel or to the current cluster mean) until no more pixels can be grouped.

As clusters only continue to grow while neighbouring pixels are below the colour threshold, edges are formed between regions of distinct colours. This results in region growing segmentation adhering to strong edges in much the same way as edge detection methods. In addition, when a single pixel is within threshold range of multiple clusters, it is grouped into the cluster which is most similar. Therefore neighbouring clusters should

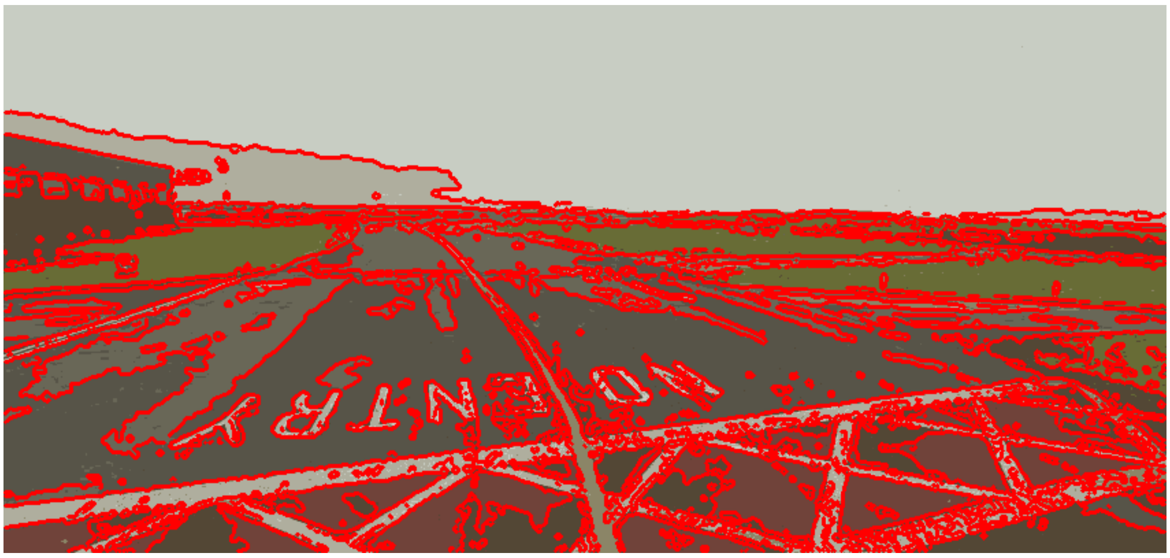


Fig. 5.7 Region growing segmentation with superimposed edges

produce a well defined boundary between two regions, even if they are separated by a weak edge.

For this work, the HALCON region growing algorithms provided by MVTec software were used [79]. Compared to colour quantisation and edge based segmentation, region based segmentation methods are computationally intensive as they require comparisons between individual pixels pairs. However, as the results are usually more consistent and appear better through qualitative human assessment, they are more popular.

Limitations

Region growing methods provides a subjectively better result when compared to 'edge based segmentation' or 'colour quantisation'; as shown in Figure 5.7, the results consist of fully connected regions of a reasonable size, which also accurately adhere to the border of the original image. As such, pure region growing methods could be used as the segmentation stage for this work.

Nevertheless, simply because the result is usable does not suggest that it cannot be improved upon. To maximise the chance of correct classification, a single cluster should cover as much of an object as possible so as to provide the most data. However, region growing algorithms produce over-segmented results in images with gradual changes in colour. Although abrupt colour changes are highly likely to represent the meeting of different objects, smooth colour gradients are generally indicative of a single object. (For example, objects with curved surfaces often have gradual colour changes due to lighting).

As regions will stop growing when neighbouring pixels exceed the permissible threshold, arbitrary boundaries will be created in regions with colour gradients. For large regions, this can occur multiple times, introducing parallel clusters next to each other. This creates ’banding’ within smooth areas, with the end results being unnatural to human vision. As the atmospheric effects of Rayleigh scattering result in colour changes towards the horizon, it was found during testing that region growing methods divided taxiways into multiple segments.

This mild over-segmentation is not a total barrier to the use of region growing for the segmentation stage. However, over-segmentation risks the increased change of misclassification. If a taxiway is broken into numerous parts, the probability that a false positive risk detection will occur increases. Therefore, methods of unifying clusters which represent a single object were explored.

5.4 Object Specific Segmentation

The ideal goal of the segmentation process is for each object within the original image to be represented as a single cluster. This form of ’Object Specific Segmentation’ would allow the entire object area to be used during classification, whilst also ensuring that only pertinent data is used. As such, methods of improving upon the results of region-growing classification were investigated.

As the region-growing technique produced an over-segmentation, a possible approach would be to ’re-cluster’ the smaller segments into larger regions which only represent a single object. However, this is evidently very difficult; as each cluster has stopped growing, they have likely reached the maximum variance allowed by region growing threshold. This would make the average data in each cluster too dissimilar to be grouped on colour alone. Alternatively, as the over-segmentation introduces region borders where no clear division exists in the original image, edge detection could be used to confirm actual borders. This again returns to the issue that edge detection requires strong edges which may not always be present on features further from the camera. Through testing it was observed that neighbouring regions with dissimilar contents but a weak border were incorrectly combined using this method, as shown in Figure 5.8. As an under-segmentation will always produce an incorrect result, the use of edges during re-clustering was not pursued.

As the original over-segmentation was the result of the region-growing algorithm stopping before isolating an entire object, adjustments to the region-growing algorithms were

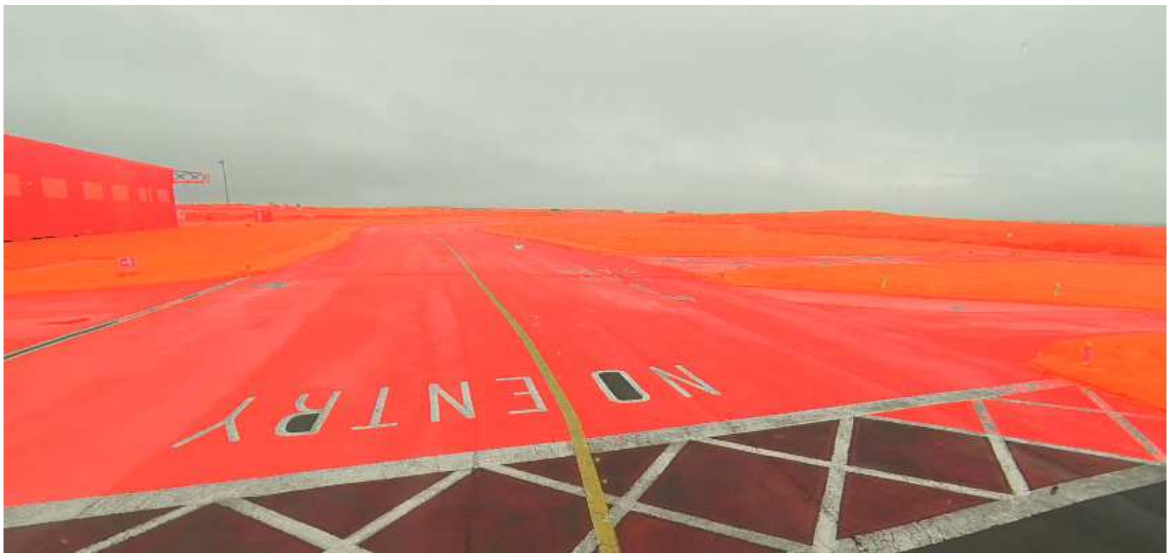


Fig. 5.8 Results of attempting re-clustering using strong region edges to separate regions

also explored. As most region growing algorithms compared new pixels to either the ‘seed’ or the current region mean, the region stops growing when new pixels become dissimilar. An alternative approach is to compare each new pixel to the pixels which currently form the region border. Regions are then formed of pixels which, whilst not necessarily similar to each other, are connected to each other via a sequence of connected pixels that do not differ more than a threshold. Due to the similarity to methods used in graph theory, this is referred to as ‘reachability’ clustering.

However, attempting to modify region growing algorithms to incorporate this change has proved unsuccessful. Although clear edges will continue to separate regions, reachability significantly increases the likelihood that neighbouring clusters will combine. If any part of the border is sufficiently gradual to allow a reachability connection to be formed, the two clusters will be combined into one. This is especially true for regions near the horizon, due to the atmospheric effects shift pixel hue and saturation. Attempting to apply reachability directly to traditional region growing usually results in a significant under-segmentation.

Although reachability cannot be applied to large clusters (as they are already too dissimilar) nor at the pixel level (as they are too similar) a solution is to apply reachability re-clustering after over-segmenting the image into small clusters. Smaller clusters are more likely to be similar to their neighbours, allowing a reachability approach to include them when forming a large cluster. At the same time, using the average data of many pixels lowers the likelihood of forming a connection across object boundaries. Therefore, an over-segmentation approach followed by reachability clustering is seen as an appropriate method of obtaining object specific segmentation without classification, and has been explored in more detail in the following section.

5.4.1 Over-segmentation

Beginning with a deliberate over-segmentation requires a different approach compared to the previously described image segmentation methods. As the clusters are to be recombined into larger regions, the initial over-segmentation must meet certain requirements; specifically the size of the clusters is of critical importance if a reachability approach is to be used. Each cluster must:

- Be large enough that the contents will be dissimilar to clusters of a different object, even without well defined borders in the original image
- Be small enough that the variance between neighbouring clusters of the same class is low, allowing them to be easily grouped into larger segments later on.

The easiest method of ensuring that each cluster is the correct size would be to divide the image uniformly, using either a rectangular or hexagonal grid. However, this extremely simple 'Grid Segmentation' approach has significant drawbacks, primarily due to the grid not adhering to the underlying borders within the image, shown in Figure 5.9. During the subsequent reachability clustering, the edge of the large region can only adhere to the edges of the merged sections. Therefore, the border of the region will not be aligned with the edge of the object within the original image, leading to depth estimation issues and introduce data into the larger region which may hinder classification.

Further more, any grid segment which lies on the border of two objects may not be clustered into either region due to the average data being significantly different from both. If the segment is not clustered, it is likely to be identified as an 'unknown' potential risk as it contains conflicting data from multiple classes. This tends to result in many small collision risks being falsely detected throughout the image, as shown in Figure 5.9 where a 10 by 10 pixel grid results in many segments containing multiple classes. These issues demonstrate how important accurate clustering is, even at the smallest scale. The over-segmentation must not only be regular in size, but must also be consistent in colour in order to adhere to object borders.

Due to the limited applications of clustering at a small scale, there are not a great number of methods which quickly divide an image into small clusters, whilst still adher-

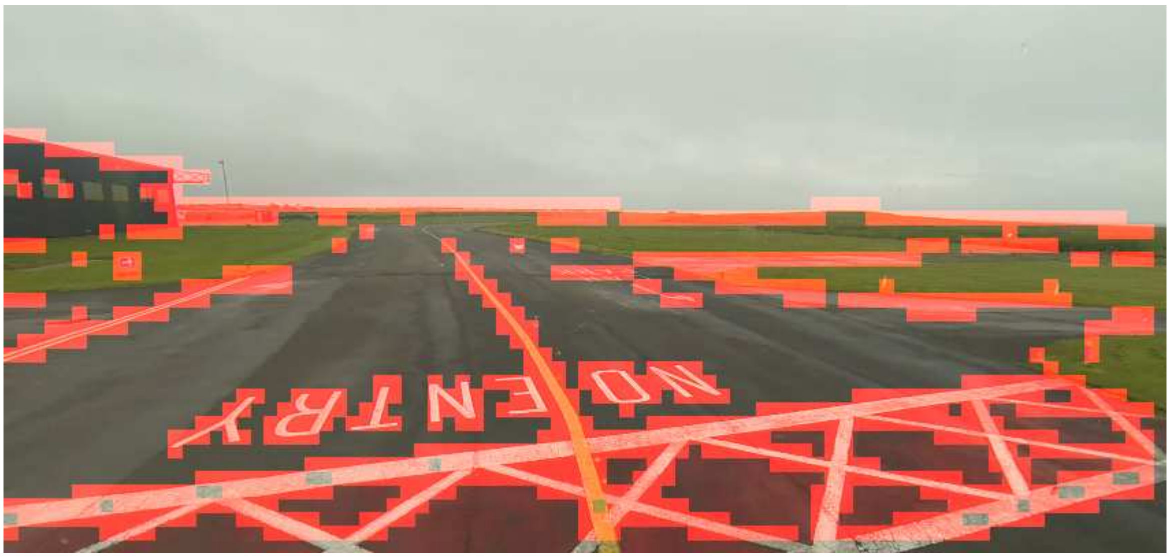


Fig. 5.9 Segments produced by a grid which contain multiple classes
[Segments produced by a grid which contain multiple classes and would likely produce a collision risk result by default. Achieved using a 10 by 10 pixel rectangular grid.]

ing to the underlying image. Therefore, the only two potential methods were explored; ‘watershed’ segmentation and ‘superpixel’ segmentation.

Watershed Segmentation

’Watershed’ segmentation is a method which uses a similar concept to ‘Edge based segmentation’, but interprets the data an alternative way. As described in Section 5.4.1, edge detection works by applying a filter to an image, in order to determine the colour gradient for each pixel compared to its neighbours. As local maxima are indicative of edges, a threshold can be applied which defines edges based on the magnitude of the response. For edge based segmentation, the intention was to directly segment the image into large regions, requiring only the strongest edges to be used.

As the goal is now to produce an over-segmentation, more edges can be detected if the threshold is lowered significantly. However, edge detection alone is only half the task; self enclosing ’contours’ must be present in order to form segments. As shown in Figure 5.10, even with extremely low threshold values (0.1% of the maximum value range), edge detection does not reliably form self enclosing regions. As such it is clear that ‘edge base segmentation’ is not capable of producing a usable over segmentation without suitable edges being present in the original image.

Despite this, the gradient image still contains useful information that can be used for

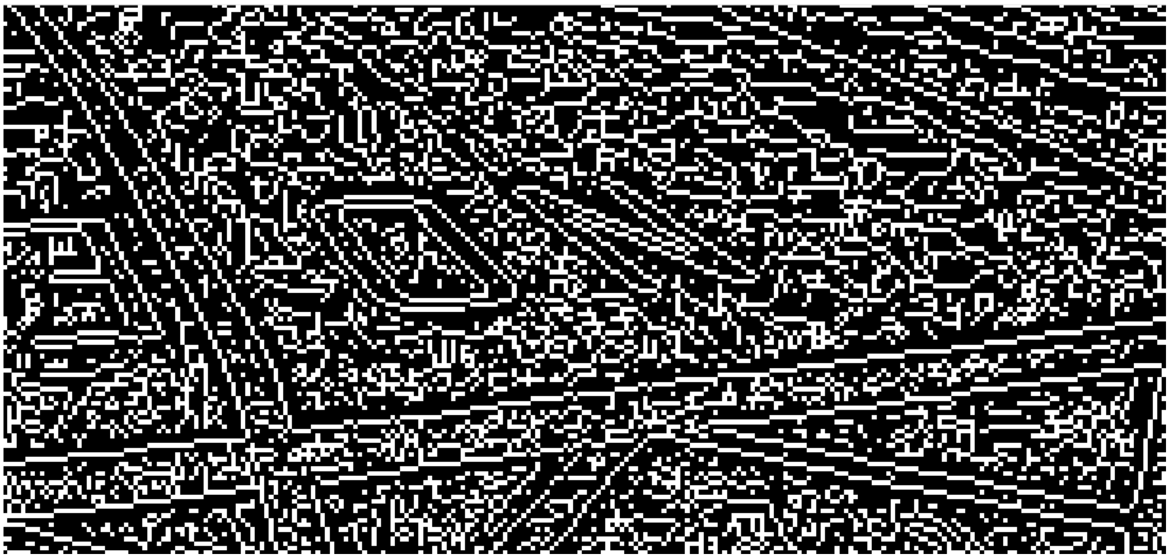


Fig. 5.10 Enlarged depiction of the results of edge detection using a low threshold

segmentation. Watershed segmentation uses the same filter response image, but rather than seek the maxima, it uses the gradient minima. Unlike maxima which occur when neighbouring pixels are dramatically different, the minima occur when neighbouring pixels share the same value (or are at least very similar). Apply a threshold to isolate these pixels produces an image where regions of minimal variance have been grouped into ’blobs’. Returning to the original image, these blobs indicate highly consistent regions which are suitable as the basis for segments. The remaining process is to expand the borders of each segment to include pixels with greater variance.

In geography, the term ‘Watershed’ refers to ridges on a landscape which separate the catchment areas of rivers. By treating each image region as a catchment area, the ’Watershed’ segmentation approach determines the position of lines which would best divide these areas. This is most commonly used for full image segmentation where large, distinct regions already present. For example, when applied to a simple binary image, the original clusters expand based on distance alone grouping each new pixel into the closest cluster, as shown in Figure 5.11.

For greyscale or colour images, the gradient of the filter response is also included to ensure the boundaries are placed in the most suitable location. Because the watershed edges are extrapolated from minima regions, they are not directly based on the maxima within the filter response. However, as the boundaries are established on both spatial distance and colour gradient, they will tend to align with strong edges within the original image.

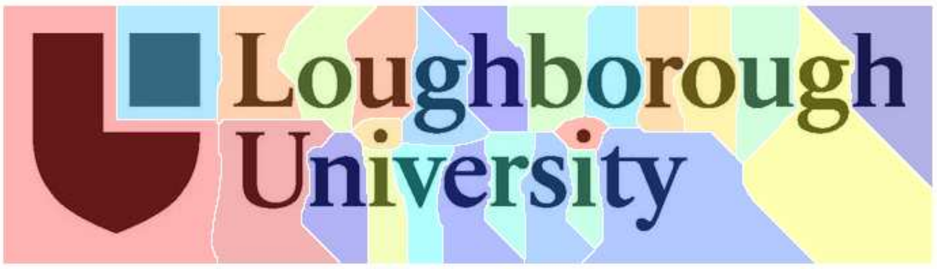


Fig. 5.11 The results of Watershed segmentation applied to the Loughborough University Logo

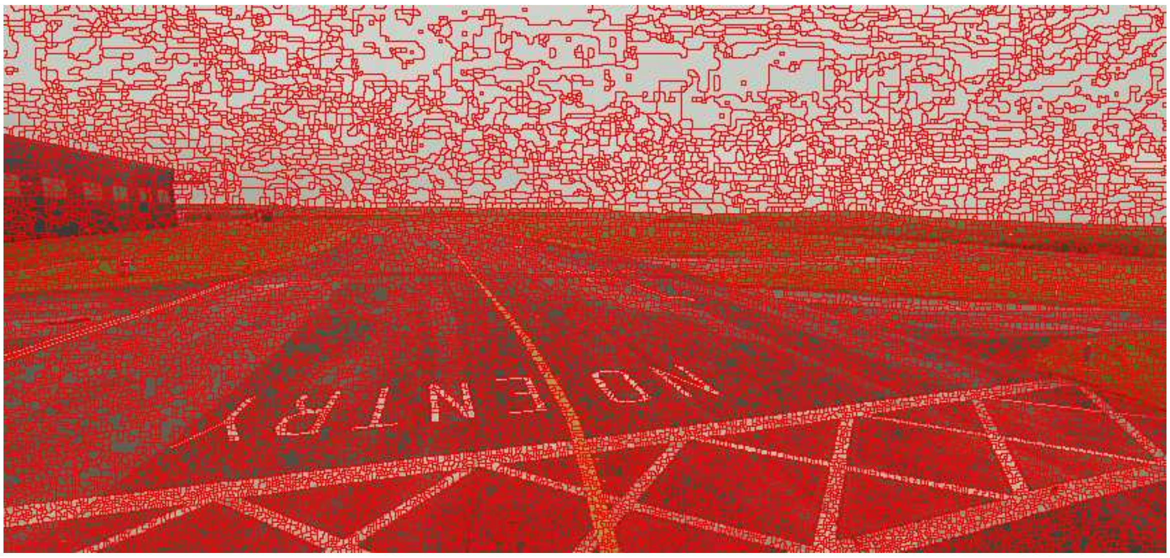


Fig. 5.12 Watershed based segmentation

Due to the images being stored digitally, a combination of colour discretisation and compression artefacts determine the exact location of the minima. As no preprocessing is applied to the aerodrome images, the minima regions are typically very small, producing small clusters which adhere to the underlying contents of the image. This results in the desired over-segmentation, as shown in Figure 5.12.

Despite this, the Watershed segmentation approach does have issues; primarily there is no method of controlling the number or size of the clusters produced. Without any form of pre-processing, the size of output clusters depends directly on the image texture within the image, resulting in highly irregular segment sizes for images of multiple object/material types. Although some clusters are already fairly large (which reduces the need to re-cluster) different materials within the image can result in other clusters being extremely small.

For highly textured surfaces, such as asphalt, this produces a dramatic over segmentation, with some regions as small as just a few pixels. During the subsequent reachability clustering, extremely small clusters are similar to individual pixels in that they risk creating a 'reachability path' between two distinct regions. In addition, as re-clustering requires that each segment is compared to it's neighbours, smaller clusters increase the overall cluster count, which increases the computational requirements. Although Watershed segmentation is a viable method of over-segmentation, alternative methods which offered greater control over the number of clusters created were also explored.

Superpixel Segmentation

Segmentation techniques which establish clusters *solely* on the contents of the image are unable to regulate cluster size. A single segmentation process (such as Watershed segmentation) could result in either many small clusters or just several large clusters, depending on the image contents. By comparison, if the desired number of clusters could be specified before segmentation takes place, the approximate number of pixels in each cluster could be determined based on the size of the image. As such, methods which attempt to achieve a specific number of segments are more suitable for producing a similarly sized clusters. This form of segmentation, which tailors the result based on user parameters, is a relatively new form of untrained image segmentation, often referred to as 'Superpixel' segmentation.

The term 'Superpixel' was coined due to the ability to use small clusters in lieu of pixels for many tasks. By generating a large number of very small clusters (which follow object borders), the average data for the cluster should be highly representative of each pixel within it. If every pixel in the original image is grouped into superpixels, an image with millions of pixels can be reduced to a meaningful representation of only a few hundred superpixels, without loosing much colour information. Applying machine vision techniques at the superpixel level can produce similar results to the pixel level, but at a much lower computational cost. (Although simply rescaling the image could also speed up processing, the original borders would not preserved, preventing additional data, such as image texture, being extracted at a later time).

Despite superpixels being a relatively new concept, there are already many different algorithms available to generate them. For example, 'Graph based' superpixel creation involves representing an entire image as a graph, in which every pixel within an image is represented as a node (such as Felzenszwalb [39]). The edges between nodes represent neighbouring pixels, and the edge weights are assigned based on colour similarity of these pixels. Superpixels are then created by minimising a cost function over the graph in its entirety. Alternatively, 'Gradient-ascent-based' algorithms can also be used, in which clusters of pixels are iteratively adjusted to maximise border gradients whilst maintaining the average cluster size (such as Quick-Shift [114]).

Determining the best method of generating superpixels is made difficult by the lack of any clear metric with which to compare different techniques. Essentially, the chosen approach must satisfy the following conditions:

- Adherence to original image borders
- Computational simplicity
- Cluster size regularity

Although an empirical review of superpixel methods could have been undertaken, one superpixel method is far more widely used than others due to excelling at all three requirements. The current state-of-the-art in superpixel generation is widely considered to be Simple Linear Iterative Clustering (SLIC). Compared to other superpixel generation techniques, SLIC is highly computationally efficient, with the work done in [87] demonstrating that a standard desktop computer (circa 2011) provides sufficient computational power for SLIC to process high resolution images in real time, easily matching the typical 30Hz refresh rate of cameras. SLIC produces superpixels of consistent size and of minimal colour range, while adhering well to region boundaries. Although difficult to quantify, it is subjectively considered to produce a better end result. Therefore, SLIC was selected as the over-segmentation approach. Figure 5.13 is an example of superpixel segmentation achieved using SLIC.

5.4.2 Reachability Clustering

As stated at the beginning of Section 5.4, the goal of the entire segmentation process is to achieve 'Object-specific-segmentation' in which unique classes within the image are separated. Following on from Section 5.4.1, as each superpixel should only represent a single class of object, direct classification of superpixels is a potential method of achieving this goal. However, as each superpixel is fairly small, there may not be adequate

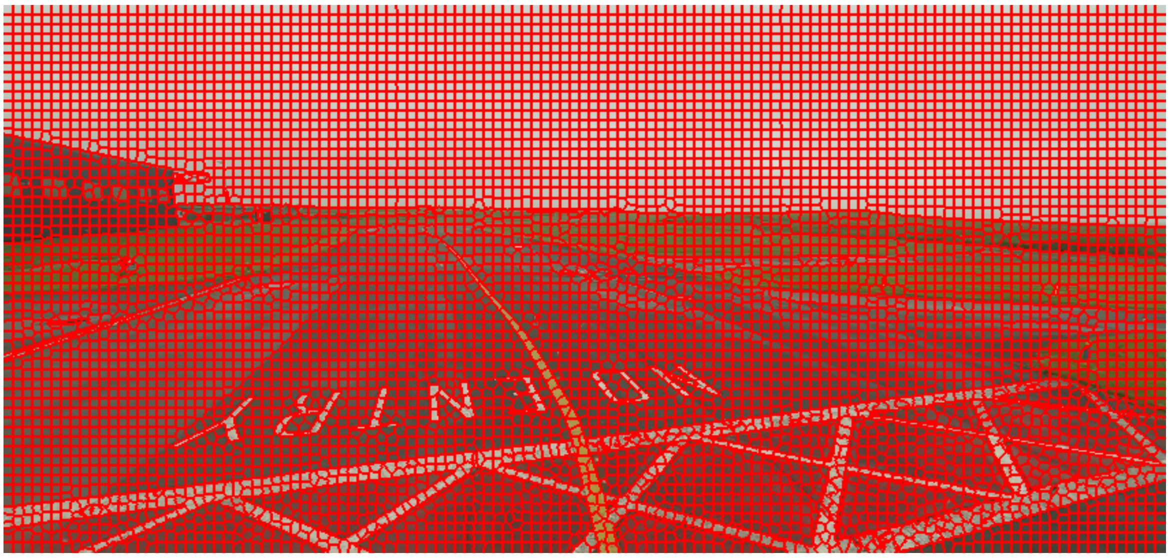
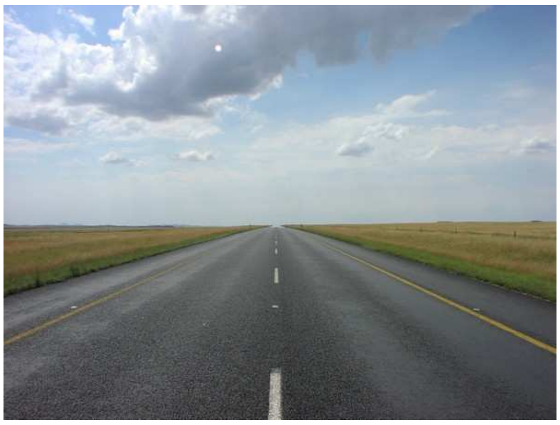


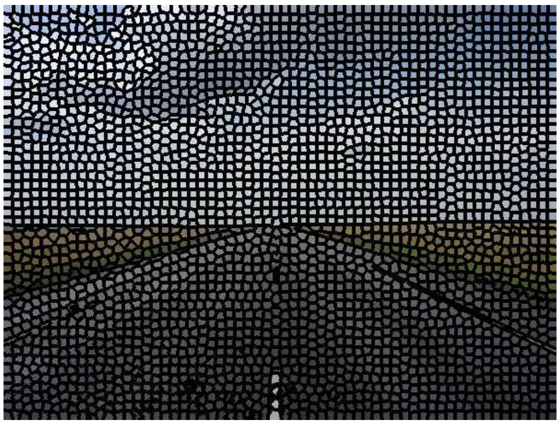
Fig. 5.13 Superpixel over-segmentation

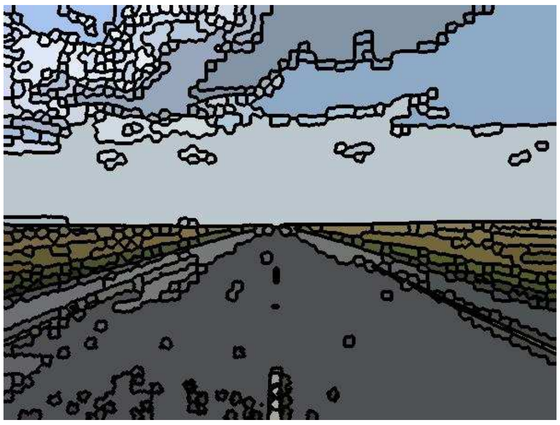
information to accurately perform classification. The problem of classification and over-segmentation is shown in [46], where segmentation is achieved using superpixels. Classification of each superpixel is then performed using texture data, but is also augmented with relative location data (e.g. pixels above trees are likely to be sky, and pixels below trees are likely to be grass). Despite most superpixels being classified correctly, the final result still suffers from misclassification, due to the small size of superpixels making it difficult to extract sufficient features.

For classification, larger clusters are preferable (where possible), as each cluster has more information for the classifier to use. Accordingly, a suitable intermediary stage is to use the over-segmented output from SLIC and apply reachability clustering; grouping the superpixels into larger, visually similar regions. As simply merging superpixels does not shift their borders, the final cluster boundaries should preserve their accuracy to the underlying image, whilst offering substantially more information to aid in classification. Therefore, this section outlines potential methods of re-clustering superpixels.

Figure 5.14a shows an aerodrome image with the SLIC superpixel over-segmentation shown in Figure 5.14b. Based on this over-segmentation, Figure 5.14c shows the results of clustering using k-nearest neighbors algorithm, whereas Figure 5.14d shows the results of reachability clustering, using the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm as outlined below.


(a) Original image of outdoor scene


(b) SLIC Superpixel over-segmentation result


(c) Nearest-neighbour clustering result

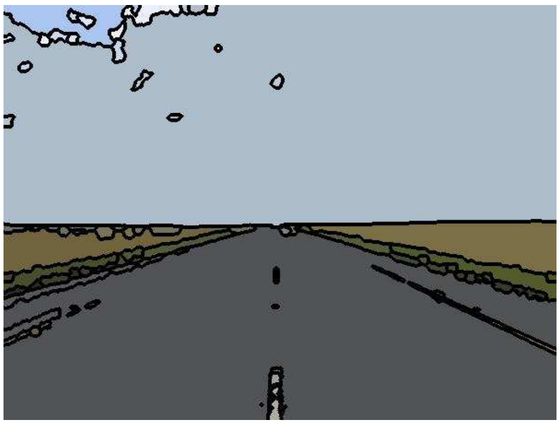

(d) Reachability clustering result

Fig. 5.14 Comparison of Reachability and Nearest neighbour clustering results

DBSCAN

Although the concept of using reachability to re-cluster superpixels has been previously explored, there is very little literature on the subject. The only prominent example is the work undertaken by Kovesi [63], in which SLIC generated superpixels are re-clustered to form larger regions, using the DBSCAN algorithm [35]. (This use of DBSCAN for re-clustering is different from a more common use of DBSCAN for superpixel generation, such as in [99]).

DBSCAN is an advanced clustering algorithm which is entirely based on the concept of reachability. For two points to be ‘reachable’ from each other, the distance between them must be less than a threshold distance, ϵ . DBSCAN builds upon the concept of reachability by introducing the ’density reachability’, in which reachable points are further categorised by the number of connections they can make. Using DBSCAN, each data point (or superpixel) is categorised as:

- **Core Point -** If a point is reachable from at least *minPts* other points, then it is considered a core point. Each cluster must have at least one core point, and the minimum number of points within a cluster is $minPts + 1$.
- **Reachable Point -** If a point is reachable from less than *minPts* but is within ϵ of a known core point, it is still considered to be part of the cluster. but cannot be used to link in further points (i.e. most cluster border points are not core). As the connection to other ‘reachable’ points is not explored, it prevents neighbouring regions with only a single link from becoming combined.
- **Outliers -** Any remaining points are considered outliers and are not clustered.

A reachability approach can cluster data which cannot be clustered using older techniques (such as k-means or Gaussian mixture clustering). As clusters are formed based on the spread of data, there is no need to specify the number of clusters beforehand. This allows an image containing an unknown number of objects to be reliably segmented. In addition, a reachability approach allows data to be clustered even when the sets cannot be separated linearly (which is of consequence for objects with gradient colour in different colour channels). This is particularly useful for outdoor images, as atmospheric scattering creates a gradual change in colour with distance. Provided that the superpixels between the closest and furthest point capture the change in colour gradually, the two extremes will be considered reachable and therefore will be combined into a single region.

Density reachability is also essential to stop over-clustering. Although superpixels are large enough that they should prevent 'path-forming', the possibility still exists that two similar superpixels could connect two otherwise disimilar regions. By requiring a minimum number of connection points, DBSCAN greatly increases the liklihood of filtering out such connections.

As the original superpixel boundaries formed by SLIC are preserved during DBSCAN clustering, the final result retains the sharp resolution required for depth estimation. If the final segmentation is still an over-segmentation, the larger clusters increase confidence in getting the correct classification results, with the remaining over-segmented regions finally combined by sharing the same class. A successful segmentation result has already been achieved using this methodology [33].

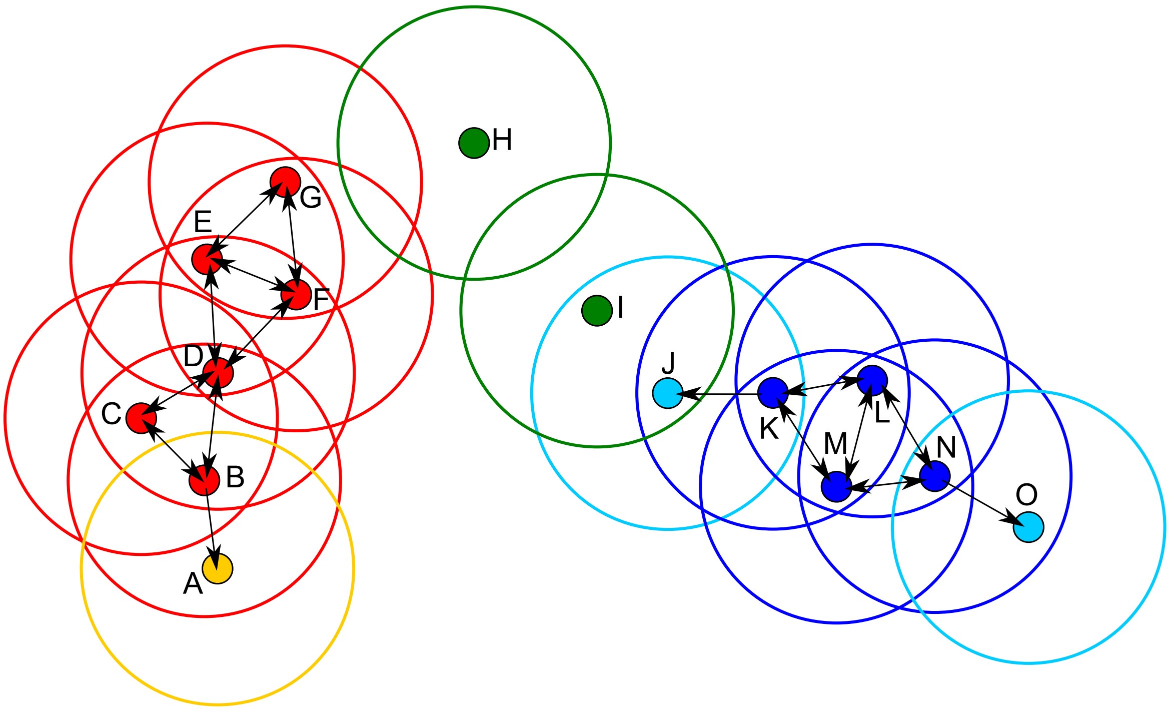


Fig. 5.15 Reachability based clustering, in which clusters are formed using a minPts value of two. Points A, J and O are connected to the cluster despite not being core points. In addition, due to point J not being a core point, point I is not included within the cluster despite point I being reachable from point J.

For superpixel re-clustering the most suitable distance metric to use is Delta-E (ΔE^*_{ab}), which is essentially the euclidean colour distance within the *Lab* colourspace. As with SLIC, colour data is used as the small size of an individual superpixel limits the amount of other data types which could be extracted for comparison. To improve computational speed and to prevent disconnected clusters forming, only the colour distance between adjacent superpixels have been considered.

Due to the contents of the image being unknown, the relationship between each set of neighbouring superpixels must be treated equally. Therefore, determining if two superpixels are 'within reach' of each other is entirely dependant on the single parameter ϵ , making the selection of an appropriate value of ϵ highly important. As the average change between neighbouring superpixels of the same class should be very small, the value required is around the 'Just Noticeable Distance', being large enough to include superpixels of the same class within a cluster but not form links with superpixels which are different.

As lighting conditions change throughout the day and the distance metric includes a brightness component, the colour distance between clusters will be effected by global illumination changes. However, to operate in outdoor conditions, cameras make use of techniques such as auto-exposure to adjust the values in the output image. Therefore, the brightness component of an image is typically scaled to the same range despite changes in the brightness of a scene. As this shifts with global illumination changes, the overall impact is very small during day time conditions.

This form of reachability will not be successful when the average brightness is very low, as the distance between superpixels of any class will tend to zero. Despite this, as the system is not currently intended to function at night, this is not considered to be an issue. Therefore, for this work ϵ was experimentally determined using aerodrome images with the intention that all testing for this work is only undertaken during day-time conditions.

One additional issue with DBSCAN is that it is not entirely deterministic. Superpixels which are equidistant from multiple clusters are placed into which ever cluster is processed first. This minor factor is unlikely to contribute much to the overall result. If a superpixel is equally likely to belong to two clusters, they are probably the same class and will be grouped together during the classification stage.

5.5 Summary

In Chapter 4, it was established that Image Segmentation is the most suitable approach for terrain classification. As many different methods are available, this chapter reviews potential methods of achieving the best segmentation and identifying why many established techniques are not suitable for this application.

As classifier-led-segmentation methods are now the most popular methods of image segmentation, such techniques would seem the most appropriate. However, the majority of classifier-led-segmentation methods use deep-learning and therefore are not viable for aerospace use. In addition, due to the remaining classifier-led-segmentation approaches tending to bias the results towards known classes and artificially shifting object borders, it was decided that an untrained segmentation approach was better suited for this work.

After reviewing several unsuitable methods of untrained segmentation, a comparison of their shortcomings established the ideal result of 'Object specific segmentation', in which each object in the image is individually segmented, regardless of scale or contents. In order to satisfy border accuracy between objects, a localised segmentation method was required, with a Superpixel implementation found to provide the best segmentation result, restricting cluster growth in both space and colour distance. As the superpixel-segmentation process divides the image into visually distinct clusters of pixels, each cluster should contain only a single object type (i.e. class), allowing the entirety of each cluster to be used during classification.

As the output from superpixel clustering is a dramatic over-segmentation, processing at this level is inefficient, whilst also limiting the amount of information available for classification. A reachability clustering approach has been found to be effective in grouping superpixels so as to form larger regions without disturbing localised borders, achieving the closest results to 'Object specific segmentation' using untrained segmentation. The next chapter (Chapter 6) builds upon the final approach selected in this chapter, and establishes the actual implementation used, in addition to some initial results.

Chapter 6

Implementation of Image Segmentation and Data Extraction

In Chapter 5, it was established that an untrained image segmentation approach was the most suitable. By clustering through similarity and reachability, the resulting clusters should each only contain a single type of object, albeit with the actual class still remaining an unknown. Therefore, to produce the desired semantic segmentation, each clusters must undergo classification.

The classification stage involves extracting features from each cluster and comparing them to known examples of each class. The best match and the degree of similarity then provides a confidence of the clusters identity. As the classification stage depends on the features extracted from each cluster, data extraction is a vital part of semantic segmentation. As such, it is discussed here within this individual chapter, separate to the actual process of classification. Prior to classification the implementation must consists of two elements:

- Using untrained segmentation to divide the image into regions, each of which should only contain a single class.
- Performing data extraction upon these regions to extract data suitable for classification.

This chapter describes the methods used to implement this approach. Figure 6.1 shows how data extraction is required for both training and classification, with stages discussed in this chapter highlighted in blue. Although classifier training and classification are different processes, in order for the classifier to function correctly, consistent data extraction techniques must be used throughout.

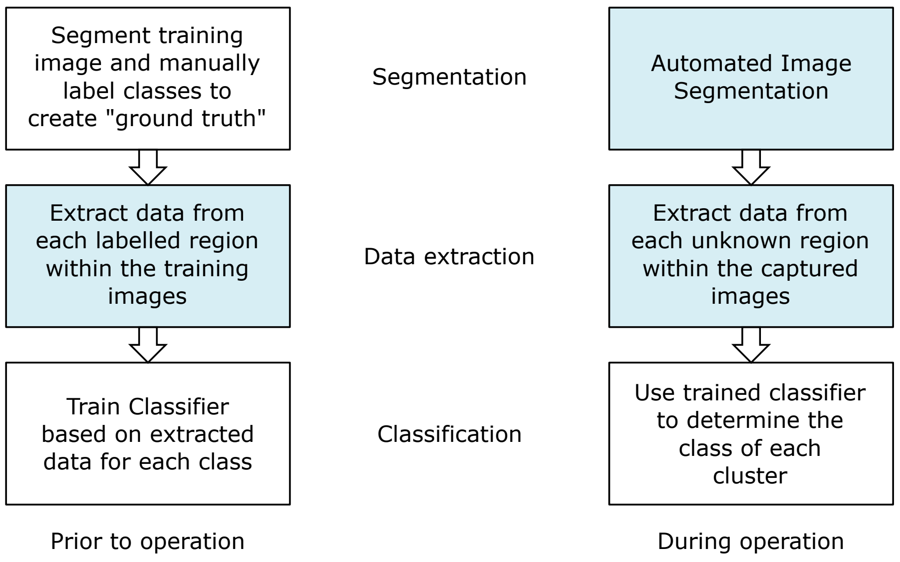

Segment training image and manually label classes to create "ground truth"
Segmentation
Automated Image Segmentation
Extract data from each labelled region within the training images
Data extraction
Extract data from each unknown region within the captured images
Train Classifier based on extracted data for each class
Classification
Use trained classifier to determine the class of each cluster
Prior to operation
During operation

Fig. 6.1 Stages of the semantic segmentation process, showing the 'repetition' of stages both *prior to* and *during* operation. To ensure prior data can be used for classification, the same data extraction methods must be used throughout.

Contributions from this chapter include:

- A novel method of graphical reachability, intended for use in combining superpixels into larger regions without sudden changes in colour (Section 6.1.2).
- The specification of Normalised Relative Luminance (NRL) and its relationship with distance for surface marking extraction (Section 6.4.3).

6.1 Untrained Image Segmentation

With the aim of achieving 'Object Specific Segmentation', as defined in Section 5.4 , two components are required: superpixel oversegmentation and the subsequent reachability re-clustering. As described in Section 5.4.1, the current state-of-the-art in superpixel generation is widely considered to be Simple Linear Iterative Clustering (SLIC). As such, this will form the basis of the superpixel approach.

6.1.1 SLIC Superpixels

SLIC is very simple to implement, with the only tunable variable being the number of desired superpixels. As such, the segmentation was implemented using the method suggested by the original authors Achanta et al. [4]. (This method was also used in the previously published work [33]).

The SLIC algorithm is fundamentally a K-means clustering approach with two inputs; the dataset (i.e. the input image of size N pixels) and the number of desired centres K (i.e., the number of superpixels). Alternatively, the approximate size of each superpixel can be used as it is simply N/K.

For K-means, cluster centres are required. This are spatial located within the XY coordinates of the image, at every grid interval $S = \sqrt{N/K}$. Clustering is then achieved using a distance metric composed of both colour and spatial distance. In order for colour distance to be applied uniformly, the CIELAB colour space is used. Despite the colour difference being revised multiple times since it's inception, Achanta et al. use the original CIE76 definition of colour distance and achieve good results, so this has been replicated here.

Each cluster centre is denoted as a 5 dimensional co-ordinate in terms of colour and location.

$$C_k = [l_k, a_k, b_k, x_k, y_k]^T$$

The colour difference formula is simply the euclidean distance in CIELAB colour space, for each other pixel compares to the cluster centre:

$$d_{lab} = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2}$$

Similarly, the spatial distance is also the euclidean distance from each pixel to the cluster centre, only within the XY image space:

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}$$

As colour distance and spatial distance are not directly comparable, they need to be normalised to allow comparison. This is done using the maximum colour and spatial distances within each cluster (N_{lab} and N_{xy}):

$$D_s = \sqrt{(\frac{d_{lab}}{N_{Lab}})^2 + (\frac{d_{xy}}{N_{xy}})^2}$$

As the clusters are formed iteratively, constantly determining these maximum values would greatly increase the computational intensity. Instead, simplifications can be made through approximation. As the maximum spatial distance in any one cluster should not exceed the distance between cluster centres, N_{xy} can be approximated as S. Colour distance is more complex as they are specific to both individual clusters and vary massively depending on the image. Instead, Achanta et al. simply replace the maximum colour distance N_{Lab} with a constant 'gain': m.

By tuning this 'gain', the relative contribution of spatial and colour distance can be adjusted in the overall distance metric. A large value of m results in spatially uniform superpixels, which may not adhere to image boundaries perfectly, while a very low value of m results in superpixels which accurately follow borders but are irregular sizes and shapes. When using CIELAB, Achanta et al. suggest that m remains within the range of 1 to 40. The overall distance metric used is therefore:

$$D_s = d_{lab} + \frac{m}{s} d_{xy}$$

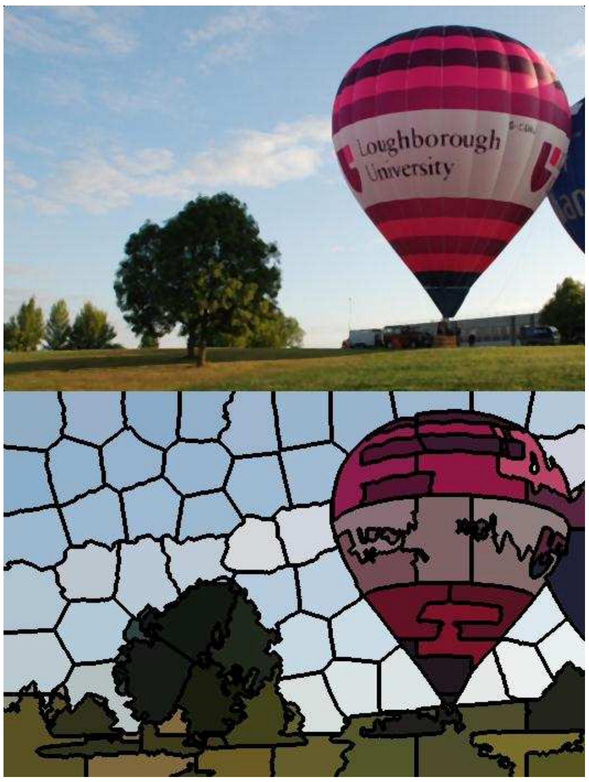


Fig. 6.2 SLIC Representation of a hot air balloon, using very large superpixels.

An iterative algorithmic approach is then used to segment the image into superpixels. Another example of SLIC superpixel segmentation is shown in Figure 6.2. Once the SLIC algorithm has been applied, the original image will have been divided into multiple small clusters, each similar in size. As superpixels are spatial constrained, the end result is always a significant over-segmentation, ensuring that the superpixels are a suitable size for reachability clustering. In addition, using both colour and spatial distance to cluster ensures that even subtle edges in the original image are reflected in the position of the superpixel borders. This minimises the risk that any superpixel will contain more than one category of object, whilst also ensuring that superpixel boundaries conform to the boundaries of objects within the image.

6.1.2 Graph Based Reachability Reclustering

Despite the success of DBSCAN in previous work ([33]), DBSCAN has not been used to re-cluster superpixels here. The foremost reason for this is computational speed. When used to cluster unrelated points within a dataset, DBSCAN is highly efficient. However, it does not include the concept of adjacency. In order to prevent disconnected clusters from forming in the 2D image, superpixel adjacency was calculated in a prior step and stored within an adjacency matrix. For each new superpixel that could be added to a cluster, both an adjacency and colour distance check were required.

As the adjacency matrix was available, a novel graph based implementation was found to be far quicker, similar to the Felzenszwalb graph based method for creating superpixels ([39]). Using the same concepts as in DBSCAN (density reachability,(ΔE^*_{ab}) distance metric and threshold value ϵ , a Region Adjacency Graph (RAG) was formed in which each superpixel was considered a node. Edges between nodes were formed based on adjacency, with colour distance between superpixels used as the edge weighting. The graph based implementation of DBSCAN was simply:

- Removing all edges above the threshold ϵ .
- Identifying core 'nodes' which have more than *minPts* connections.
- Remove all edges which are not connected to a core.

Defining this in graph theory notation, let graph $G = (V, E)$ where:

V (or $V(G)$) are the vertices which represent each superpixel and
E (or $E(G)$) are the edges created from the adjacency matrix.

Determine the vertices which represent colour distance greater than the threshold ϵ:

$$V_{EX} = V(G) > \epsilon$$

Remove these edges to form a reduced graph, often already divided into several disconnected sub-graphs:

$$G_1 = G - V_{EX}$$

Core points are identified as vertices where the maximum degree of a vertex (number of connected edges $\Delta(G_1)$) is greater than *minPts*:

$$V_{CORE} = \Delta(G_1) \geq minPts$$

Finally, remove edges not connected to a core. This is most easily defined by establishing the neighbourhood (set of adjacent vertices) of each core point and combining them into a single graph.

$$G_{REACHABILITY} = \sum N(V_{CORE})$$

As the neighbourhood of core vertices will included connected vertices which aren't core themselves, this produces the appropriate re-clustering far quicker than the process of visiting each point in turn.

Returning to the previous implementation where DBSCAN and adjacency were considered separately, DBSCAN simply established colour similarity between a superpixel and all the superpixels already within a cluster. If a new superpixel was similar to enough superpixels already within the cluster, it was included even if those superpixels were not directly adjacent. An additional benefit of this new graph based method is that it requires that both similarity and adjacency to be satisfied before a superpixel can be added to a cluster, ensuring both local superpixel similarity as well as similarity to the cluster as a whole.

To further to minimise the risk of multiple classes being captured by a single cluster, the graph is purposely divided using a threshold which produces a higher number of clusters than optimal, in order to ensure clusters do not expand into neighbouring

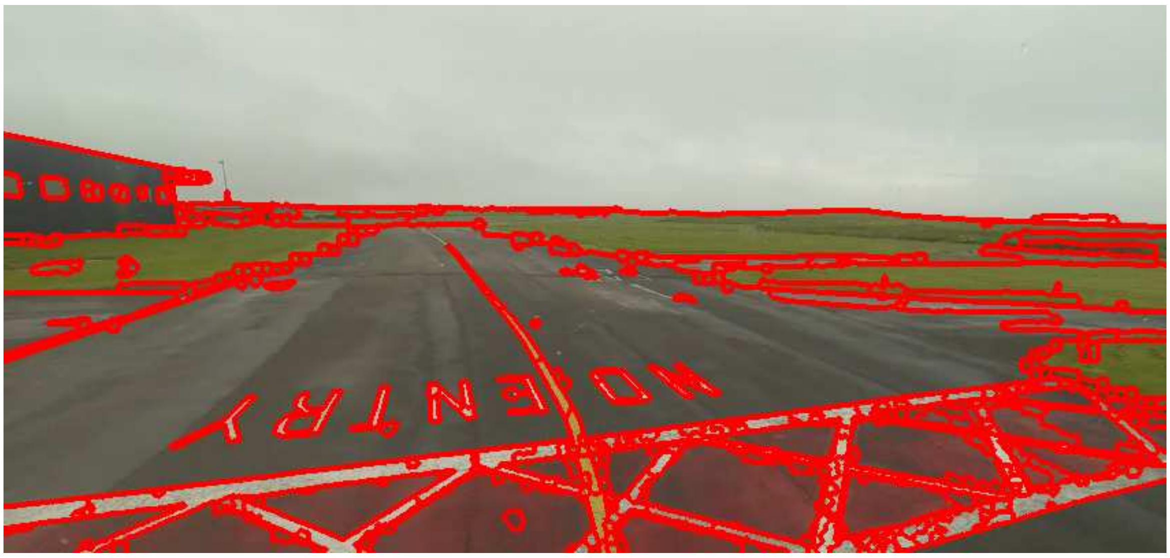


Fig. 6.3 Superpixel based reachability clustering

regions. Figure 6.3 shows the results of re-clustering superpixels, which is identical regardless of whether DBSCAN or graph based re-clustering is used.

6.1.3 Results

The chosen reachability clustering approach has been found to produce better results when compared to the other techniques which have been previously discussed. Despite this, determining the actual quality of a segmentation result is difficult.

As shown in Figure 6.3, the chosen segmentation approach would appear to successfully divide an image into clusters, each of which contain only a single class of object. Although this is in accordance with the desired result, simply inspecting the image does not provide an indication of the quality of the segmentation process. For simple images consisting of solid blocks of colour, it would be straightforward to assess whether the segmentation process had produced the optimal result. However, when dealing with complex scenes (containing objects which themselves have sub-components of different textures and colours) assessing such a result is extremely difficult.

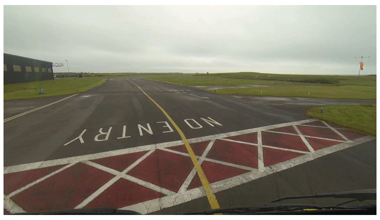
Work by Mohammad et al. [75] attempts to provide a numerical method for evaluating the quality of a segmentation, based on the similarity of the final result to a manually defined 'ground truth'. By comparing the automated result to that produced by a human, the difference between expectancy and actual result can be used to assess quality. The downside to this approach is that it requires the manual ground truth to match the intentions of the segmentation process. A data set for comparison is offered by Mohammad et al. with which segmentation processes can be assessed. However, the dataset assumes that the segmentation process is attempting to achieve object-specific separation. For this work, a deliberate over-segmentation has been retained to prevent any risk of under segmentation. Therefore, simply comparing to full objects in scenes is likely to indicate poor performance if the segmentation result does not match, even if it is viable for the classification stage.

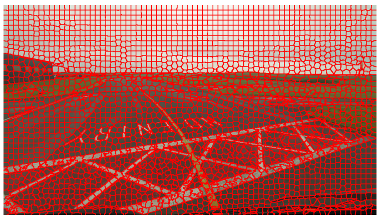
Instead, the quality of the outcome should be assessed on the requirements of the segmentation. As the goal is to pass the results to a classifier, under-segmentation would likely result in misclassification. Attempting to achieve universal segmentation using a single technique requires the approach to be cautious, in order to prevent under segmentation. Therefore the primary assessment of segmentation quality should be to determine if any instances of under-segmentation have occurred. Nevertheless, the purpose of the re-clustering phase is to allow large regions to form to provide sufficient data for classification. As retaining an over-segmentation reduces the amount of data available to classify each cluster, this should also be avoided. Therefore, assessing the segmentation involves:

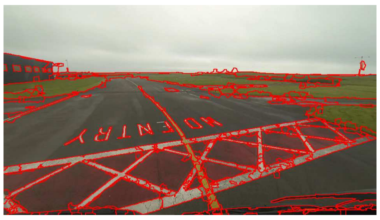
- Determining the presence of any under-segmentation within the final image result.
- Determine if the amount of over-segmentation is acceptable.

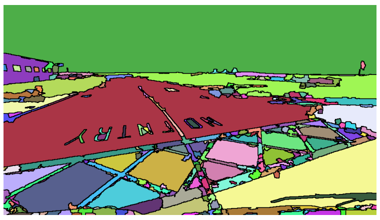
Following on from the work of Mohammad et al, nine images were manually segmented to provide a ground truth, against which the automated results were compared. The following figures show the final segmentation result applied to each of the aerodrome images. The top row, from left to right, show the original image, superpixel over segmentation and the final cluster outlines. The bottom row, again from left to right, shows a false colour representation of the automated segmentation result, false colour manual segmentation (ground truth), and finally regions of 'undersegmentation' in the final segmentation result, highlighted in red.

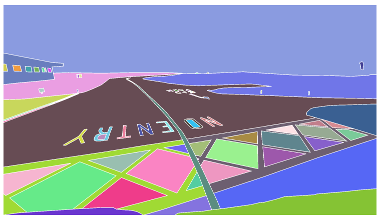
Of note, these images were provided for this work by BAe Systems. During capture, the camera was mounted in a position such that part of the vehicle test-bed was visible in the bottom of the image. For the sake of depth extraction, the characteristics of the camera must be retained - i.e. the image has not been cropped. Instead, a Region Of Interest (ROI) will be used to remove these sections during the classification stage. Therefore, the results of segmentation within this area are not reviewed as they are not relevant to the task.


(a) Original Image


(b) Superpixel Segmentation


(c) Reachability Re-clustering


(d) Re-clustering with Artificial Colouration


(e) Manual segmentation with Artificial Colouration

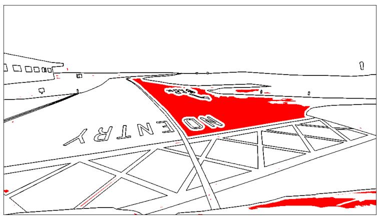

(f) Difference between manual and automated segmentation

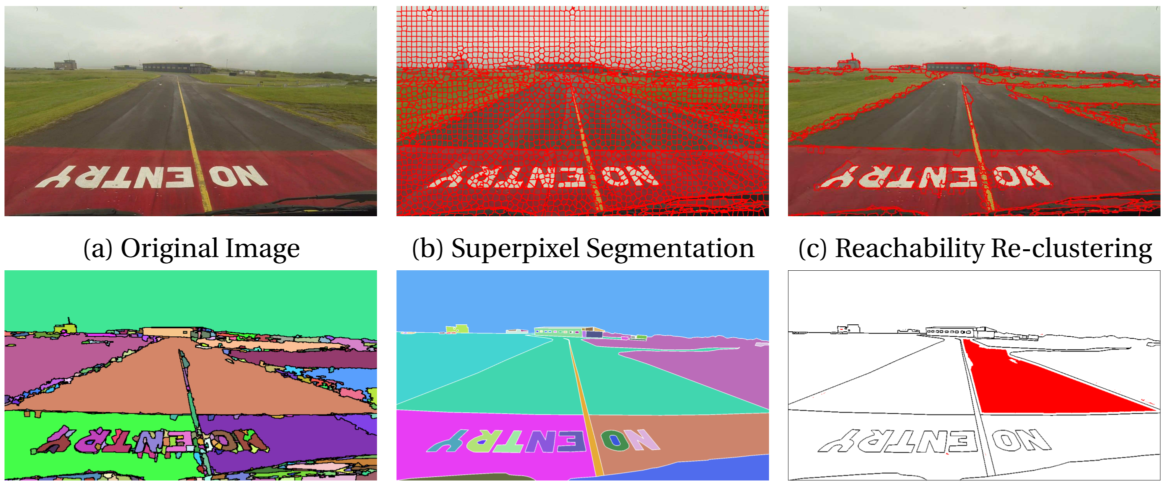
Fig. 6.4 Results of segmentation applied to aerodrome image 1

Figure 6.4 shows the segmentation approach applied to an image with a very busy foreground. From Figures 6.4c and 6.4d it is clear that the end result is slightly over-segmented, primarily on the surface markings in the foreground. Looking at the image, with the worn paint work and inconsistent colouring, this would appear to be an accurate reflection of the actual taxiway conditions and therefore would be an appropriate segmentation output.

Comparing Figures 6.4d and 6.4e produces Figure 6.4f, where elements which are undersegmented in Figure 6.4d are highlighted in red.

Although a large proportion of Fig 6.4f is in red, the cause is simply that the manual segmentation approach maintains a consistent taxiway centreline to the horizon, splitting the taxiway into two clusters. By comparison, the automated segmentation approach fails to separate the taxiway in the distance, with the taxiway centreline becoming too small and indistinct to produce superpixels, as shown in Figure 6.4b. As such, although the under-segmentation may appear extreme, in actuality the two areas both represent unpainted asphalt surface.

The weak detection of the distant taxiway lines is an under-segmentation, however it's importance (considering the taxiway lines are around 50 metres from the camera before becoming indistinct) is minimal. Therefore, this image can be assessed to be suitably segmented.



(a) Original Image (b) Superpixel Segmentation (c) Reachability Re-clustering

(d) Re-clustering with Artificial Colouration (e) Manual segmentation with Artificial Colouration (f) Difference between manual and automated segmentation

Fig. 6.5 Results of segmentation applied to aerodrome image 2

Figure 6.5a shows a very similar scene as Figure 6.4a, with a busy foreground and a taxiway divided by a centreline. From Figure 6.5d, it can be seen that the results of the automated segmentation are also very similar to the results for the previous figure. In the exact same way as Figure 6.4, the manual centreline extends to the horizon, splitting the taxiway into two clusters. Again, as the centreline extends into the distance, superpixel resolution is not small enough to isolate the centreline, leading to both sides of the taxiway merging and being considered a single cluster. This produces a near identical ‘under-segmentation’ as occurred in Figure 6.4. As the nearby centrelines are well defined, the imprecise distant lines should be of limited consequence for the classification stage. As with the previous image, Figure 6.5d shows that an over-segmentation occurs around the multi-coloured floor markings in the foreground, sometimes as small as single superpixels. This is primarily due to compression artefacts in the image, resulting in a very small area of distortion around the lettering. Although the same distortions were present in Figure 6.5, the worn markings were responsible for the majority of the over-segmentation. By comparison, in this figure the compression artefacts are the main cause. Compression artefacts such as these occur when two neighbouring regions of intensively different colour meet in a sharp edge. This issue is very hard to resolve within the segmentation stage, as the image storage format has altered pixel values on each side of the cluster borders. However, as the average pixel values remain mostly unchanged, it is likely that this will be resolved during the classification stage. When deployed in an actual system, a lossless image format would be more appropriate for this task.

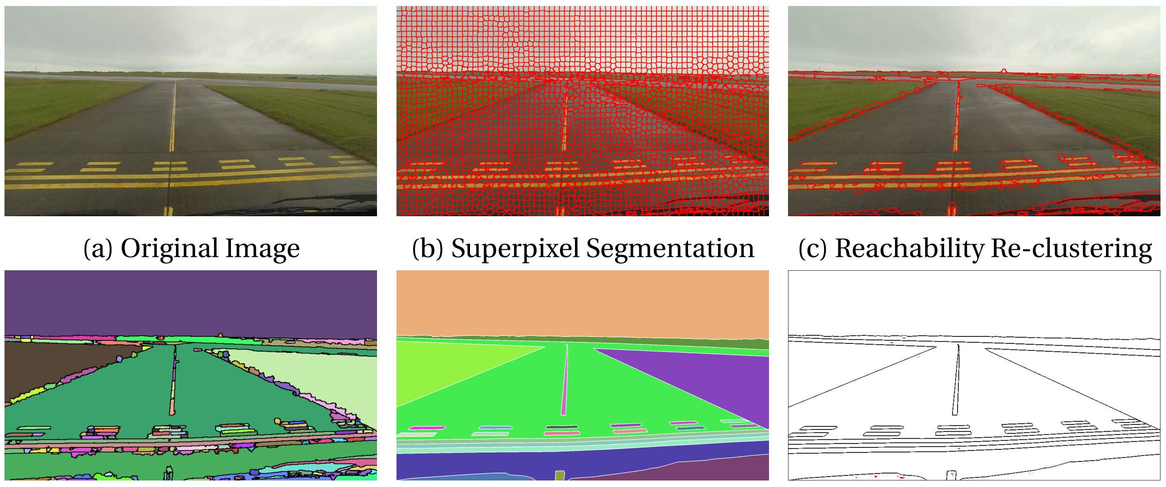

(a) Original Image (b) Superpixel Segmentation (c) Reachability Re-clustering (d) Re-clustering with Artificial Colouration (e) Manual segmentation with Artificial Colouration (f) Difference between manual and automated segmentation

Fig. 6.6 Results of segmentation applied to aerodrome image 3

Unlike the previous two aerodromes images, Figure 6.6 shows no major under-segmentation. This is due to the taxiway centreline having a clear starting point in the Figure 6.6a leading to both sides of the taxiway being connected in both the automated and manual result. Again, over-segmentation occurs, especially around the stop bar markings in the lower part of the image. This is again caused by compression artefacts. Although the exact compression method depends on the format in which the video was encoded, the storage size of the image is typically reduced by grouping pixels together which share a similar colour and storing all their data

Essentially, to reduce file size image compression is already somewhat similar to superpixels in that small regions of the image are 'described' as sharing the same hue. This is most clearly seen when the image is displayed using Hue-Saturation-Value (HSV) colourspace, as shown in Figure 6.7. Despite the taxiway outside of the yellow markings being grey, this visible colour difference is a result of variance in the 'Value' channel alone. From the 'Hue' channel, the entire taxiway is considered yellow and the sky is green. Only the influence of the other two channels allows the differences to be seen. Despite the 'Saturation' channel being far more precise than hue, the data is still averaged over a small range of pixels, resulting in shared pixel values extending across clear borders within the original image. This results in superpixels close to object borders having slightly different average values than those at object centres. Again, this can be resolved by using cameras which do not compress the image. However, for this work, the classification stage should resolve most of the over-segmentation issues.

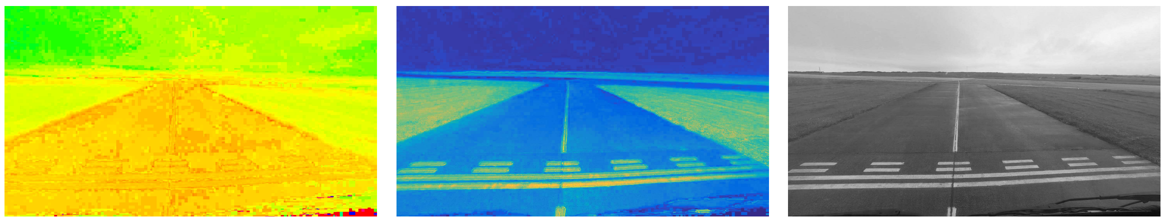

(a) Hue (b) Saturation (c) Value

Fig. 6.7 Image compression artefacts in the Hue, Saturation and Value channels

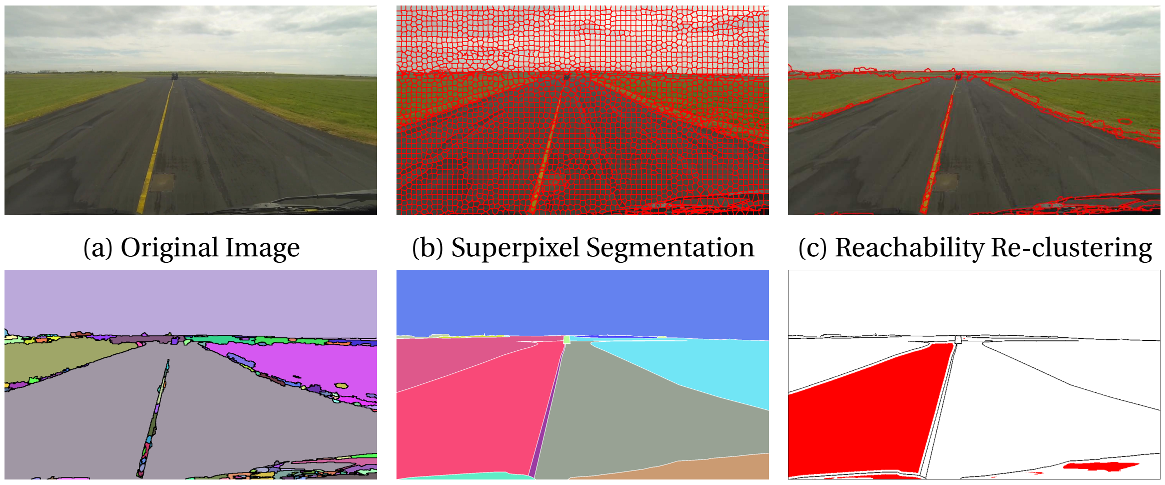
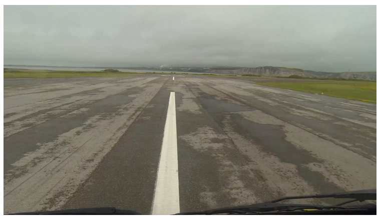
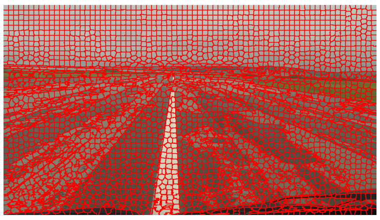

(a) Original Image (b) Superpixel Segmentation (c) Reachability Re-clustering (d) Re-clustering with Artificial Colouration (e) Manual segmentation with Artificial Colouration (f) Difference between manual and automated segmentation

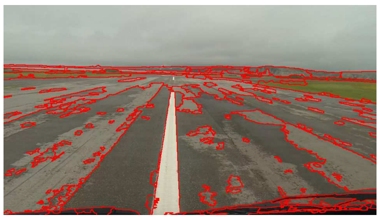
Fig. 6.8 Results of segmentation applied to aerodrome image 4

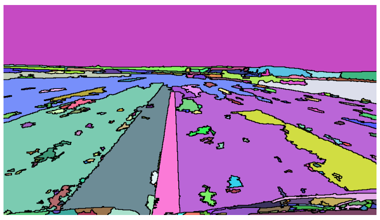
Figure 6.8 shows very empty scene, with a similar result to the first two aerodrome images (Figures 6.4 and 6.5) in which the taxiway is considered under-segmented due to the same difficulty in discerning the taxiway centreline at distance.

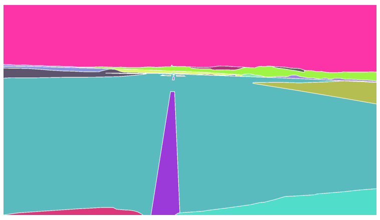
One noticeable success in Figure 6.8 is the accurate object-specific segmentation of the vehicle (a Land Rover discovery) on the taxiway ahead of the camera. Despite similar ‘Saturation’ and ‘Value’ to the taxiway itself, the vehicle has still been individually segmented. It is the re-clustering that allows this difference to be captured which also introduces the over-segmentation closer to the camera. As detecting the presence of this collision risk is essential, the segmentation approach can be considered appropriate, provided that the classification stage can correctly identify the vehicle as a non-terrain feature.


(a) Original Image


(b) Superpixel Segmentation


(c) Reachability Re-clustering


(d) Re-clustering with Artificial Colouration


(e) Manual segmentation with Artificial Colouration

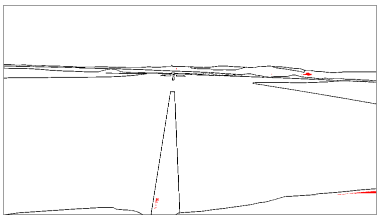
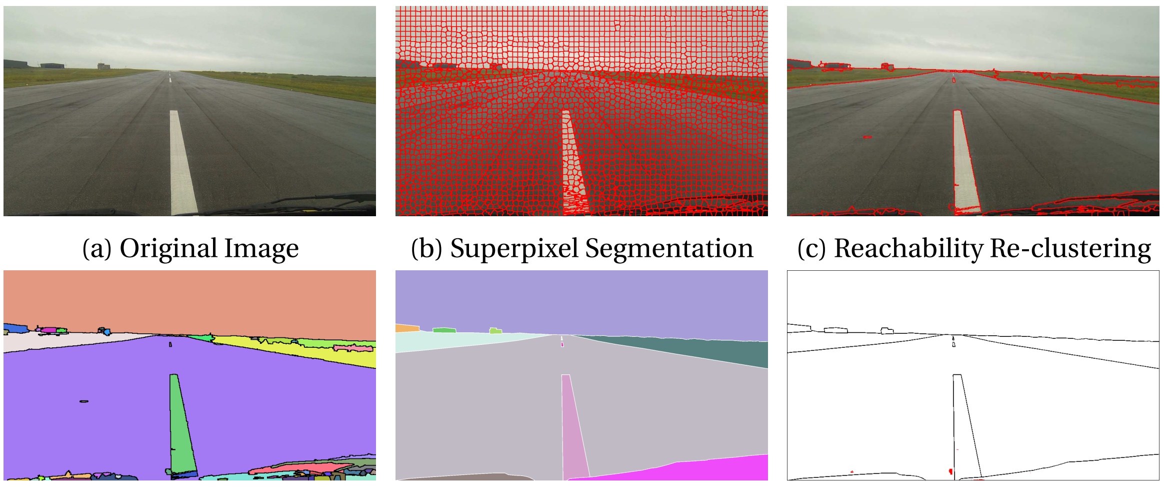

(f) Difference between manual and automated segmentation

Fig. 6.9 Results of segmentation applied to aerodrome image 5

Figure 6.9 again shows no under-segmentation, in part due to the camera being positioned on a runway with no taxiway centre line to divide the asphalt in half. However, compared to the manually segmented image the automated result has produced a dramatic over-segmentation, as seen in Figure 6.9d.

The cause of this over-segmentation is clear from Figure 6.4a, where the runway material can be seen to be highly worn. Although a human would correctly interpret the runway as a single object, the difference in material colour has produced the highly segmented result. However, as each segment still represents asphalt, this over-segmentation should be resolved through classification.

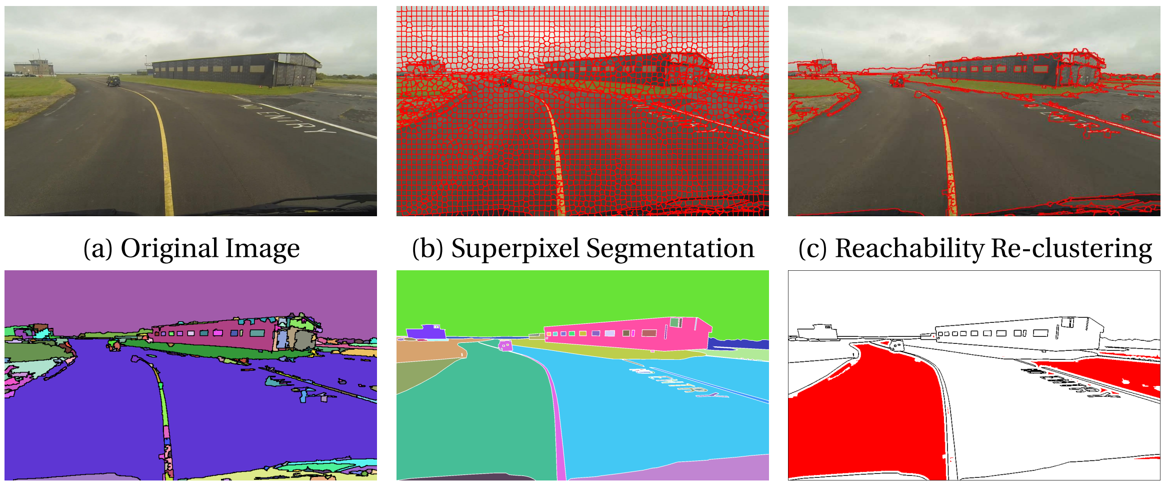


(a) Original Image (b) Superpixel Segmentation (c) Reachability Re-clustering

(d) Re-clustering with Artificial Colouration (e) Manual segmentation with Artificial Colouration (f) Difference between manual and automated segmentation

Fig. 6.10 Results of segmentation applied to aerodrome image 6

Figure 6.10 shows an image taken from the camera on a runway with consistent surface materials. As such, the final automated result is extremely similar to the manual segmentation, with neither significant over-segmentation or under-segmentation. The most significant different is the seperation of the greenery on the right side of the image into two clusters, whereas the manual approach only identifies one. As this actually reflects the actual state of this area, with nearby grass and bushes intermixed with distance fields, the automated approach could be considered more accurate than the manual segmentation result, which simply segmented based on colour.



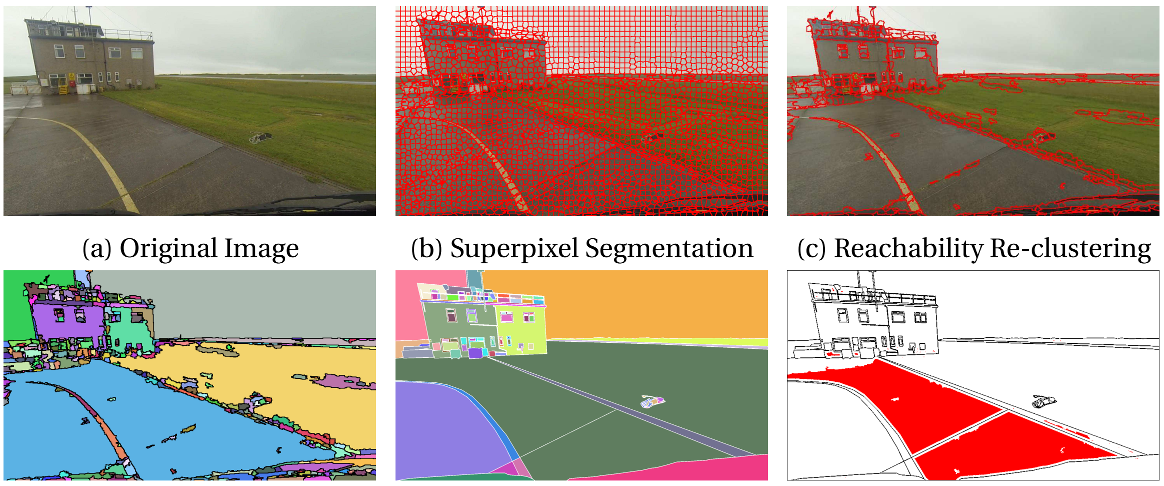
(a) Original Image (b) Superpixel Segmentation (c) Reachability Re-clustering

(d) Re-clustering with Artificial Colouration (e) Manual segmentation with Artificial Colouration (f) Difference between manual and automated segmentation

Fig. 6.11 Results of segmentation applied to aerodrome image 7

Figure 6.11 shows a more complex scene, with buildings, vehicles and surface markings. Again, under-segmentation has only occurred on the ground, due to weak surface markings not segmenting the asphalt in the same manner that a human would expect.

Over-segmentation primarily occurs on the buildings and vehicles within the image. However, this accurate reflects the subcomponents these objects are formed from, and therefore can sill be considered an appropriate segmentation.

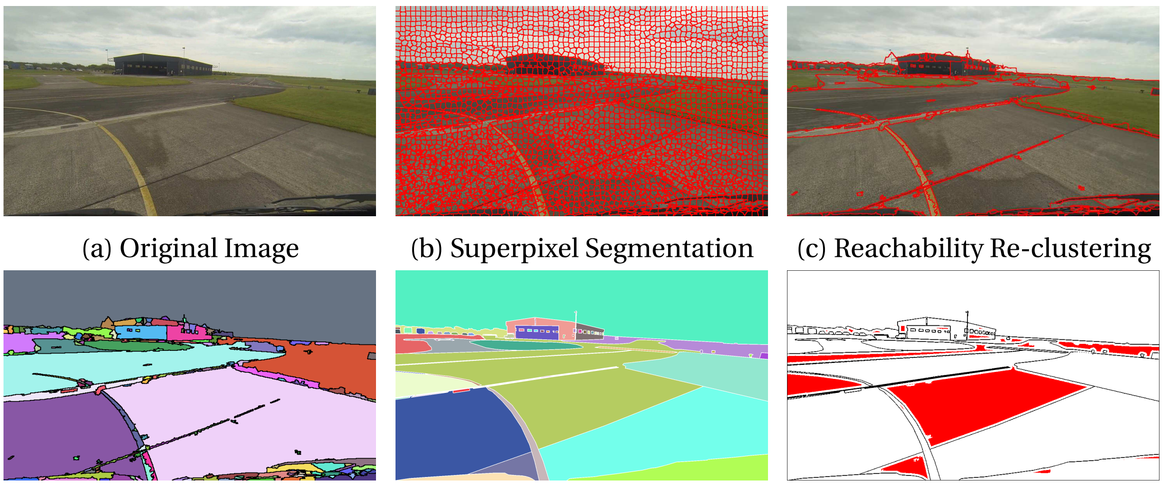


(a) Original Image (b) Superpixel Segmentation (c) Reachability Re-clustering

(d) Re-clustering with Artificial Colouration (e) Manual segmentation with Artificial Colouration (f) Difference between manual and automated segmentation

Fig. 6.12 Results of segmentation applied to aerodrome image 8

Figure 6.12 shows the camera much closer to a building. Unlike previous images, the surface material is concrete blocks, rather than asphalt. Although the taxiway centre-line is again responsible for an under-segmentation, an additional under-segmentation is produced by the manual segmentation adhering to the edges of the concrete blocks which make up the taxiway. Due to the borders of each block being very narrow, the line itself does little to alter the values of the superpixels on either side. As such, the re-clustering effort combines the same material from different surface slabs, despite the visible edge. As this has no effect on the classification stage, this is seen as an appropriate result. Even if the edge were to create two separate regions, this separation would be lost during classification stage as each cluster would share the same class type.



(a) Original Image (b) Superpixel Segmentation (c) Reachability Re-clustering

(d) Re-clustering with Artificial Colouration (e) Manual segmentation with Artificial Colouration (f) Difference between manual and automated segmentation

Fig. 6.13 Results of segmentation applied to aerodrome image 9

Figure 6.13 shows many regions indicated to be under-segmented, due to the presence of lines in the manual segmentation which are not well preserved in the automated result. However, Figure 6.13d and 6.13e also show an example of the opposite, in which the manual result combined regions of taxiway which were considered separate by the automated approach.

Although this could simply be percieved as an over-segmentation, it is in fact simply an example of two potential segmentation results, of which neither is more accurate. As all surface materials are the same throughout, both the manual and automated clustering results can be considered accurate, despite their difference in grouping.

6.1.4 Conclusion

In Chapter 5 it was established that the ideal result is 'object specific segmentation', in which an image is divided into visually distinct clusters of pixels, with each cluster containing only a single object type (i.e. class), allowing the entirety of each cluster to be used during classification. To achieve this, the chosen approach was determined to require two stages; a superpixel-segmentation process which creates an initial over segmentation (which adheres well to object borders), and a reachability re-clustering which groups similar superpixels into larger regions.

A method of achieving this result was already used in work prior to this thesis [33], where SLIC superpixels were combined with the DBSCAN algorithm for reachability reclustering. This chapter has built upon that work, replacing the previous DBSCAN algorithm with a novel solution, consisting of a computationally more efficient graph-based process. In addition to being more efficient, reachability between superpixels is restricted to where both adjacency and similarity are locally present, reducing the risk under-segmentation.

Testing is performed using images captured under realistic aerodrome conditions. The results can be empirically judged as a suitable segmentation. Although object specific segmentation is not always achieved, this is difficult in environments where objects can have varying colours per object. Over-segmentation does occur (sometimes due to the presence of artefacts caused by image compression), however this is not considered a failure as this can be resolved during the classification stage. In addition, although the alternative error of under-segmentation was demonstrated to occur (in comparison to manually segmented images) in each case the clusters combined were of the same class, with only weak taxiway markings failing to separate two asphalt regions. This form of under-segmentation would have minimal effect on the subsequent classification stage.

By establishing that untrained segmentation will be used, the image-processing tasks required to generate a semantic segmentation can now be defined. Despite the segmentation stage preparing clusters for classification, the intermediate stage of 'Data Extraction' is still required, in which relevant data pertaining to the contents of each cluster must be extracted from the image.

6.2 Data Selection, Extraction and Interpretation

Data extraction consists of the following elements:

- **Selection:** Choosing data that will allow the classifier to differentiate classes

- **Extraction:** How to process raw image data to extract this data type
- **Interpretation:** How much additional processing occurs prior to classification

When an image is stored in a digital format, it is converted into a three dimensional matrix of pixel values; typically consisting of several thousand rows and columns, as well as the three 'layers' which form the colour channels. For example, an image matrix for a high definition image (1920 by 1080 pixels in size) will consist of 6,220,800 individual elements. Despite each image consisting of a vast amount of data, each individual value means very little alone, and it is usually impractical to extract information from pixel values directly. Machine learning approaches methods, such as ANN can automatically determine the rules required to interpret the scene, allow raw pixel data to be used. However, in order to achieve the same result without deep-learning, human interpretation will need to be manually coded into the system.

Therefore, data extraction techniques are required to produce meaningful information. As many different methods exist, extraction techniques must be selected based on their suitability for the task. For example, attempting to differentiate red and green apples using object shape would not be the most appropriate strategy. In this way, selecting the appropriate data extraction methods is just as important as accurately extracting the data.

Although data extraction methods simplify the information compared to the raw pixel values, the resulting data is often still very complex. Using colour as an example, the average data per cluster could be directly provided to the classifier. Assuming standard 8-bit colour storage, this would provide 16,581,375 potential discrete colour inputs which the classifier would need to interpret. If this data were instead interpreted in a prior stage (for example, 'binning' raw colour in categories such as "red", "yellow", "black" or "grey") the smaller number of potential colours would make it easier to establish a relationship between classifier data input and class.

Preparing colour data for classification is straightforward; as colour data is continuous, colour-space can be divided into sections based on manually defined limits. As each colour section (e.g. red, black, green) is only indicative of class, multiple classes can share the same same colour category, with the overall classifier using multiple types of data to determine the actual class.

By comparison, other data can only be used to directly estimate a class. For example, highly complex data (such as image texture) typically correlates with only the specific class which it represents. This presents two options; either expose the lowest level of data possible to the final classifier (which is likely to be far more complex than colour data) or implement a separate classifier within the interpretation stage, providing information for the overall classifier to act upon.

Using multiple classification stages is computationally more intensive, but allows a simpler final classifier if estimates of class have already been made. Rather than a discrete class, the probability of being a class can be passed to an overall classifier. This concept is further explored in Chapter 7.

6.2.1 Data Loss through Image Digitisation

Prior to investigating methods of extracting data, the issues of data loss through image digitisation should be discussed. When a digital image is captured, the limitations of the camera hardware reduce the available data within the image; i.e. the quality of the camera sensor (Charge-Coupled Device (CCD)) determines the image resolution and colour range which can be captured. In addition, further information is lost due to the storage format imposing discretisation, data compression and range reduction in the chosen colour channels.

Discretisation and resolution limitations are unavoidable when using digital images. However, the severity of data compression and range reduction depends heavily on the colour space chosen. For example, as early colour televisions were Cathode Ray Tube (CRT) based, the image on the screen was created using separate Red-Green-Blue (RGB) colour channels. However, RGB images are data intensive and difficult to compress, (as brightness, saturation and chroma information is distributed throughout all three colour channels). Instead, the YCbCr colourspace was created as the standard for television broadcast, where Y is the brightness and Cb and Cr are co-ordinates within two chroma axes. As human eyes detect changes in the brightness more precisely than chroma, the chroma data can be sent a massively reduced rate, without much reduction in final image quality, known as chroma sub-sampling.

Despite CRT televisions having been replaced with more modern alternatives, newer implementations of YCbCr continue to be used, with chroma sub-sampling still present in most modern video systems. The data used for this work has clearly undergone chroma sub-sampling, in addition to the effects of modern compression strategies, such as inter-picture prediction. (in which only a small number of image frames within a video stream are complete, with the other frames simply updating the pixels which change when necessary).

Cameras which function without this loss of information are available, however they are typically used for industrial machine vision applications in controlled lighting environments. In addition, returning to Chapter 3, a motivation of this work is to make use of equipment already available onboard UAS. As the causes of data loss stated above are likely present in any image originally designed for transmission to the remote pilot, the ATS must be able to function with this type of image.

Despite the chroma components suffering around 75% data loss, the brightness component should remain fairly intact. The superpixel segmentation process has already been shown to provide a suitable output while using both chroma and brightness components of colour. This suggests that brightness data is still pixel accurate. As such, the brightness component should be used as the source of as much information as possible.

6.2.2 Data extraction over multiple frames

Prior to data extraction, it is useful to consider the fact that the system is not limited to dealing with individual frames in isolation. As the ATS is expected to be constantly classifying each captured frame, data from multiple image frames could be used to generate the classification result. Potential methods include using previous frame data as a prior in classifier, or splitting the image into persistent regions, with the new data from each frame used to modify or update the contents.

As the aircraft moves fairly slowly on the ground, the difference between subsequent frames captured by the camera should be minimal. Due to reachability clustering, regions awaiting classification are relatively large, increasingly the likelihood that a region in the previous frame can be matched in the next, purely on shape and location. If the boundaries have only shifted slightly, the contents of the image are highly likely to be the same.

Despite moving slowly, the number of comparable frames is very small. When the aircraft is turning, finding communicability with the image space is unlikely. Therefore, the ability to work on a single frame is still required. Furthermore, simply biasing the classifier to previous estimates does not eliminate misclassification. As such, introducing persistence within the classification stage risks keeping errors. As clusters are not preserved between frames, prior data would need to act at a pixel level, with new clusters matched to previous based the proportion of classes within the new cluster from the previous frame.

As a result, data extraction for classification and classification itself will occur on a frame-by-frame basis. However, both aircraft localisation and collision risk detection will be tracked over time, This will be achieved independently of the classifications stage and outside of the image domain.

6.3 Texture

Colour data is highly appropriate for segmentation, as a single variable (i.e. colour distance) can be used to determine similarity between pixels and clusters. However, this same simplicity makes colour less useful for classification. As multiple classes can share the same colour and objects of the same class can be different colours, colour cannot be used as a unique identifier. Instead, colour can at most be considered indicative of class. Even in scenarios where classes are expected to have unique colours, colour cannot be relied upon to be consistent. Within naturally lit scenes, changes in lighting conditions (both seasonal and throughout a single day) dramatically alter the perceived colour of objects. Therefore, colour alone is too variable and insufficiently unique to be used to classify clusters after segmentation.

Instead, machine vision more commonly uses image texture for classification. Rather than being stored in individual pixels, texture data describes the variance in pixel intensity within a defined spatial region. As position and intensity are used to define texture, there are near infinite combinations of texture that can be observed. As such, texture data is often highly indicative of the material of an object, and can be considered close to a unique identifier. Therefore, as most objects are made of a known material, surface type can be used to infer object type, allowing classification through texture alone.

For known objects under observation in controlled conditions, texture based classification is fairly straightforward. However, in outdoor scenes, the variability of surface texture is enormous. Lighting, weather, climate, material age, distance from camera and atmospheric effects can all alter the appearance of a material. In addition, although a single surface type (such as asphalt) may conform to an expected surface appearance, the variability within each class can still be very large. Therefore, a texture recognition approach will require a large training set for each material type, in order to encompass the potential range of texture associated with each class.

6.3.1 Texture Extraction Overview

Before it is possible to extract texture, the concept of image texture must first be defined. Although texture is the change in colour over distance, there is no intrinsic scale upon which texture operates. Just as the resolution of an image defines how many pixels constitute an object, the texture within that object is also equally variable. Different scales of texture can also exist within each other; for example a tiled floor will have both a macro texture of the interlocking tiles, and the micro textures of the tile material. In addition to scale, there are many other variables to consider; such as the directionality of change, the measure of variance and the storage method. Due to this complexity, there is no singular method to describe or store texture data.

Instead, a method must be selected in order to consistently extract data in a manner appropriate for the scenario. As texture data is widely used, various techniques have been proposed to represent texture in a more succinct manner. These representations are often referred to as 'descriptors' as they provide the means to compare different examples of textures. Texture descriptors commonly fall into two categories; 'model-based' and 'statistical' [7].

Model-based methods are the older technique, being introduced to overcome limitations in early computer hardware. As an image is essentially a large matrix, operations performed on the matrix often require many times the memory necessary to store the original image. Early computers often lacked this capacity, and therefore methods which simplified data were keenly sought. As fitting a generic model of texture to the data allows thousands of pixels to be replaced with just a few variables, model-based texture descriptors were highly popular. For classification, parameter comparison techniques can be used to compare the data to known examples. Popular models used for texture analysis include fractals, autoregressive models, fractional differencing models, and Markov random fields.

With increased computational power, these techniques remain popular as they can be applied at very high speed. However, as model based techniques discard much of the data, they often produce inferior results to direct statistical comparison. As computers now have the ability to process information en masse, all the data within an image can be extracted and statistical approaches have become more common. Methods include co-occurrence matrices, grey-level sum and difference histograms, Laws' masks, frequency domain methods, and Gabor filters. Work undertaken by Schwartz (2012) [94] provides a comparison of common statistical texture descriptors for three-dimensional object classification. It concludes that different descriptors outperform each other on different texture types, and at the time of writing there is no single descriptor that is universally better than others.

However, many texture descriptors can only work with certain types of data. As texture descriptors are essentially a pattern recognition task, the data may have to presented in a certain way to extract a consistent result. Therefore, rather than select a texture descriptor based on capability, it is instead easier to select a descriptor based on 'applicability'. i.e. feature descriptors which can function in aerodrome conditions. The selector texture descriptor needs to be:

- Invariant to rotation - Objects in the aerodrome can be approached from any angle.
- Invariant to scale - The distance between the object and the camera is not fixed
- Invariant to changes in brightness - As the images are acquired outdoors, the lighting conditions can vary.

With these restrictions in place, many common feature descriptors can be eliminated from the selection process. As finding an optimal descriptor for outdoor environment is not part of this work, the final choice of descriptor was based on methods which have already been proven to function well in outdoor semantic recognition. As such, a traditional Gabor filter approach and a Local Binary Pattern (LBP) feature extractor were selected.

6.3.2 Gabor Filter Bank

Gabor filters are linear filters, which are primarily used for edge detection [86]. They are especially popular in feature extraction for texture analysis, as the output is thought to be similar to perception in the human visual system [71]. Modelled using a series of highly eccentric Gaussian ellipses, the orientation of the major-axes determines the directionality of the edges detected. Unlike small scale edge detection methods (such as Canny or Sobel) the large size of a Gabor filter allows edges to be detected at any angle, rather than within rows/columns. By varying the size of the filter, different scales of response can be detected. Therefore, in order to capture patterns occurring at different scales and orientations, texture extraction requires a 'bank' of filters to be used.

For this work, the MR8 filter bank was selected, due to its previous success in texture classification [113]. This filter bank consists of two distinct Gabor filters, each applied at 3 scales and rotated to 6 orientations each, in addition to two isotropic filters, as shown

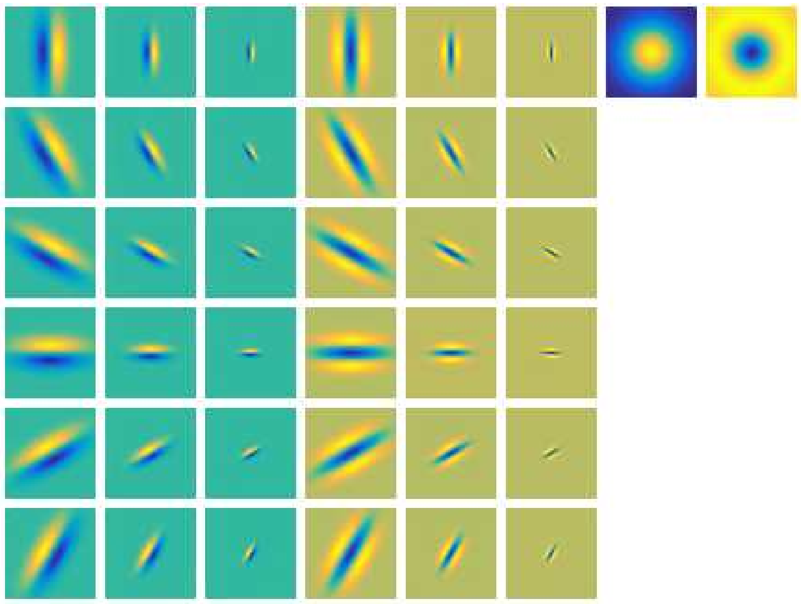


Fig. 6.14 The Maximum Response Filters (MR8) filter bank. Each column represents a single feature response rotated, which can be collapsed after application to produce 8 response channels.

in Fig. 6.14. The MR8 filter bank was specifically designed to describe the appearance of three-dimensional surfaces. To accomplish this, they work with image elements that are mostly invariant to differing lighting conditions and camera orientations, while also benefiting from simple implementation.

To extract texture data, each of the filters within the filter bank are individually applied to the original image. Despite being a texture classification process, it is possible to introduce colour data by applying the filters to each colour channel independently. This would allow texture and colour classification to occur in a single process, albeit with a significant increase in processing required (Assuming three colours channels in the input image, the resulting textons would be three times as long, i.e. 114 filter responses).

However, this coupling has been found to be disadvantageous when the precise colour of the surface is not constant. In an outdoor scene, materials are exposed to different lighting conditions throughout the day, shifting the colour. In addition, certain materials (such as metal) can retain a distinctive texture but have many colours which could be associated with that class. During testing, combining colour data within the texture classifier was found to actually reduce performance. Instead, the original image is converted into an intensity image for processing. (This is achieved using the Y component of the YCbCr colourspace, which is equivalent to a weighted sum of the R, G, and B components, of 0.2989R + 0.5870G + 0.1140B. This is known as Luminance).

Each of the 38 filters within the filter bank are individually applied to the original image, producing 38 'filter response' images. Theses are then concatenated into layers; similar to how colour channels are stacked in colour images. As a result, each pixel in the original image is replaced by a 38-element vector, forming a multi-channel response which represents the texture at that point.

For scenarios in which texture is expected to conform to the same orientation (i.e. rotationally dependant), all 38 responses can be used to maximise the distinctiveness of the texture response. However, for aerodrome use, detection will focus on terrain features and surface markings, which can be approached from any angle. As such, texture alignment is not necessary, as introducing rotational constraints into the classifier can only reduce performance. Instead, as all 38 filters are still required for extraction, an additional step is introduced after the responses are created. Rotational invariance is achieved by 'collapsing' data from the various orientations, by taking the maximum single response for each scale of filter. As such, the final result consists of 8 responses (two Gabor filters at three scales and two isotopic filters), each consisting of the maximum orientation result (hence the name Maximum Response Filters (MR8)).

To allow for comparison, both the filters and the order in which they are arranged must be kept consistent throughout. As the original pixel data is directly replaced by a filter response, a benefit of this approach is that it produces spatially cohesive results, i.e. the position of the filter response matches the original image. Therefore, data can be grouped directly into the clusters created during the segmentation phase. As Gaussian filter responses are pixel orientated, each response vector is also independent of its neighbours. This allows the textons (covered in Section 6.3.4) from each pixel within a cluster to be amalgamated into a single vector, without reference to their original neighbours.

6.3.3 Local Binary Patterns

A feature of the MR8 filter bank which provides both advantages and disadvantages is the size of the filters used. As can be seen from Fig. 6.14, the sample size of each filter is large, capturing 49 by 49 pixels (2401) pixels in total. The large sample area allows the method to capture texture data over a wide range of scales, allowing both large and small texture features to be detected. As the imagery is expected to mainly consist of large regions (taxiway, sky and grass), large sample sizes provide more data for the classifier, which should increase confidence in the result.

However, the large number of pixels required impacts the precision with which each filter can be applied. Near cluster boundaries, a large texture descriptor will sample from multiple regions at once, potentially capturing data from multiple object types and therefore lowering the accuracy of classification. To provide a better estimate of texture for smaller regions, a smaller descriptor chapan be used, minimising the risk of sampling data from outside the class. Previous work [22] has demonstrated that 'small image patches', as compact as 3x3 pixels, are sufficient for texture data extraction. As the area is so small, filter based methods are unsuitable. Instead, the LBPs texture descriptor has been chosen as an additional texture extraction method.

LBP assigns each pixel in the original image a texture response, based on it's direct neighbours. Comparing the intensity of the surrounding pixels, those with greater intensity are labelled as '1', whilst all others are '0'. Reading in a clockwise direction from the upper right neighbour, the neighbours in sequence produce a binary number which describes the relationship of the central pixel to it's neighbours, i.e. the localised texture. With 8 neighbours, an 8-bit binary number results in 256 different possible textures. As with the filter response above, rotational invariance is also possible, by collapsing the non-unique patterns down into groups, reducing the possible outputs to just 59.

Despite it's simplicity, LBP has proven capable at small scale texture classification and is widely used for situations which require small texture features extraction, such as in human face recognition [6]. In addition to rotational invariance, LBP is also highly invariant to changes in brightness. As a total brightness change will shift the intensity of all pixels together, the relative difference between neighbouring pixels will remain the same.

The results in Varma (2003) [113] demonstrate that LBP can outperform filter-bank derived textons in certain conditions. However, the main drawback of LBP is that highly localised sampling can fail to detect larger texture features. Therefore, both MR8 and LBP are to be used in combination for texture based classification.

6.3.4 Textons - Filter response simplification

As shown in Figure 6.16, for each pixel in the original image, the MR8 filter bank responses are gathered into a vector. This vector represents all the filter responses, and therefore the texture, for that single pixel within the image. As this is a single point measure of texture, the term 'Texton', is commonly used [56]. As such, a multi-filter texture

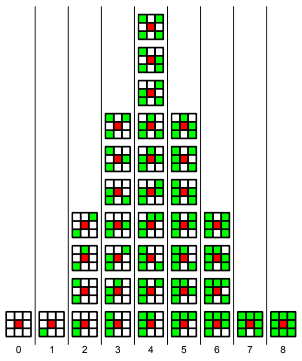


Fig. 6.15 36 Rotationally Invariant Local Binary Patterns

response image is formed of ‘textons’ in the same manner that a colour image is formed of pixels.

Both LBP and Gabor approaches produce spatially cohesive response images. However, while LBP produces a single response, the MR8 filter bank produces a separate response for each filter used. As stated above, rotationally invariant LBP can produce 59 possible responses for each original pixel. By contrast, assuming the response is stored in an 8-bit format, there are 255^8 possible Texton responses from an MR8 filter bank. As this represents an enormous amount of variability, discretisation is required to simplify the results. This was performed by broadly following the methodology outlined in the widely known ‘Textonboost’ work [101] (originally proposed in [65] and explained in greatest detail in [128]).

As multiple filter responses cannot be easily binned, discretisation is achieved by replacing each raw texton with the closest ’Typical texton’ from a ‘texton dictionary’. This allows a reduced texton set to represent all texture in the image, as shown in Figure 6.17. To create these ’typical textons’, image data containing all of the potential classes is convolved using the same filter bank, which creates several hundred thousand raw textons, covering the entirety of feature space. K-means clustering can then be used to find a desired number of ‘typical textons’ from the data.

To finalise the dictionary, the number of words must be decided upon. Greater numbers of typical textons can increase classification accuracy, however the computational cost would also increase. Work done in [10] uses textons in terrain classification for robots, demonstrating real world application for a similar task. In that work, a hierarchical classification is used, with two separate applications of textons. The first uses only 20 typical textons and is meant to rapidly determine classes with unique textures. The second is a more in depth analysis with 40 textons, designed to distinguish between very

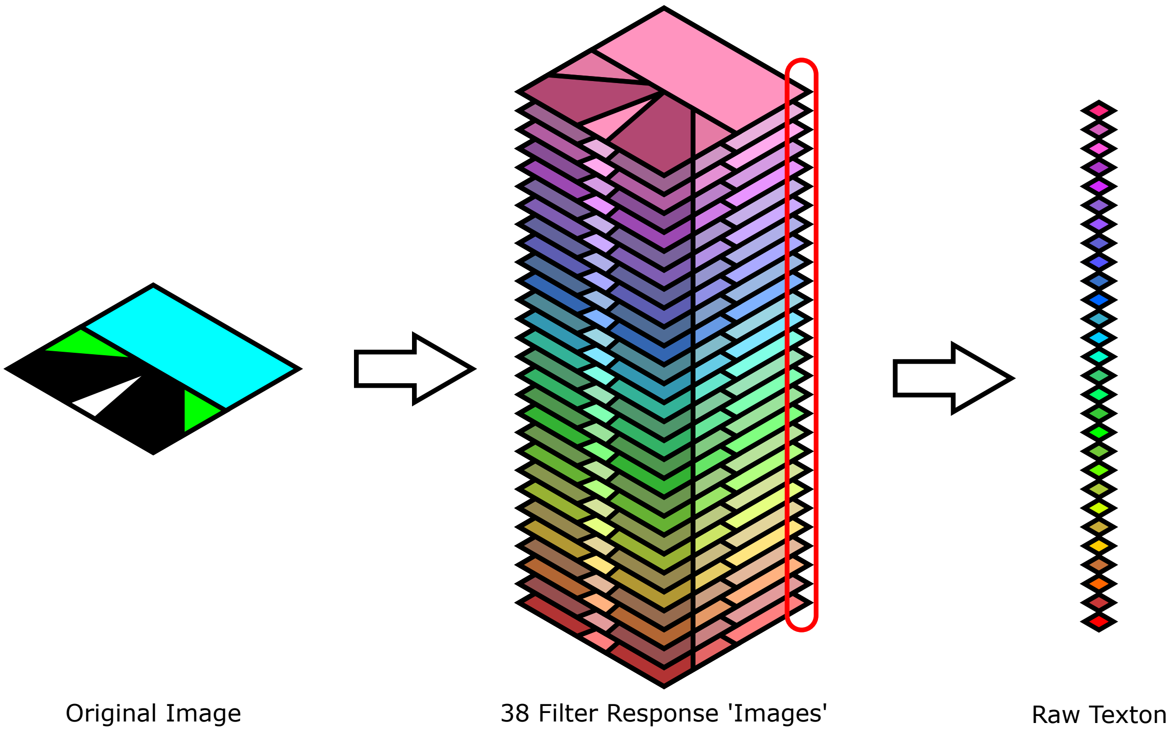


Fig. 6.16 Visual depiction of a ‘texton’, created by applying each filter within the Maximum Response Filters (MR8) filter bank, then combining the ‘pixel response’ from each filter into a vector.

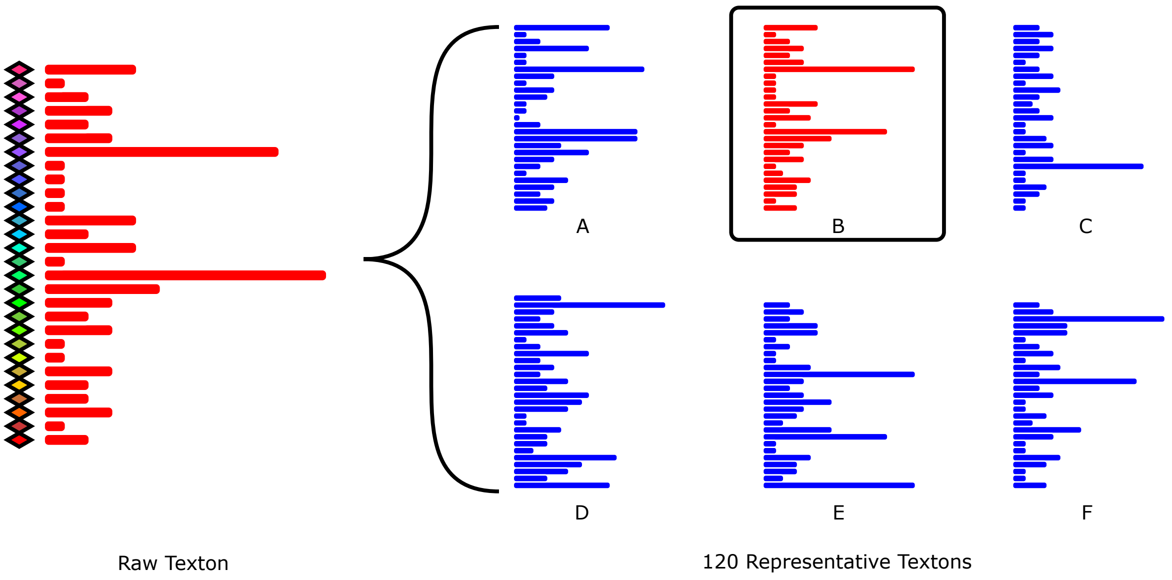


Fig. 6.17 Discretisation of Textons is achieved by comparing each unique texton to specially created ‘representative’ textons. As each response is independent, they can be used as co-ordinates within multi-dimensional space to determine which representative texton is most similar.

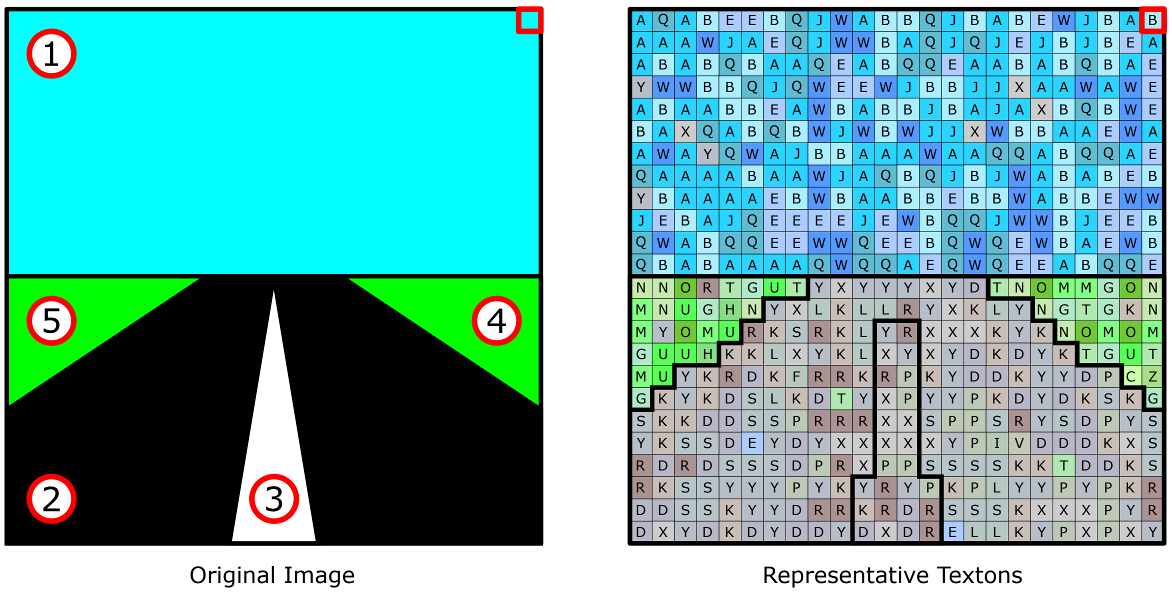


Fig. 6.18 High simplified example of a texton representation of an aerodrome image.

similar texture classes.

The reason for using a hierarchical classification in [10] is the limited processing power on board a small robot. However, as this is not considered an issue for the future UAS, a larger spread of textons can be used in a single stage. Therefore, for this work the texton dictionary will consist of 120 typical textons. As shown in Figure 6.18, the texture data is reduced down into a single indexed response image, much like the LBP result, where each pixel is replaced by a single integer representing a local texture response. To allow comparison between clusters of different sizes, the area of each cluster is used to normalise the result, producing a description of texture which is cluster size invariant.

A drawback of textons is that they are computationally intensive. However since their inception computational power has increased enormously and more literature has started to use them in image segmentation, such as [101]. Of further benefit, [101] suggests that as textons rely on dense features, both highly textured and untextured objects receive the same level of distinction. This is highly useful when dealing with objects with gradual texture changes, such as occur over distance (e.g. a runway tending to the horizon).

6.3.5 Histographic Texture Comparison

As both LBP and MR8 texture descriptors are spatially cohesive, the results for each cluster occupy the same positions as the pixels in the original segmentation, making extraction very straightforward. (The process described here will refer to Textons created using the MR8 filter bank, however the same texture comparison method is equally applicable to the LBP results).

After segmentation, the Texton response of each region can be simply extracted using an image mask. As the order of the Textons is no longer important, each cluster can be represented as a vector of its texton contents. As each possible texture response is independent from all others, a further simplified output (such as mean) is not possible. Instead, as the output from both MR8 and LBP consists of discrete responses, the texture data is most easily represented by the total number of each response within that cluster, much like a histogram. Therefore, comparison to known examples of each class is made in this form.

Figure 6.19, shows the normalised histogram outputs, based on the clusters established in Figure 6.18. Despite the difference in cluster size, the normalisation produces similar histograms for regions of the same type, which can then be used for classification. (The painted runway centreline and the taxiway surface produce a similar texture result, as occurs in reality).

In order to classify clusters, the cluster data must be compared to each expected class. As the texture training data consists of many thousands of independent samples, a method of representing the potential texture range of each class must be determined. In addition, a distance metric is required which defines similarity/dissimilarity between the cluster and each class. As the texture data is stored as a histogram, both parametric and non-parametric methods of histogram comparison have been investigated.

For clarity, both parametric and non-parametric methods use parameters. The difference is that for parametric models the parameters are defined by the user prior to application to the data (typically by creating a model), whereas non-parametric methods determine the parameters directly from the data.

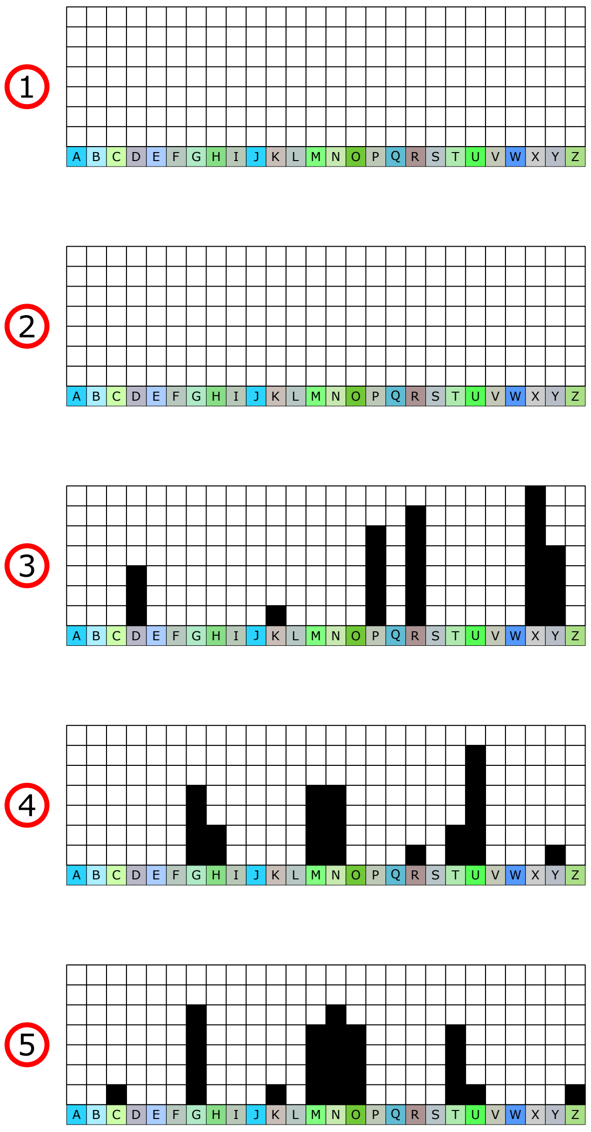


Fig. 6.19 Texton response histograms for each cluster established in Figure 6.18

6.3.6 Distance Metrics

Non-parametric methods are often preferred over parametric methods as they as simpler to implement; as the parameters are extracted directly from the dataset, they do not require the user to understand the underlying principles in order to produce a good result. However, as non-parametric comparison methods rely on entirely on comparing points within the dataset, the distance metric used can greatly affect the result. Therefore, in addition to comparing different non-parametric methods, different distance metrics must also be considered.

As the data for each cluster appears similar to a histogram, histogram similarity metrics would seem a viable method, with *Intersection similarity* and *Hellinger similarity* both tested. However, as each sequential entry in the histogram is actually an independent count of texture and unrelated to its neighbours, the applicability of these methods is questionable.

In [33], the Bhattacharyya coefficient measure was used as the distance metric. However, this was found to have computationally intensive. As it is not possible to produce an 'average' texture for each class, many examples of each class were required in order to cover the entire feature range. As the Bhattacharyya coefficient measure can only compare two distributions directly, every sample in each class required an individual comparison. Although the results were favourable, histographic comparison methods which were easier to perform en masse were sought instead.

Therefore, additional distance based metrics are also explored. As each texton response is independent, it is possible to treat each of the 120 manual chosen textons as individual dimensions within a multi-dimensional feature space. Distance can then be calculated by combining the results from each texton, much like colour distance is the Euclidean extension of distance within each individual colour channel. As each 'dimension' is independent, the use of Euclidean 2-norm geometry is not guaranteed to provide the best indicator of distance. As such, the Manhattan (or taxicab) distance metric (also known as Euclidean 1-norm distance) was also included. Four different distance metrics were considered in total:

- Manhattan
- Euclidean
- Intersection similarity
- Hellinger similarity

	K-Nearest Neighbour		Average Distance to 10 nearest Neighbours	
	MR8	LBP	MR8	LBP
Manhattan	19.91	46.26	88.73	87.54
Euclidean	62.92	70.23	88.85	87.62
Intersection Similarity	53.34	59.36	88.73	87.54
Hellinger Similarity	82.37	37.52	88.94	68.53

Table 6.1 Percentage correct classification of pixels for distance comparison methods

6.3.7 Non-Parametric Distance Comparison

In order to determine if a non-parametric method is suitable for classifying texture data, one hundred typical aerodrome images (captured during aircraft taxiing at Coventry airport, UK) were manually classified to provide 'ground truth' images. Eight classes were chosen to test the suitability of texture classification. These were; Taxiway surface (asphalt and concrete), Grass, Plants (all bushes and trees which could not be considered grass), Sky, White painted surface markings, Yellow painted surface markings, Buildings (based on corrugated metal walls) and Vehicles (based on metal panels on aircraft and ground vehicles).

Of the one hundred classified images, eighty were used to provide training data for the classifier, whilst the remaining twenty were used for testing. For the twenty test images, clusters were generated using the *superpixel and reachability clustering* method defined in Chapter 5. Texture data for both LBP and MR8 texture descriptors was then extracted from each cluster, using the methods explained previously in this section. Combinations of each distance metric and non-parametric comparison methods were then applied, with the results shown in Table 6.1.

The simplest non-parametric distance method for classification is direct nearest neighbour comparison. Direct nearest neighbour is most suitable when classes are distinct, or when there is only a single data point is attributed to each class. For texture, the training set contains many thousands of examples of each class, with certain classes overlapping. As such, direct nearest neighbour distance was only found to work well when using training data extremely similar to the test environment. As this form of comparison was found to give highly unpredictable results, it has not been explored in further detail.

Instead, two other non-parametric methods of distance comparison have been tested. The first is the popular k-Nearest Neighbours (KNN), in which a threshold distance is established around cluster data point. The number of samples of each class within that threshold distance are then used to assign probability of belonging to said class. From Table 6.1 it can be seen that the success of using KNN is highly variable, depending on the distance metric used. The difficulty in using KNN to compare texture data is the inconsistent local structure of the data. Although certain classes are grouped tightly, others are much more dispersed. As such, the number of points within the decision boundary was class dependant, and therefore produced highly varying results.

The second method of distance comparison has been specially devised to overcome the issue of inconsistent local structure. Rather than a fixed threshold, the average distance to a fixed number of the closest examples of each class is used as an indicator of similarity. Empirical testing found that the mean distance to the closest 10 examples of each class provided a suitable measure of distance. From Table 6.1 it can be seen that the average distance method is far more successful in correct classification compared to KNN. In addition, the results are also fairly consistent, regardless of the distance metric. Therefore, the conclusion can be made that the distance metric is not overly important, provided that that the method of comparison is suitable.

Given that the average-distance method performs far better than KNN, this method is seen as the best example of non-parametric classification for texture. However, there are issues with using this technique in the final system. In order to determine the 10 closest points, a sorting algorithm must be used. Such algorithms are famously computationally intensive, especially considering that the data set contains many thousands of samples and has a high dimensionality. As the sorting process must be repeated for every cluster within the original image, this can require significant processing time.

As image segmentation is already computationally demanding, requiring additional time during classification reduces the rate at which the system can function. Approaches which access the training data directly during classification will always be slow. The alternative is to pre-process the data in order to create a model which improves the speed of classification. Although a model is introduced, texture is still described using the raw response, thus such an approach is parametric classification, rather than texture modelling. Therefore effort has be made to see if comparable results can be achieved using parametric methods, without the large computational time.

6.3.8 Parametric Distance Comparison Using Support Vector Machines

Non-parametric models are highly fitted to the data set on which they were created. As the methods proposed above require examples for comparison, they cannot infer that a cluster could belong to a class unless it can be compared directly to known examples. Although this makes them statistically more 'robust' to misclassification, this requires an extremely large data set to ensure that the entirety of each class are included. Alternatively, individual points within a dataset can used to generate a model with which each cluster can be compared.

Conventionally, model fitting requires creating a model bespoke to the problem space. Although such an approach usually guarantees good results, it can be very time consuming and becomes increasingly difficult as dimensionality increases. As such, a method which can establish a simple parametric model for any dataset in high-dimensional space is very useful. Therefore, a SVM approach has been explored. SVM are supervised learning models, well regarded as efficient classifiers. Rather than define a specific model, SVM uses training data to determine a 'hyperplane', which divides space into two regions. This can either enclose a single class within a volume, or can separate infinite space into two regions denoting different classes. New data points are then classified based on which side of the hyperplane the are located, and the distance from the hyperplane then establishes the confidence in the result. Although hyper-planes are linear, a kernel function can be used to map the training data into increasingly higher dimensional space, until a linear result can be achieved (which is required for single-class classification). As this is very fast to process, SVM are highly efficient classifiers and have proven to be very reliable.

As the kernel function is responsible for establishing the classification model, the implementation determines if the SVM is either parametric or non-parametric. For example, Gaussian Radial Basis Function (RBF) kernels are non-parametric as the distance between the training points is used to establish the kernel matrix. As the kernel is based directly on the dataset, the complexity of the kernel is dependant on the amount of data available in the training set. This allows the complexity of the kernel to adapt to the dataset, ensuring a suitable model is generated. As such, Gaussian RBF kernels are the most common form of kernel used for SVM.

However, as stated above, non-parametric models rely on large datasets. Despite using eighty aerodrome image for training, certain classes (such as surface markings) represent a much smaller proportion of the data set than others (such as asphalt or sky) as they occur less frequently at aerodromes. This makes RBF based SVM more likely to reject examples of the class which are not suitably similar to the known examples. Although this could be resolved by specifically gathering examples of under-represented classes, access to aerodromes environments is difficult and the available dataset was limited to what could be collected from aircraft mounted footage. Instead, the alternative approach is to use a parametric kernel, as this can accurately fit a hyperplane using a much smaller data set. As such, this work makes use of a polynomial kernel, with an experimental derived degree parameter of 30 (due to the extremely high order of each Texton co-ordinate i.e. 120 dimensions).

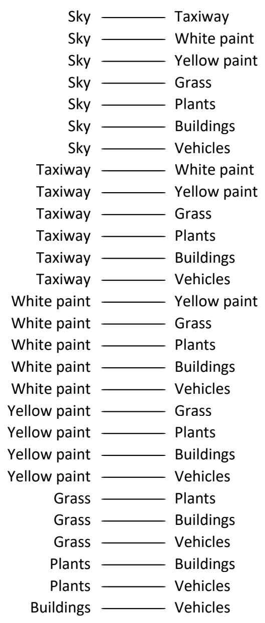
Due to the ease of application and suitability for this work, SVM has been the only parametric approach considered for classification. The major limitation of SVM is that each hyperplane can only divide space into two regions, (i.e. SVMs are binary classifiers). Therefore, methods of extending SVM to allow multi-class classification are required.

6.3.9 Multi-class SVM

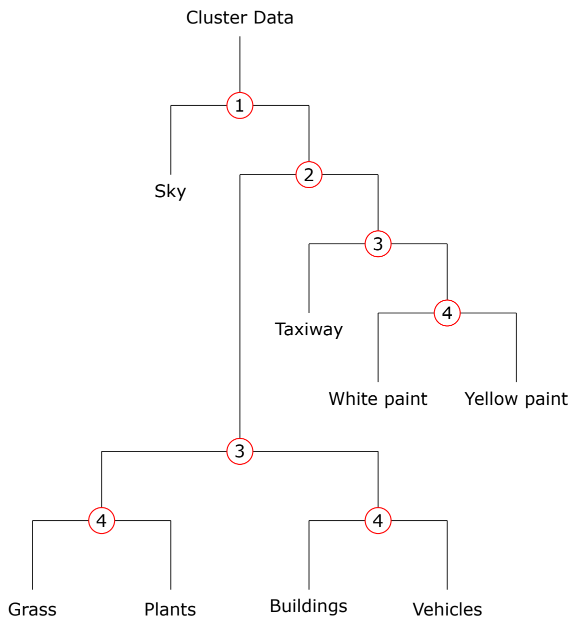
As a single SVM can only ever distinguish between two different classes, multi-class SVM classification is primarily achieved by simply adding additional SVMs. A method of multi-class SVM classification which relies almost entirely on adding more classifiers is Pairwise Majority Voting (PMV). As the name suggests, SVM classifiers are created for each ‘pair’ of potential classes. As the data under classification may not belong to either class, the winning class receives a ’vote’, despite the fact that it may not actually be the correct.

As the correct class should always ‘win’, the class with the most votes after all classes have been compared to each other is then considered to be the overall winning class. As shown in Table 6.2, this produces a result as good, if not better, than the aforementioned non-parametric methods.

However, despite remaining a common approach for SVM, this method has many drawbacks. As many classifiers are required, the process is computationally intensive (although still far quicker than non-parametric approaches). In addition, when compar-



(a) Pairwise Majority Voting



(b) Binary Decision Tree

Fig. 6.20 Comparison of PMV and BDT voting complexity for SVM.

ing two classes of which neither is the correct classification, a winner is still declared. Although a majority vote should always go to the true class, it is often common for the next most likely class to lose by a single vote, only being beaten by the true class. As such, confidence in the end result is often difficult to extract, as strong correlation appears the same as weak correlation.

To resolve these issues, a BDT based implementation has been suggested in [70]. Rather than vote for independent classes, a sequential process leads to the end result with far fewer classification attempts. Beginning with the entire training data set, a clustering approach based on maximum separability divides the data set in two [123]. Each subset then undergoes the same process, until only individual classes are represented. SVM classifiers are then created to produce the vote at each stage, leading to high speed classification. Due to a single vote dimissing entire classes at once, far fewer votes are required.

Figure 6.20a shows the PMV votes required to differentiate between just eight classes, requiring 28 SVM classifications to occur each time. By comparison, by combining classes based on maximum separability, the BDT approach requires a maximum of four SVM classifiers to determine the final class.

Table 6.2 shows the results of the SVM classifiers. As there is very little difference in terms of accuracy, yet significant improvements in online computation efficiency, SVM

	MR8	LBP
Best distance comparison result	88.94	87.68
SVM Voting	88.91	87.70
SVM BDT	89.01	87.80

Table 6.2 Pixel-wise correct classification for SVM implementations

BDT is the most appropriate parametric classification approach. In addition, as the results are very similar to the best results from direct, non-parametric comparison, BDT based SVM has been selected for classification of both MR8 and LBP overall.

6.3.10 Conclusion

Image texture data is commonly used for classification, as classes can often be uniquely identified using texture alone. As texture is defined as the change in pixel values within a neighbourhood, there is no inherent unit of measurement, requiring feature descriptors to define texture in a comparable form. Due to elements of texture existing at different scales, two different forms of feature descriptor has been selected; the MR8 filter bank to capture larger texture features and LBP to capture highly localised texture.

As classification occurs within a segmented image, texture definitions must also be spatially constrained so as to only sample data from specific areas. A texton based implementation, in which each pixel within the image is assigned a localised texture response, has been implemented as the solution converts an MR8 result into a form similar to LBP, allowing the same interpretation methods to be used for both feature descriptors.

In each case, the contents of a cluster require comparison to known examples of each class, with the smallest 'distance' used as a measure of similarity. To select an appropriate solution, both parametric and non-parametric methods have been tested, along with several distance metrics. Non-parametric methods are simple to implement, but are often computationally intensive as they need to access the training data directly during classification.

The best results from non-parametric distance comparison used the average distance to the closest ten examples of each class Despite good results, this requires an additional sorting algorithm to determine the ten closest points. As an aerodrome data set is likely to contain many thousands of samples with high dimensionality, this can require significant processing time, especially as the sorting process must be repeated for every cluster within the original image. The alternative is parametric classification, where the training data is pre-processed to create a model which improves the speed of classification.

From a comparison of several techniques, an SVM approach which uses a BDT to minimise the processing cost has been chosen. The allows comparable results to the best of non-parametric methods, with a greatly reduced computational requirement.

Although BDT based SVM has been found to produce a fairly accurate texture only classification, the results are still far from perfect, with less than 90% accuracy overall (shown in Table 6.2). This is mainly due to classes with very similar texture (such as asphalt and surface markings) being difficult to differentiate on texture alone. Therefore, data fusion with other types of information will be required within the classifier.

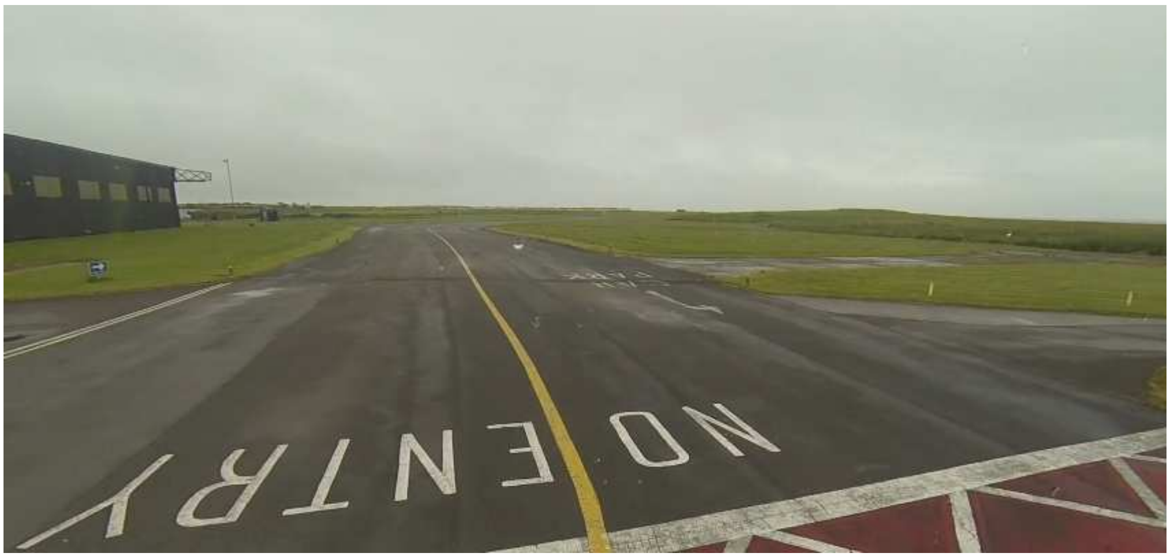
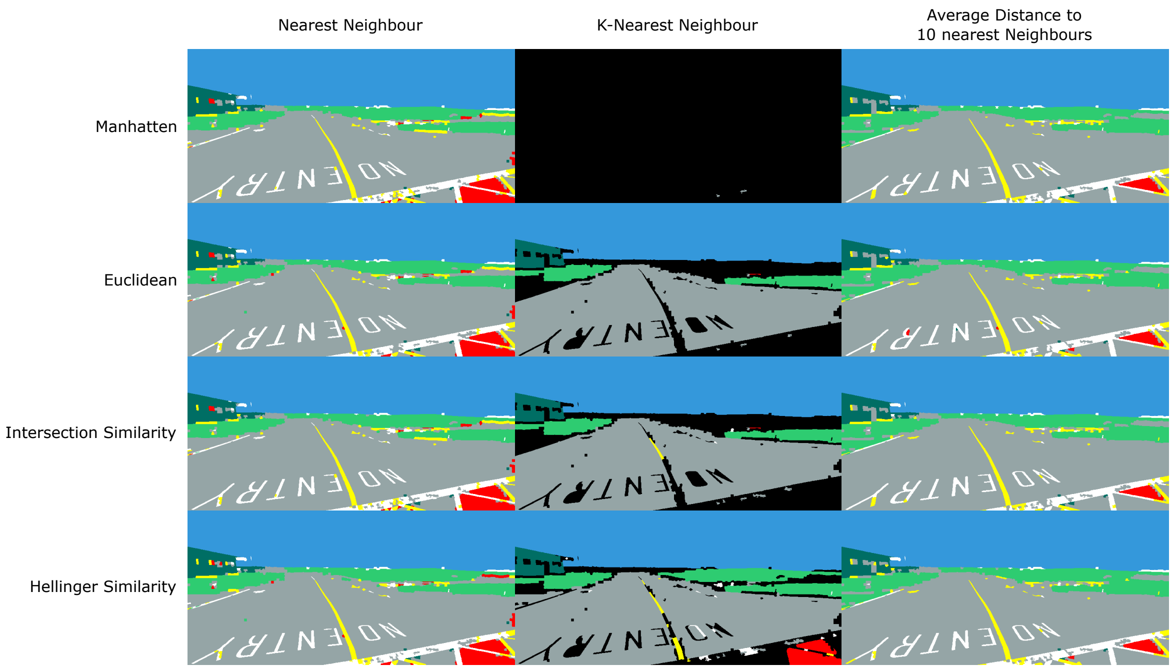


Fig. 6.21 Original example image used for comparing texture classification results



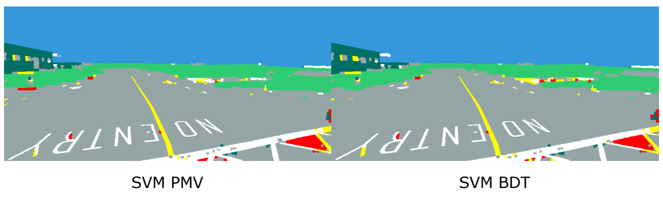


Fig. 6.22 MR8 Texture classification results for example image shown in Figure 6.21. Each results shows the most likely class, without removing results with low probability. The black regions for K-nearest neighbour classification signify insufficient neighbours were found to make an estimate. Aside from these 'missing' classification results, all of the classifiers correclty classified the majority of clusters, confirming that texture is an appropriate method of cluster classification. For precise results, please refer to Tables 6.1 and 6.2.

6.4 Colour

From Table 6.2, it can be concluded that the SVM texture classifier is at most only around 90% accurate. The 10% error is the result of similar classes being misclassified as each other, with asphalt and painted surface markings being difficult to differentiate on texture alone. Examining the performance per class, around 30% of surface marking clusters were misclassified as asphalt.

At most aerodromes, there are many different types of lines on the ground in order to convey information. Both taxiways and runways have different marking conventions to help guide aircraft, whilst most ramp areas will have additional lines specifically for ground support vehicles. Despite not posing a collision risk to the aircraft, correctly detecting surface markings is important as they can provide additional information. As other vehicles follow surfaces markings, they could be used to gain contextual information, such as the most likely route a ground vehicle will take through the aerodrome. Alternatively, as surface markings themselves are in known locations, they are also potentially useful for localisation. Therefore, correctly extracting surface markings can potentially improve the control of the UAS.

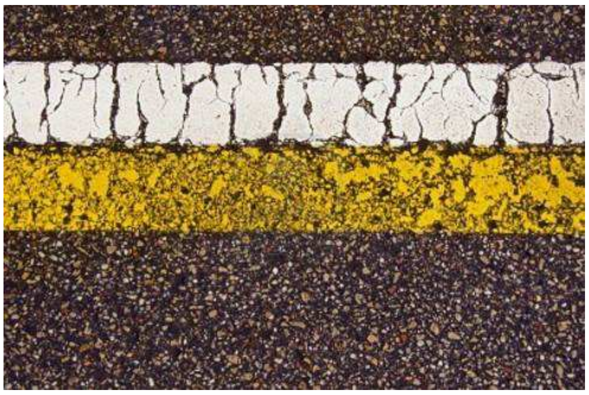


Fig. 6.23 Examples of surface types with similar texture data but differing colour

To improve the classification result, additional data is required which can differentiate between classes with similar texture more easily. For painted surfaces markings, colour data is the obvious choice, as can be seen in Figure 6.23. Although colour cannot be used as a primary classifier (due to multiple classes sharing the same colours) the use of colour data to improve a primary texture based classification approach is well established [52].

In machine vision, the term ‘colour’ refers to the wide array of different methods of representing the value of individual pixels. The components of colour include brightness

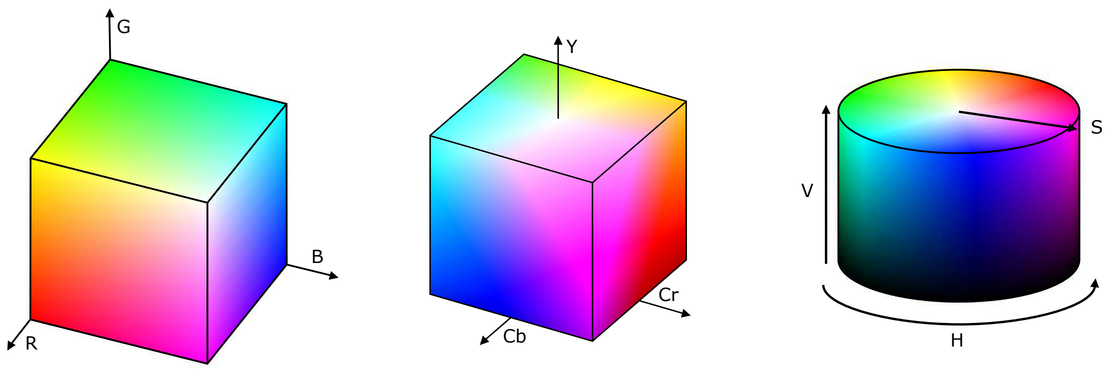


Fig. 6.24 Comparison between RGB, YCbCr and HSV colourspaces. The RGB colourspace is cubic, with changes in chroma, saturation and brightness distributed throughout all three colour channels. The YCbCr colourspace is also cubic, however brightness data has been isolated within channel Y. The chroma and saturation data remain distributed within the Cb and Cr channels. In contrast, HSV colourspace is cylindrical, in which colour component changes are isolated within independent colour channels.

(luma), saturation and hue (chroma) in addition to many other variants. Different representations separate or combine components in 'colour channels' to make the data better suited for its application. (For example, the common RGB colourspace is both similar to how the human eye captures chroma, as well as being ideal for computer display). Although changing how colour is represented does not modify the data, it can simplify the data extraction process and make interpretation far easier.

As different colourspace representation alter the ability to interpret colour information, this section discusses colour extraction methods; beginning with conventional colour extraction from the YCbCr, RGB and HSV colour spaces. In addition, a novel application of relative luminance is also proposed, specifically intended to improve the detection of taxiway surface markings.

6.4.1 Colour Extraction in Different Colourspaces

In Section 6.2.1, the issues with image storage and the loss of data, especially in the chroma components, were expressed. Despite the reduction in data, compared to what could be obtained using a custom camera, compressed chroma data remains an important source of information for classification. Although conversion from one colourspace to another cannot restore the lost information, it can make processing easier.

As colour information for display still makes use of RGB, images are often processed within the RGB colourspace. RGB is designed to reflect how digital screens display images (which are themselves based on how human eyes detect chroma) with chroma, brightness and saturation data intermingled in the three colour channels. Interpreting RGB as a co-ordinate system, the colourspace is most easily represented as a cube, as shown in Figure 6.24.

In a previous work Eaton (2015) [33], RGB colour information was used directly to produce a classification estimate, similar to the result of the texture classifier suggested above. To determine the most likely class based on colour alone, the entire colour spectrum was simplified to just 15 colours (12 different hues in addition to white, grey and black). This small number of colour should be sufficient to different not only between asphalt and surface markings, but also between the surface markings themselces (as runway markings are white, taxiway lines are a bright yellow/orange colour and instructional markings are typically in red).

As determining the boundaries between colours is difficult (human perception of colour is non-uniform) the domain of each discrete colour was mapped through a manual training process. New data within the RGB colourspace was then assigned one of the 15 discrete colour classes depending on these boundaries. Simple observation then linked each discrete colour class with each object type. For example, a green cluster would be assigned a high probability of being grass.

Although combining colour and texture data in this manner provided a better result than texture alone, this simplistic implementation required that objects remain the same colour. However, outdoor scenes are subject to atmospheric lighting conditions which can dramatically alter the perceived colouration. During a single day the colour of an object appears to change depending on cloud cover and the position of the sun in the sky. For example, the yellowing effect of evening light can make white taxiway markings appear yellow, while white taxiway markings in low light can be darker than asphalt in bright light. (Seasonal effects can make the situation far more complex but have not been considered in this work).

Although the RGB colourspace is highly practical for storage and display, it is not well-suited for image processing. As simple changes (such as an increase in brightness) affect all three channels at once, two visually similar colours may have dissimilar values in the RGB colour channels, making it difficult to create rules for discretisation. As such, it was concluded that attempting to use colour within the RGB colourspace was not the best solution.

Machine vision tasks more commonly use other colourspaces instead. For example, the Superpixel generation approach used in Chapter 5 made use of the *Lab* colourspace, due to the colourspace being specific designed for uniform colour distance based on human interpretation. Both the Lab and YCbCr colourspaces have the advantage of separating out the brightness component into independent channels. However, as both saturation and chroma components remain combined, these colourspaces only offer a small advantage over RGB. Instead for this work, colour classification is primarily achieved within the HSV colourspace, which is commonly used for image classification [30].

6.4.2 Hue, Saturation and Value

As the name suggests, HSV is a representation of colour in which Hue, Saturation and Value are entirely independent. Unlike most other colour components, Hue is continuous and repeating, most common expressed using a 'colour wheel'. As such, unlike RGB, and YCbCr, when interpreting HSV as a co-ordinate system the colourspace is most easily represented as a cylinder, as shown in Figure 6.24.

HSV is designed to make human interpretation easier, separating colour data into channels reflecting how human vision functions. The main benefit is that the image brightness (*Value*) is separated from the chroma information, making changes in brightness easier to observe. For example, overcast skies will reduce the brightness of pixels which make up a taxiway centreline. In RGB colourspace this would affect all three colour channels making it difficult to anticipate the change. By contrast, in HSV only the *Value* channel should be affected in any significant way. This separation allows Hue based colour classification to be more robust to changes in lighting, as well as allowing for intuitive rules to be established for classification.

As Hue is 'circular', directly extracting the average Hue from multiple pixels can be difficult. Despite the cylindrical colour wheel fitting well with human understanding of colour, in physics differences in Hue are caused by variance in the wavelength of light. As the visible spectrum has two ends, the circular nature of hue is purely down to human perception. Instead, the cluster data is first averaged within Lab colourspace, before conversion to HSV. Once converted, the HSV colour can then undergo further discretisation.

Discretisation is commonly performed prior to classification as many classifiers can only work with discrete data. In addition, a discrete data set is much smaller than the original data, significantly increasing classifier performance. (As nearly all images are stored using 8-bit integers, colour data is already discretised; however HSV colour is typically stored in 256 bins per channel, with each image potentially containing up to 16 million unique colours. Despite being discrete, the colours are far too numerous to be of

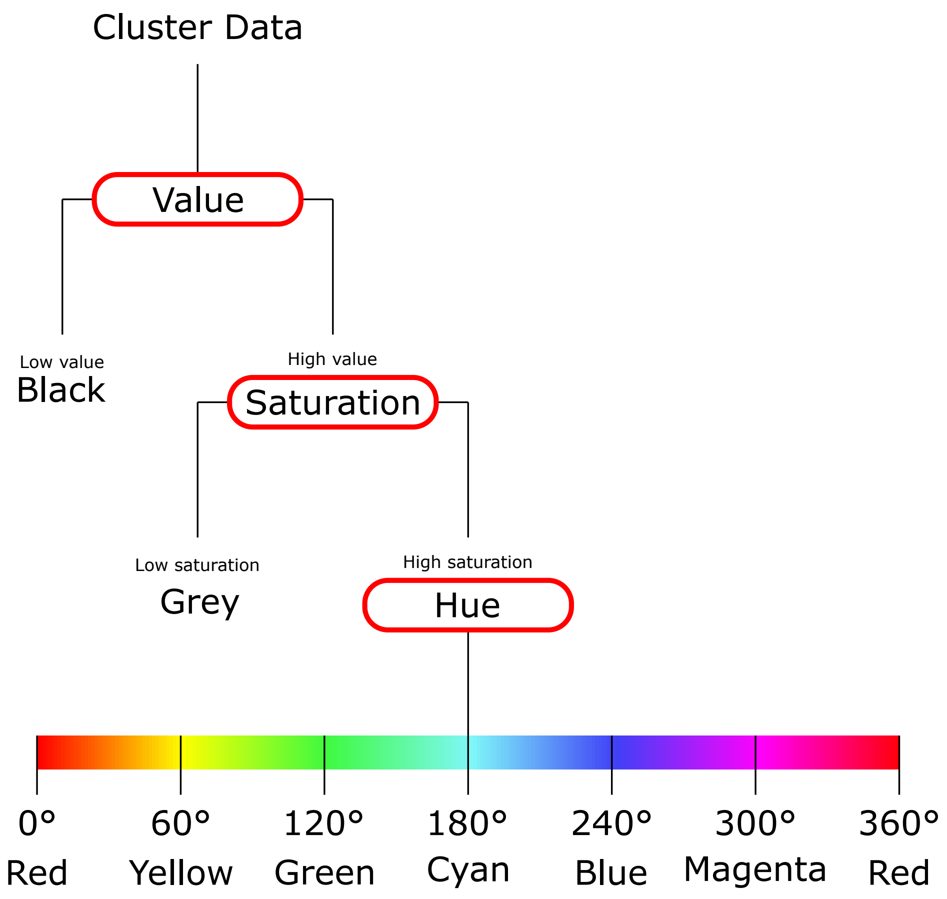


Fig. 6.25 Potential colour extraction process, using the HSV colourspace

direct use and the data must be instead be binned into wider groups).

In practice, discretisation affects all three colour channels simultaneously. However, the logic behind HSV classification is more easily thought of as a sequential process as shown in Figure 6.25.

The 'first' channel to process is *Value*, which represents distance from black within the cylindrical co-ordinates. Unlike luminance (which is true distance from black in terms of photon intensity), the colour primaries (i.e. red, green and blue) and pure white will all share the same maximum *Value*, despite primary blue only being around 10% as bright as pure white. This is extremely useful, as it provides clear distinction between dark coloured pixels and black, making colour classification more robust for classes such as grass and asphalt. As all other classes are typically bright in colour, low *Value* is considered indicative of asphalt. The *Saturation* channel denotes the intensity of colour and is therefore applicable to the classification of every region which is not identified as black. Colourful objects have high *Saturation* values whilst grey objects have low values. Yellow

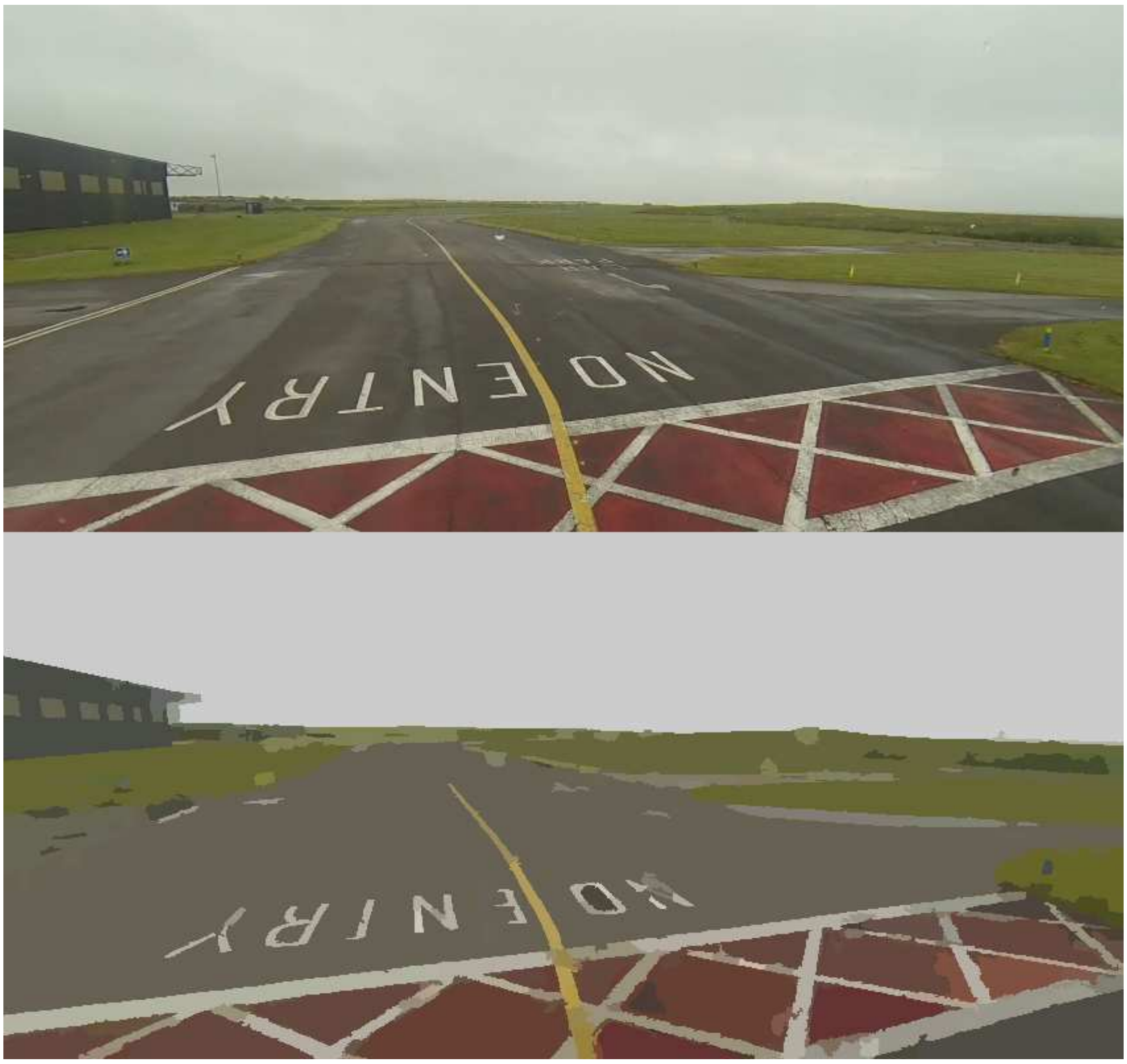


Fig. 6.26 Image represented using discretised HSV colours only

and red surface markings, and grass were all expected to be colourful whilst white surface markings and asphalt were not. To align with testing conditions, sky was assumed to be grey to reflect the overcast skies typical of UK weather.

As both *Saturation* and *Value* are highly influenced by changes in lighting conditions, large groupings were deemed appropriate to attempt to mitigate small fluctuations. As such, both channels are only assigned 10 states, in increments of 0.1 between 0 and 1.

By comparison, small changes in *Hue* can dramatically alter the perception of an object, especially at high levels of saturation. Therefore, *Hue* is discretised into 24 discrete states. As hue represents the angular position within the cylindrical colourspace, it is discretised in between 0 - 360° in 15° increments. As *Hue* and *Saturation* are difficult to

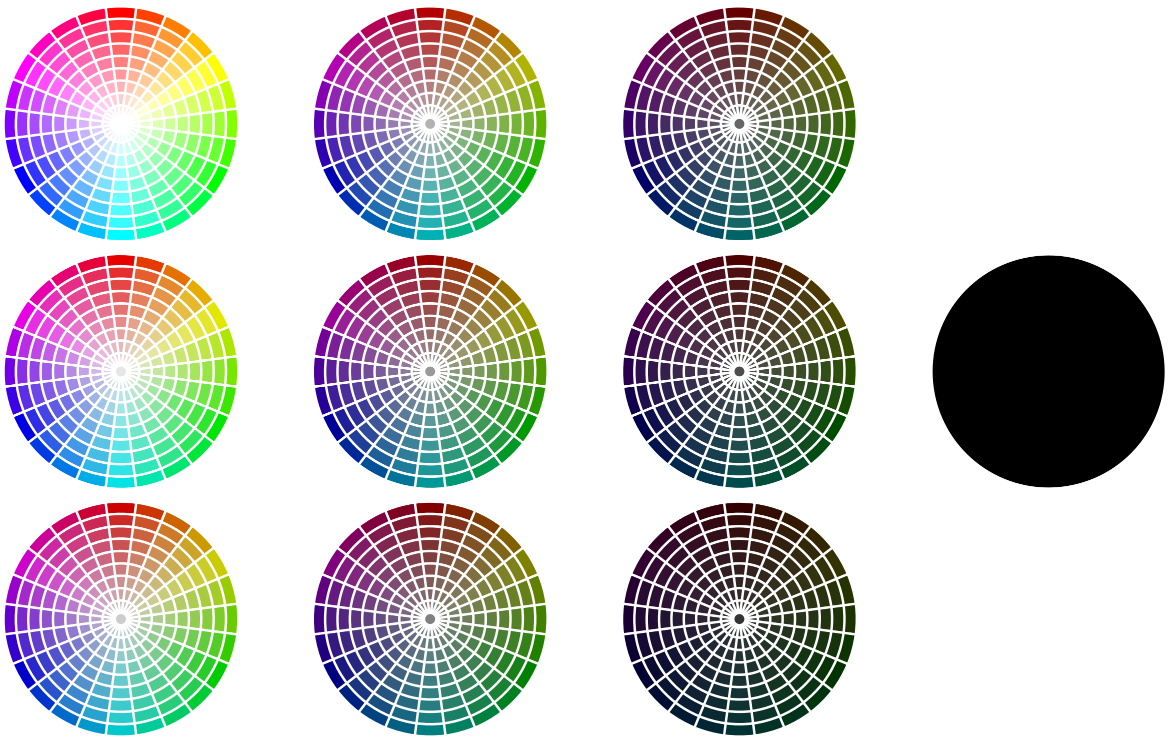


Fig. 6.27 The 2161 discrete colours used for HSV discretisation, arranged in colour wheels of equal Value with the angular change representing Hue and radial change representing Saturation. The final colour wheel is solid black, as all Hue and Saturation data is discarded once the Value drops below 0.1.

perceive at low *Value*, a further simplification is made that all colours below 10% *Value* are considered black. This produces a colourspace with 2161 discrete colours in total. Figure 6.27 shows these colours in multiple colour wheels seperated by value, while an image converted to show these discrete colours is shown in Figure 6.26.

6.4.3 Normalised-Relative Luminance

The purpose of including HSV data within the classification process is to differentiate between surface types which have similar texture profiles. However, this technique cannot be applied when classes have both similar texture and colour. Despite focusing on only a small number of classes typical of most aerodromes, it has been found that several of the classes used for this work cannot be reliably differentiated using the methods outlined above.

If class colouration was consistent, further data extraction within the HSV colourspace could provide a solution. Rather than only using the mean data, the distribution of pixel colours within each cluster could be compared to the expected distribution of colours for each class, providing additional information for comparison. Alternatively, a more advanced intermediary stage, such as SVM classification, could be used. However, due to environmental factors, the colour of each class can vary quite substantially. Attempting to include this variance in addition to differentiating between extremely similar classes is an extremely challenging classification problem.

Specifically for this work, the classes with similar colour and texture are asphalt, white surface markings and sky (during overcast weather). As the sky class has many other features which allow it to be correctly classified (covered in more detail in Section 6.5) the main requirement is a source of additional information to help differentiate aerodrome surface markings from asphalt.

As aerodrome surface markings are highly similar to road markings, it would seem sensible to review methods used within the related field of SDC. As with this work, colour and texture data is commonly used to find surface markings. Although more recent approaches make use of deep-learning (which conceals how the markings are actually detected) the majority of manually defined techniques primarily identify the markings based on image texture [106]. As with this work, there can be difficulty in confirming the identity of surface markings, especially at range. As such, confidence in markings is commonly increased through comparing the orientation of the markings to the overall road. Using the effects of perspective, it is possible to increase the classification confidence of markings which met the same vanishing point as the road boundaries [73]. However, as the boundaries of taxiways are runways are often denoted by surface markings themselves, this method is unsuitable for use in an aerodrome as parallel lines are not indicative of surface markings alone.

Returning to the image itself, the main difficulty in differentiating the classes is caused by both classes having low *Saturation* (i.e. they are grey). When the value of *Saturation* is low, the *Hue* component of colour loses all importance. Therefore, the ability to differentiate between colours depends on *Value* alone. As aforementioned, the *Value* channel does not represent brightness in the same way as human vision. Human eyes recognise specific frequencies of light more intensely than others, attributing different levels of brightness based on *Hue*. A combination of all frequencies (i.e. white light) should be recognisably brighter than any specific colour. By comparison, within the *Value* channel, data is simply the maximum distance from black (equivalent to the largest component of colour within the RGB colourspace).

Despite humans interpreting pure white as much brighter than pure blue (as shown

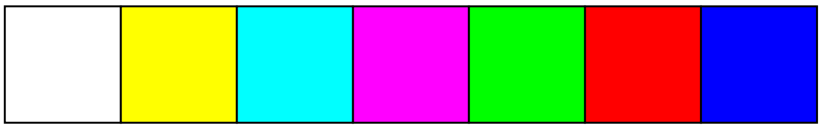


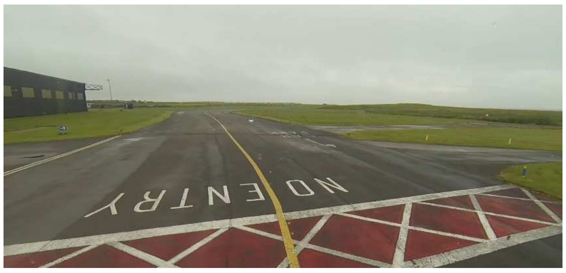
Fig. 6.28 Example of colours which share the same $Value$ within HSV colourspace, (the three colour primaries and white), indicating how $Value$ is visibly separate from the human interpretation of brightness.

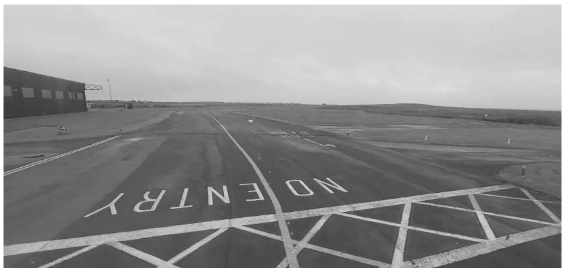
in Figure 6.28), this interpretation of brightness results in all primary colours, secondary colours and white sharing the same *Value*. As such, if the taxiway is not pure grey (e.g. a concrete taxiway), the numeric difference between the white surface markings and asphalt is greatly diminished. Therefore other interpretations of colour (i.e. colourspaces) are likely better at differentiating these classes than *Value*. A range of alternative methods of expressing brightness within images are shown in Fig. 6.29.

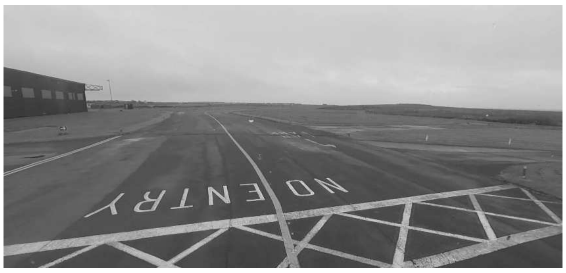
The closest alternative colour space to HSV is Hue-Saturation-Lightness (HSL), another cylindrical colour space which exchanges *Value* for *Lightness*. HSL was not selected for colour classification as the three colour channels are not uniformly distributed. That is, *Lightness* varies depending on *Saturation*, making it more difficult to use the colour channels individually. Despite using the same name, the *Saturation* channel in HSL is very different from that used in HSV. Shown in Fig. 6.29f, without data from the *Saturation* channel, much of the detail in the image has been lost. As such, despite the *Lightness* component always ranging between between pure white and black, it is too connected to *Saturation* to be used alone, with even less to differentiate white surface markings from asphalt than is present in when using *Value*.

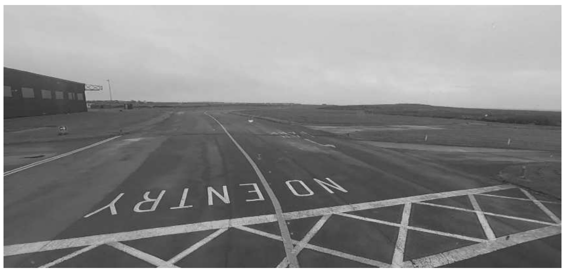
Returning to the RGB colourspace, component average intensity is the simplest method of producing a 'greyscale' image. However, it is not commonly used in computer vision as it fails to represent brightness as a human would interpret it. As surface markings are often worn, an extremely sensitive measure of brightness is required. For humans, brightness relates to the number of photons of certain wavelengths entering the pupil. As the area of the pupil and the number of photons define the brightness, it is photometrically the intensity of light incident on an area, commonly referred to as *Luminance*. In addition to absorption, *Luminance* can also be used to measure reflection. As the amount of light reflected from an object depends upon it's surface, accurate measures of *Luminance* can be used for material classification. Therefore, *Luminance* would be extremely useful as an additional data source for classification.

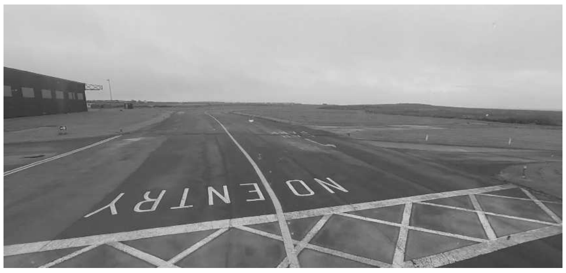
Unfortunately, when images are captured by a camera, automatic white-balancing and aperture effects prevent the actual *Luminance* value from being captured. Moreover,


(a) RGB colour image


(b) CIELAB L*


(c) Relative Luminance
$Y'_{709} = 0.21\text{R} + 0.72\text{G} + 0.07\text{B}$


(d) Component Average Intensity
$I = \frac{R}{3} + \frac{G}{3} + \frac{B}{3}$


(e) HSV - Value
$V = max(R, G, B)$

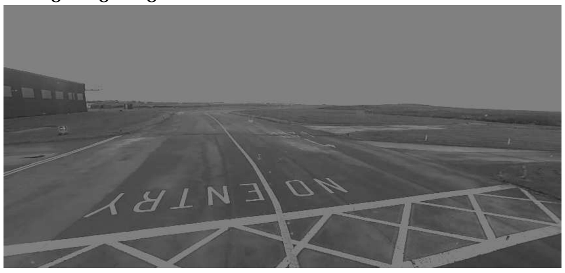

(f) HSL - Lightness
$L = \frac{1}{2}(max(R, G, B) + min(R, G, B))$

Fig. 6.29 Comparison of different methods of representing brightness

unless raw data is recorded, the storage format of digital images significantly alters the *Luminance* values. As most image formats are 8-bit, only 256 possible levels of brightness can be captured. Therefore, *Luminance* for images is commonly expressed relative to a known baseline and as such is known as *Relative Luminance*. In order to operate in a range of lighting conditions, the maximum and minimum *Relative Luminance* values within an image typically correlate to maximum and minimum *Luminance* values within the scene, rather than a fixed datum.

Further complicating the matter, *Relative Luminance* is not directly captured by most camera systems. For cameras which directly capture RGB (using the International Telecommunication Union (ITU) Regulation BT.709 primaries), *Relative Luminance* can be calculated from the linear RGB components. Eq. 6.1 reflects how humans interpret light, with green photons appearing over 10 times brighter than blue photons with the same intensity.

$$Y'_{709} = 0.2126R + 0.7152G + 0.0722B \tag{6.1}$$

However, during compression, many camera systems use *Gamma*-compression to reduce file storage size. Unless a direct conversion from (gamma-corrected) R'G'B' back to RGB is known, Eq. 6.1 will instead calculate *Luma*. Despite the visual similarity, this is another further step from actual *Luminance*, as the reference white level is also lost. However, for this work, the actions outlined below (which make use of either *Luma* or *Relative Luminance*) alter the data such that the difference between them is mostly unimportant. As such, this work uses the term *Relative Luminance* is used to refer to either. This is of additional benefit as it allows cameras systems which both do and do not use gamma-compression to make use of the same technique. Alternatively, the far more complex but perceptually accurate CIELAB colourspace could be used, as the *L** component is essential based on *Luma*, with some slight modification to correct for human vision at high chroma. However this has not been explored.

Cluster Normalisation

For human vision, the perceived intensity of colours is always influenced by other colours within the same scene. The classic "Checker Shadow Illusion" [5] demonstrates how human perception is based on relative colours, rather than absolute. In an aerodrome environment, under low light, the white surface markings may be less bright than plain asphalt in direct sunlight. It is only through comparison to other clusters that their bright-

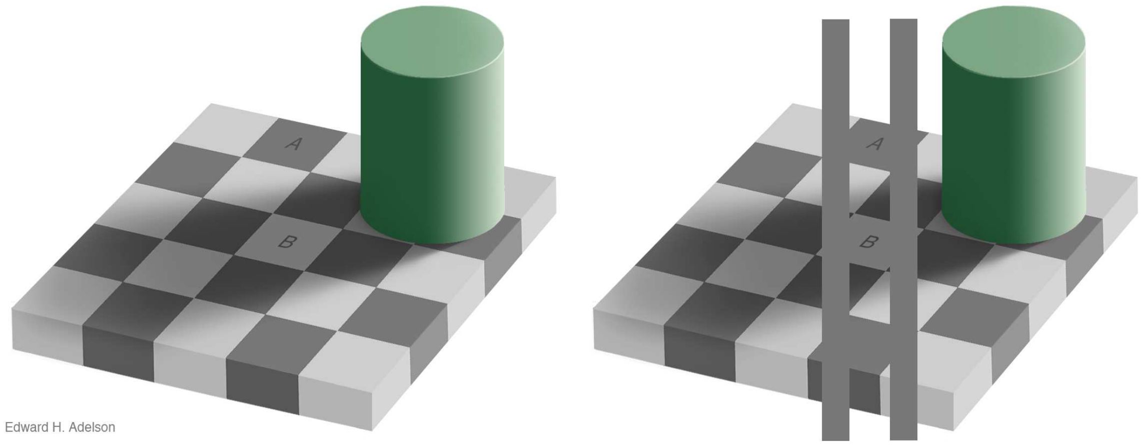


Fig. 6.30 The "Checker Shadow Illusion", intended to demonstrate how the perception of a single colour is relative to its surroundings.

ness becomes apparent. Therefore, comparison between pixel clusters is the easiest way to differentiate between them.

As the name suggests, *Relative Luminance* is already defined relative to the brightness of the clusters within the image. For outdoors scenes the range of *Luminance* is often greater than the sensor can capture. Aside from shaped materials which focus light towards the camera, sky pixels should always be brighter than ground pixels [49]. As such, the maximum *Luminance* value is usually defined by sky pixels, whereas the lowest *Relative Luminance* is usually a dark material, such as asphalt. Therefore, the brightest and darkest pixels are used to define the range for *Relative Luminance*.

As sky pixels are usually significantly brighter, the difference between asphalt and white surface markings is usually limited to only a small fraction of the total range of *Relative Luminance*, despite them being extremely dissimilar to the human eye. As both surface types are found on the ground, a better range of *Relative Luminance* can be found by separating ground and sky region(s) within the image. Although sky clusters can be difficult to identify on colour and texture alone (as overcast clouds can be similar in texture to examples of distant asphalt) additional information is readily available. Determining which clusters represent sky is made easier as the sky is constrained in terms of position (i.e. it always appear above the horizon line). For this work, an horizon detection algorithm is used and high-*Value* low-*Saturation* regions which appear wholly above the horizon are considered to be sky. (This information is also useful for classification and is covered in greater detail in Section 6.5).

Once the sky has been identified, this work suggests increasing the perceived difference between surface markings and asphalt by re-normalising the *Relative Luminance* values within the image, relative to the maximum *Relative Luminance* of any region on the ground. As the minimum value of *Relative Luminance* is often defined by asphalt and the brightest pixels on the ground as often white surface markings, this new output, which we refer to as NRL, then represents a fairly consistent measure of the brightness of pixels on the ground. A benefit of this approach is that as NRL strongly emphasises what appears bright to human eyes, it is also highly effective in detecting yellow surface markings. Relative luminance can be derived from RGB colourspace using Eq. 6.2, where R, G and B are the respective mean pixel values in each colour channel per cluster, and Y is the relative luminosity of each cluster.

$$Y_i = 0.2126\overline{R}_i + 0.7152\overline{G}_i + 0.0722\overline{B}_i \tag{6.2}$$

NRL can then be calculated using Eq. 6.3.

$$NRL_i = \frac{Y_i}{Y_{max}} \tag{6.3}$$

Figure 6.31 shows an example of NRL values for each cluster, clearly indicating the suitability of NRL for surface marking detection.

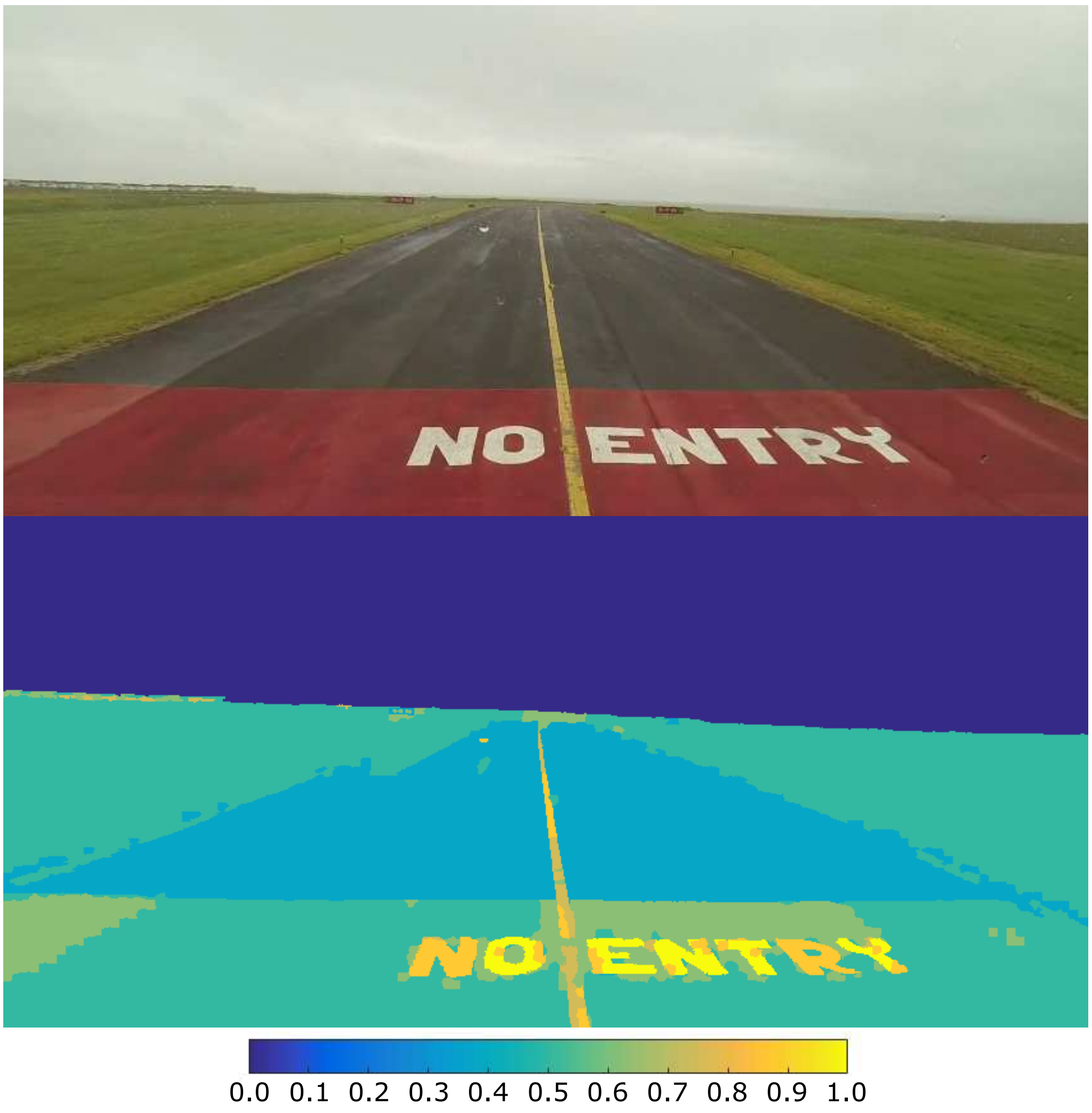


Fig. 6.31 Example of Normalised Relative Luminance, based on the original top image. It should be observed that despite being the brightest object in the original image, the sky region has been reduced to lowest likelihood of being classified as surface markings through the use of horizon detection. Instead, the surface markings within the image receive the highest NRL values, as intended. A more detailed evaluation of NRL imagery occurs in Chapter 8.

6.5 Horizon Line

In Section 6.4.3, the concept of NRL was established, in which luminance is re-normalised with respect to the ground based maximum, rather than the image-wide maximum. In order to achieve this, the terrestrial clusters in the image need to be identified. As this process occurs prior to classification, the identity of each cluster is unknown. Therefore, an independent method of reliably separating ground and sky is required. As the horizon line is, by definition, the point at which the earth's surface and the sky appear to meet, the sky and ground regions can be reliably categorised based on their position relative to the horizon line. However, as the horizon line itself is unknown, methods of determining the horizon line are required.

This section outlines multiple approaches to horizon line detection. In addition, the position of the horizon can be used to elicit additional information from the image. Therefore, additional uses of the horizon position for classification are also discussed.

6.5.1 Straight Lines and Lens Distortion

When viewed from close to the ground, the horizon should appear as a straight line between the ground and the sky. However, the effects of some camera lenses can warp the image such that the horizon line is curved. To ensure that the horizon line is straight, the images used for this work have undergone correction to remove the effects of lens distortion. This was simply achieved using software provided by the camera manufacturer, applied immediately after capture. As such, all the images used for this work were already free from radial distortion and the horizon is assumed to be a straight line.

As all radial lens distortion can be corrected through software, it is assumed that any future ATS will apply distortion correction prior to image processing. As methods of correcting camera distortion are well established, they have not been explored in detail here.

6.5.2 Horizon Position Estimation

Although many UAS are fitted with rotating cameras for airborne observation, the assumption for this work is that the camera is mounted on the aircraft in a fixed position, without the ability to pan, tilt or zoom. As such, the position of the camera with respect to the environment is dependant entirely on the movement and position of the aircraft.

As the aircraft is moving across flat ground and the height and mounting angle of the camera are known, the simplest method would be to assume a static horizon line based on trigonometry. Similar work done in [111] estimates the location of features captured by a camera for use with SDC. In this case, the work assumes a fixed angle between the car and the ground. For road vehicles, this angle can be considered fairly constant, which would allow ground and sky to be reliable separated using a static horizon line.

However, aircraft have comparatively flexible suspension (as the undercarriage is designed to absorbed impacts during landing) and a high mounted thrust line which causes aircraft to pitch and bounce as they taxi. Although the height of the camera will be subject to suspension flex as the UAS taxis, the variance is expected to be small enough so as to have limited effect on the output. Of greater influence is the effect of pitch and roll, where a small change can effect the accuracy of the output immensely. Therefore, a static horizon line is not sufficient and the position of the horizon will need to be actively estimated in order to produce an accurate result.

A non-visual solution would be to continue to use to height and angle of the camera, but to alter the position based on inertial feedback. As INS is present on all unmanned aircraft, both the attitude and angular rate of the aircraft can be calculated. From the inertially derived angle of the aircraft travelling over flat ground, it is a matter of trigonometry to determine the position of the horizon within the image. However, despite aircraft INS being highly accurate, it is not attached to the camera. Any flex between the camera mount and the aircraft can cause slight variance which produces a different angular result. In particular, the high frequency vibrations typical of ground vehicles tend to affect smaller rigid objects more. During testing it was found that although inertially derived horizon position was consistent close, it was often not precisely aligned with the horizon.

Furthermore, the discrepancy between when the image is taken and when the Inertial Measurement Unit (IMU) data is obtained can make fusing inertial and visual information very difficult. Any delay between the rates at which sensors operate can cause massive discrepancies when the vehicle is at speed, as described in [80]. The researchers at Oxford University Robotics Research Group (OURRG) experienced such an issue when combining data from both laser range finders and cameras. To overcome this, OURRG put a great deal of work into software that could identify the phase shift between the apparatus, factoring in hundreds of variables; such as temperature, humidity and electrical draw of the system. The need to develop a similar system for this work would make the project infeasible.

As the use of this data is in interpreting the image, it is sensible to consider methods which attempt to determine the attitude of the aircraft from the image itself. Compared to inertial estimates, visual estimation is far more processor intensive. However, if the horizon estimate is drawn from the same image as the clusters, there is no risk of mismatch.

Ground based horizon estimation methods are also commonly found in SDC research, such as [115]. The work in [115] makes use of 'vanishing points' to estimate the horizon. On a straight road, the parallel edges of the lanes will appear to converge over distance. As these features are ground based, the convergence point should be on the horizon line, even if the horizon itself is not visible. Whilst this has been shown to be highly effective for road traffic, it requires many features to be present in every frame to correctly estimate the horizon position. Often aircraft will leave taxiways for apron areas where there are no suitable markings on which to apply this technique. In addition, taxiways do not have lanes, so the technique would have to use the edges of the taxiway and the taxiway centreline instead. This is an issue as not all taxiways have borders, and where borders are present they are not always parallel. With so many limitations, this technique is considered to difficult to reliably implement.

The simplest horizon detection methods work on the assumption that ground and sky pixels differ dramatically, even on overcast days. The horizon line can be extracted by examining the edge between the two regions. In aviation, methods such as the dark channel approach (similar to that outlined in [124]) are already used extensively on Micro Unmanned Aircraft System (MUAS) and have been found to be highly effective at providing an additional source of attitude data to compliment the output from the IMU.

The obvious limitation of this approach is that the system is compromised by the presence of any foreground object, such as buildings or other vehicles. As these objects are not classed as sky, they result in the horizon line shifting 'upwards', making the result inaccurate. As such, a method which uses both inertial and visual data has been implemented. Despite the image and sensor data always being slightly offset, an inertial-derived horizon position should still be fairly accurate. As such, for this work a second horizon line is also calculated based on the roll and pitch angles provided by the aircraft IMU.

6.5.3 Dark Channel Horizon Estimation

The visual horizon detection process uses a dark channel method, as described in [125]. This works on the assumption that ground pixels will always have darker components when compared to sky pixels. Working in RGB, each pixel in the original image is replaced by the lowest value in either R,G or B within the local neighbourhood. This is achieved

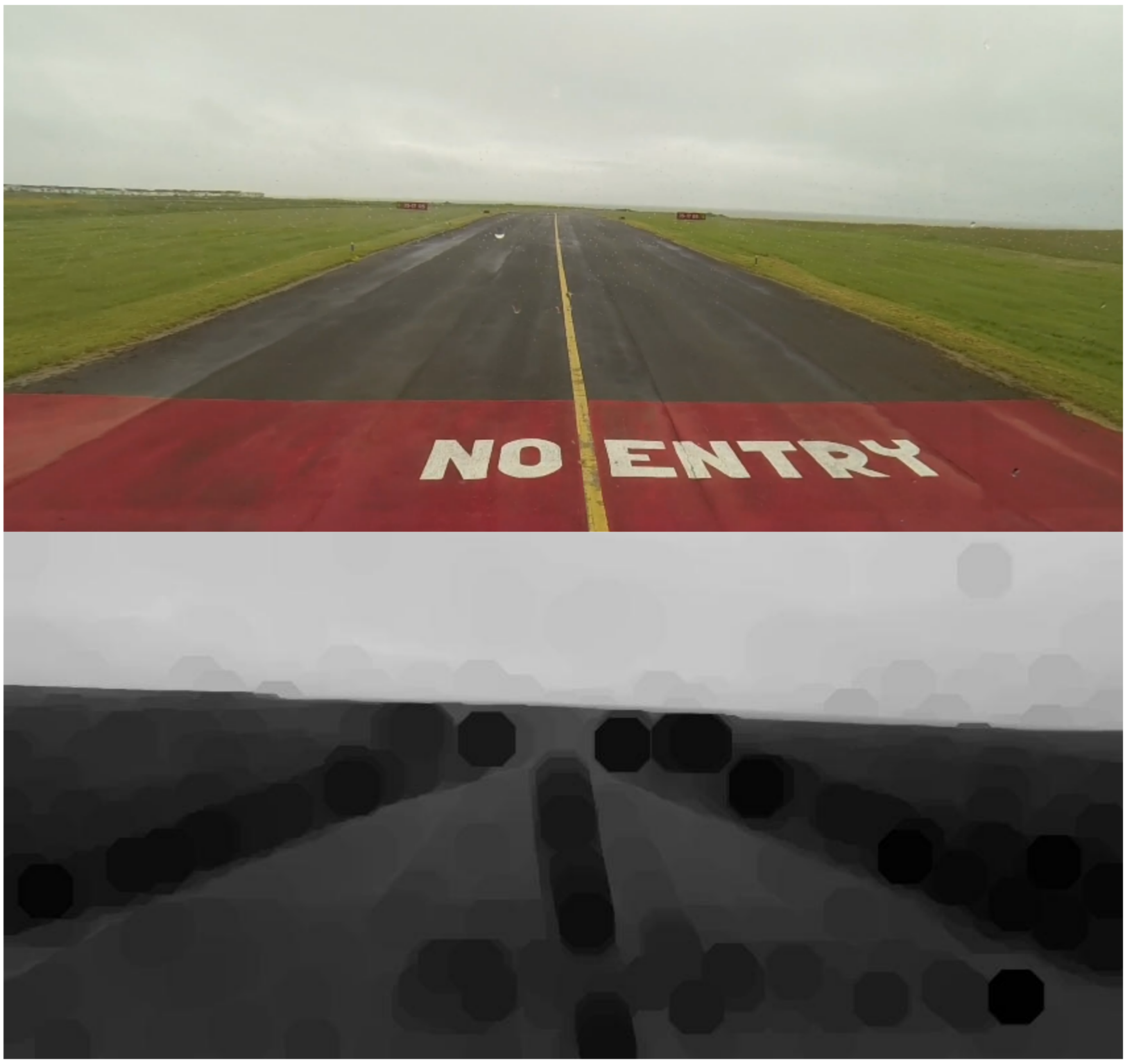


Fig. 6.32 Example of Dark Channel Extraction, based on the original top image. The distinct difference between sky and ground pixels allows the horizon line to be easily determined.

by first replacing each pixel in the image matrix with the minimal value along the third dimension, before applying a morphological erosion. This produces a dark channel image, as shown in Figure 6.32. From this example the split between ground and sky pixel is clear, allowing for the horizon line to be extracted.

For images such as the one shown in Figure 6.32, horizon extraction is simple. However, the horizon line is often partly obscured, preventing the use of straightforward line estimation techniques, such as regressive least squares. Instead, a techniques which pivots potential horizon lines around the lowest sky pixels in the image is used. The horizon line detection process is as follows:

- An initial horizon estimate is created based on camera and aircraft geometry
- The inertial sensor readings are used to correct the horizon estimate based on the attitude of the UAS.
- Visual horizon extraction occurs:
  - A dark channel representation of the image is created
  - A threshold is applied to create a binary image of sky/ground pixels.
  - Connected component analysis is applied to the 'sky' region.
  - A potential horizon line then pivots around the lowest sky pixels(s) in the image. The angle which minimises the amount of ground pixels above the line whilst never crossing into the sky region is then taken as the horizon.
- The horizon line from each captured frame is used as the initial estimate for the next, ensuring the horizon line is consistent over time.

As ground objects can completely obscure the horizon, the dark channel derived horizon line cannot differ from the attitude derived horizon line too greatly. If it does so, the assumption is made that the UAS is facing a large object (such as a building) and therefore a visually derived horizon line will not be accurate. Instead, the inertial estimate will be used.

6.5.4 Additional Uses of Horizon Line

In addition to separating the image into ground/sky clusters for NRL extraction, the horizon line can be used for several other applications. Used directly, the horizon line can reduce the amount of the image in which classification is required. As image processing algorithms can be highly computationally intensive, a common approach is to restrict the image area to only that where the feature will lie, commonly referred to as a ROI. By bisecting the image with the horizon, only the bottom half needs to be processed when seeking features only found on the ground (e.g. taxiway markings). Using ground feature detection algorithm on the region above the horizon line is simply a waste of processing power.

In addition, the position of the horizon line can also be used to determine the attitude of the camera. Just as the inertial attitude of the aircraft can be used to provide an

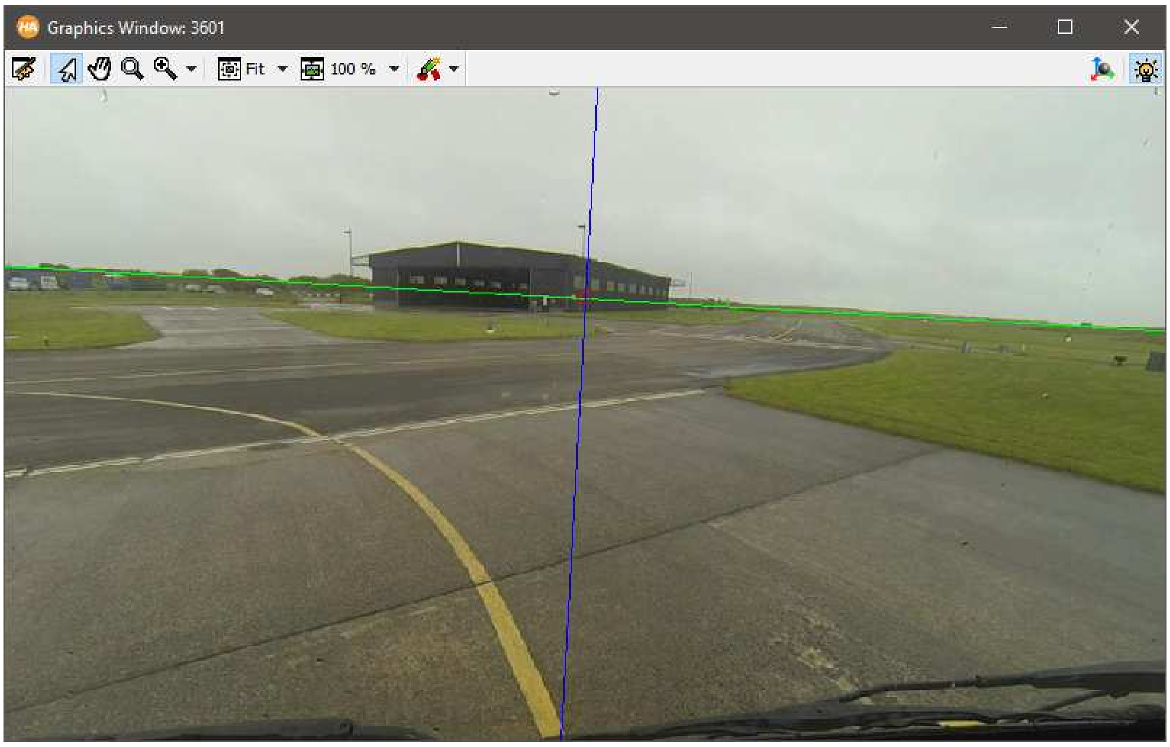


Fig. 6.33 Example of Visual Horizon detection, implemented within MVTec Halcon. This figure shows the extracted horizon line, in addition to the true vertical relative to the camera, allowing both pitch and yaw to be calculated.

horizon line estimate, the opposite is also true; with the angle of the horizon line within representing the camera's roll, whilst the vertical position of the horizon line within the image indicative of pitch. Finally, the position of a cluster relative to the horizon can also be used directly as an indicator of class, which is discussed further in the next section.

6.6 Spatial Data

In addition to the contents of each cluster, the position of the cluster within an image can also be used to help with classification. Four types of spatial information are considered here:

- Horizon relative
- Immediate foreground
- Context from Neighbours
- Aerodrome map

6.6.1 Horizon Relative Classification Data

For use in data extraction, the position of a cluster relative to the horizon can be used directly as an indicator of class. Continuing to use the method first suggested by Eaton and Coombes (2016) [24], the position of each cluster relative to the horizon can be considered a logical check, limiting potential classes during classification. Each cluster can be assigned one of three possible states, based on it's position relative to the horizon:

1. *Above* - Clusters entirely above the horizon are either airborne or the object represents sky. In either case, such clusters are not relevant for ground operations and the probability of being a collision risk is low.

2. *Below* - Clusters which are wholly below the horizon line can be considered on the ground and require the other forms of data to aid in classification

3. *Intercept* - Clusters which extend across the horizon line and have a significant portion either side of the horizon are objects that extend up from the ground and therefore represents a possible collision risk.

As the bottom of the sky region was used as the basis for horizon line, practically all ground clusters which border the sky will be classed as intercept. As such, a large cluster (such as the taxiway) which extends only a single pixel over the horizon line would be incorrectly considered as an intercept class. To avoid this, the horizon line is treated as a 20 pixel tall band rather than a pixel tall line. As shown in Figure 6.34, horizon detection alone is capable of detecting potential collision risks, when a larger cluster extends across the horizon line.

6.6.2 Immediate Foreground Classification

Similar to logical check used in Section 6.6.1, an additional assumption can be made regarding the immediate foreground in front of the camera (i.e. the lower-most region in the image).

Assuming that the UAS begins navigating from a position of safety (i.e. on top of a navigable surface such as asphalt) if the ATS does not err in its navigation, the UAS should still be positioned on a navigable surface. As this surface extends throughout the clusters, the closest collision risk possible occurs at the uppermost edges of these bottom regions. If the system ever fails to identify the surface it is presently on top of (perhaps due to shadows or water reflections) the safest method of continuing would be

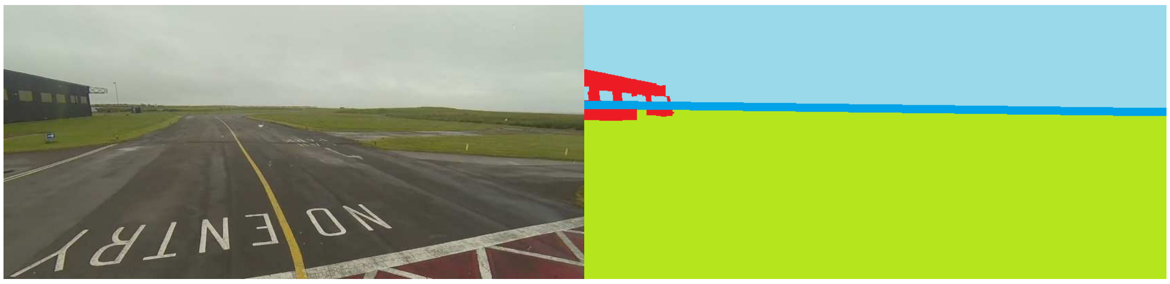


Fig. 6.34 Horizon based cluster classification. In the original image to the left, the only object which extends to each side of the horizon line is the building. This result is clearly reflected the horizon based classification on the right, where only the building cluster has been assigned the *Intercept* class. Although the windows in the building are not classified as intercept, as collision avoidance should always estimate to the bottom of objects, this collision risk has been potentially detection through horizon line extraction alone.

to remain within the bounds of whichever region it is currently sat upon. As such, simply identifying the closest cluster to the aircraft provides a method of potential collision risk detection, entirely without classification. This assumption is used in [97] as part of an image process strategy which allows a small robot to perform visual collision avoidance without classification.

However, as the edges of the lowermost clusters can only be used to suggest the limits of known safety, the clusters beyond are no more likely to represent collision risks. As the segmentation process aims for an over-segmentation, and taxiways have surface markings, the likelihood of the lowermost clusters meeting directly with collision risks is fairly low. As such, the most appropriate use of this information would be that each cluster which is present at the bottom of the image should receive a unique data type, signifying it is less likely to be a collision risk.

Furthermore, as the camera is mounted straight forward and mounted high on the airframe, the minimum distance at which the ground can be seen is over 10 metres. As such, the possibility exists that a large collision risk close to the camera could prevent the ground itself from being seen. If the bottom of the image was simpled labelled as safe, this would likely result in collision.

Despite successful application of this approach in [97] for a small robot, due to the different mounting position of the camera, this approach will not be pursued in this work.

6.6.3 Classification through Neighbour Cluster Context

Aside from the absolute position of the cluster within each captured image, additional information can be extracted from the local position of a cluster relative to its neighbours. When a human observes a scene, each object is identified using not only its own appearance, but also the context of its surroundings. For example, a distant brightly coloured object might be hard to classify alone, but if it were located on a road it would most likely be a car. For terrain classification, contextual clues such as *'grass is usually alongside taxiway'* and *'taxiway centre lines should be bordered either side by asphalt'* are useful.

As the relationship between clusters depends on adjacency, the spatial relationship between regions can be modelled through the use of probabilistic graphical models, e.g. Conditional Random Field (CRF) or Markov Random Field (MRF).Previous works have already included contextual information within the classification process; the method described in [9] uses the edges within a graph to represent the relationships between neighbouring clusters. Although a graph based classifier is required to incorporate this spatial information directly, the relationships between clusters could instead be amended in a subsequent graphical stage, just as the colour distance between superpixels was used to form clusters after the initial over-segmentation in Section 6.1.2.

Although the inclusion of spatial data has been shown to improve classification of known classes, as with other classifiers the use of prior knowledge can bias the classifier towards expected conditions. For example, the expectation that a taxiways are bordered by grass may cause difficulties where the taxiway was bordered by a different material, such as an asphalt road for ground vehicles. When working in a controlled environment with known terrain features, this form of contextual classification may actually hamper performance; any misclassified terrain clusters make it more difficult to match the UAS location to a map.

For collision risk detection, as all objects on taxiways pose collision risks, there is little context that can be used. Any object which is not clearly a terrain feature is already considered a potential risk. As such, the inclusion of graph-based contextual classification is not useful for such a small number of classes and will not be included within the classifier. **As introducing data from neighbouring clusters will bias unknown regions towards known classes through expectancy, the contents of neighbouring regions are not used to help identify clusters in this work.**

6.6.4 Map Data

Similar to how context for each cluster can be obtained from neighbouring clusters, the aerodrome map can also be used as additional source of information. As GPS data should be available, the position and heading of the UAS is known. Using a model of the camera and representing the aerodrome map as a flat plane within 3D space, a image of the aerodrome terrain from the perspective of the camera can be created. This allows an additional source of classification data for each cluster.

Several attempts to incorporate this data within the classifier were made for this work. However, many issues were found. Certain issues were able to be resolved, such as the lack of building occlusion when using a 2D map. (Despite being present on the aerodrome map, when projected into 3D space the buildings were simply 2D terrain features. As such, when the clusters which represented the building were aligned with the data in the map, they were instead associated with the objects behind the building. Therefore, all buildings and known collision risks on the map were given 3D height in order to create occlusion).

However, occlusion in the opposite direction was much harder to resolve. When an unknown collision risk appeared in the original image, the collision risk object would receive an artificially increased score of belonging to which ever class would be visible to the camera, had the collision obstacle not been present. As this is typically a terrain, the chance of an unknown risk being detected is reduced.

Although methods of overcoming this issue were tested, (such as comparing the shape of clusters to the known terrain features and only assigning a classification estimate if the shapes were similar) the biggest issue was that of GPS position accuracy. During testing, GPS position was only accurate to within a few meters, resulting in the virtual camera position producing a distinctly different image, as shown in Figure 6.35. As such, the terrain features being used to provide an additional classification estimate could not be relied upon to be correct.

The UAS localisation has already been identified as a potential area which could benefit from visual data, by using the visual obtained data to verify the externally derived position data for the UAS. However, if the visually extracted information has already been influenced by the GPS derived position of the UAS, the biasing towards the expected position would limits the methods effectiveness for localisation correction. As such, no spatial information is included within the classifier - either from the aerodrome map or neighbouring clusters within the image.

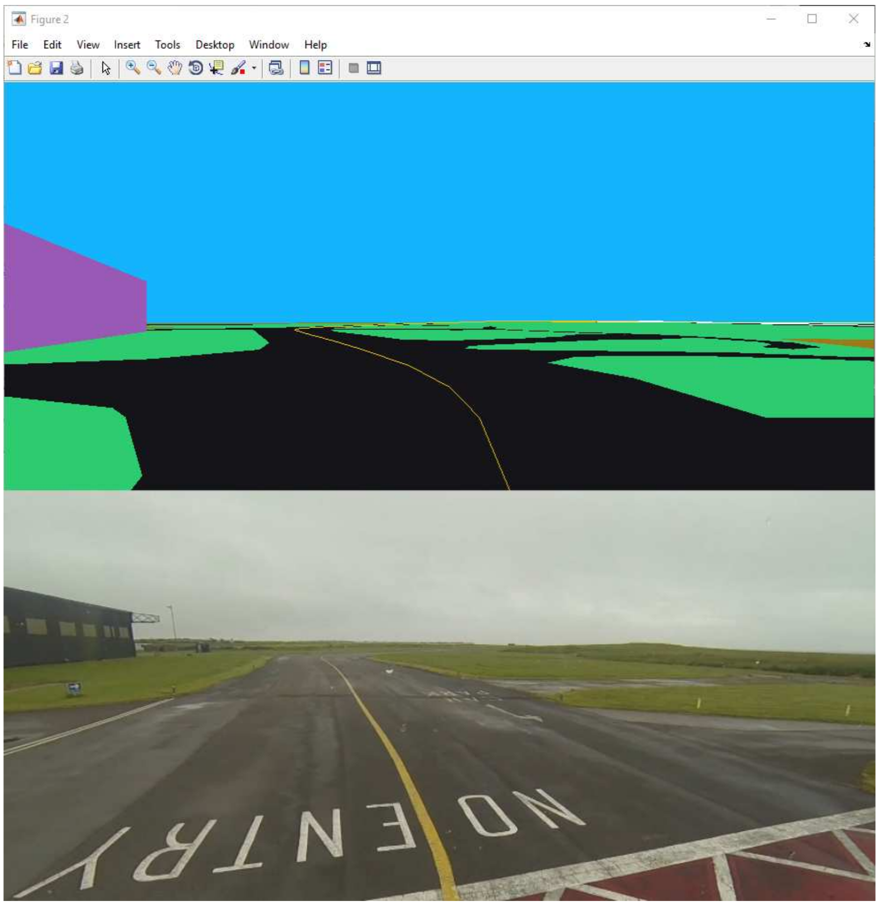


Fig. 6.35 Example of the difference between simulated camera position and actual image from camera, based on test data at Walney Airport

6.7 Summary and Conclusion

After reviewing many potential methods of achieving image segmentation in Chapter 5, this chapter has focussed on the actual implementation used. In the prior chapter it was established that the ideal result would be ‘object specific segmentation’, in which each cluster would not only represent a single object, but would also contain as much of that object as possible. This approach was intended to capture more information per cluster, strengthening the confidence in the final classification.

The selected approach consists of two stages; a superpixel-segmentation process which creates an initial over segmentation and reachability re-clustering to achieve as close to ‘object specific segmentation’ as possible. SLIC was once again selected for the initial over-segmentation as the method produces superpixels which are an ideal size for reachability clustering. In addition, using both colour and spatial distance to cluster ensures that even subtle edges in the original image are reflected in the position of the superpixel borders. This minimises the risk that any superpixel will contain more than one category of object, whilst also ensuring that superpixel boundaries conform to the boundaries of objects within the image.

Although the DBSCAN algorithm was presented in the previous chapter and has been used in prior work [33], a novel form of reachability clustering has been introduced here. This graph-based process was found to be more computationally efficient, whilst also restricting reachability to only neighbouring superpixels, ensuring that superpixels must be both adjacent and similar for the cluster to expand. After comparing aerodrome images segmented both manually and using the automated approach, it was empirically determined that the chosen method was nearly as capable as a human, with only minor discrepancies under aerodrome conditions.

Although object specific segmentation is not always achieved, this is difficult in environments where objects can have varying colours per object. Furthermore, as under-segmention is potentially very dangerous (with potential collision risks being ‘merged’ into larger terrain clusters and therefore not identified) the re-clustering is undertaken to ensure that an over-segmentation is always more likely to remain. Despite this change, the algorithm produced large, single-class clusters suitable for data extraction and classification.

Following the successful implementation of the segmentation stage, ’Data Extraction’ is required to allow the classifier to determine the contents of each cluster. For some types of data, the extraction approach includes training a pre-classifier such that the output is an estimate of class. In comparison, other data can already be used to provide an estimate of class, based on human interpretation. To use both types of data together, the data extraction process collects useful information, all of which is then passed to a subsequent stage for data fusion and final classification.

As different forms of image data have different strengths and weaknesses, the adopted approach begins with Texture data, as this allows a good estimate of class without any additional information. Two different forms of feature descriptor were chosen (MR8 filter bank and LBP), due to the different scales in which they operate. As MR8 data has many millions of potential outputs, a closest-texton based representation is used for easy comparison to instances of known classes. Actual comparison is achieved using a pre-classifier, which provides an full estimate of class. From a comparison of several techniques, an SVM approach which uses a BDT to minimise the processing cost has been chosen.

Although texture data provides an estimate of class directly, it is shown to be only around 90% accurate. Therefore, additional information can be passed to the classifier to improve this result. This additional data includes colour data, which is used to help differentiate between classes with similar textures. To minimise the effects of variable lighting conditions, the HSV colourspace is used due to having separate chroma and luma components.

As surface markings are of particular interest for localisation, an additional method of detecting highly reflective surface clusters has been introduced, known as NRL. Unlike luma, which varies with the lighting conditions, NRL is relative to other ground clusters within the same image. This ensures that NRL should highlight surface markings until it is too dark for the camera to function. Finally, potential collision risk estimation is extracted, based solely on cluster position relative to the horizon line. Clusters which intercept the horizon can be considered less likely to be terrain features and more likely to be an object which poses a risk to the UAS.

As classification accuracy typically dependant on the amount of information provided, nearly all forms of data that could be extracted have been incorporated. However, data which could artificially bias the classifier have not. Therefore, data which is not directly drawn from a cluster, such as map data and contextual information based on the classification of neighbouring clusters, has not been used.

Chapter 7

Classification

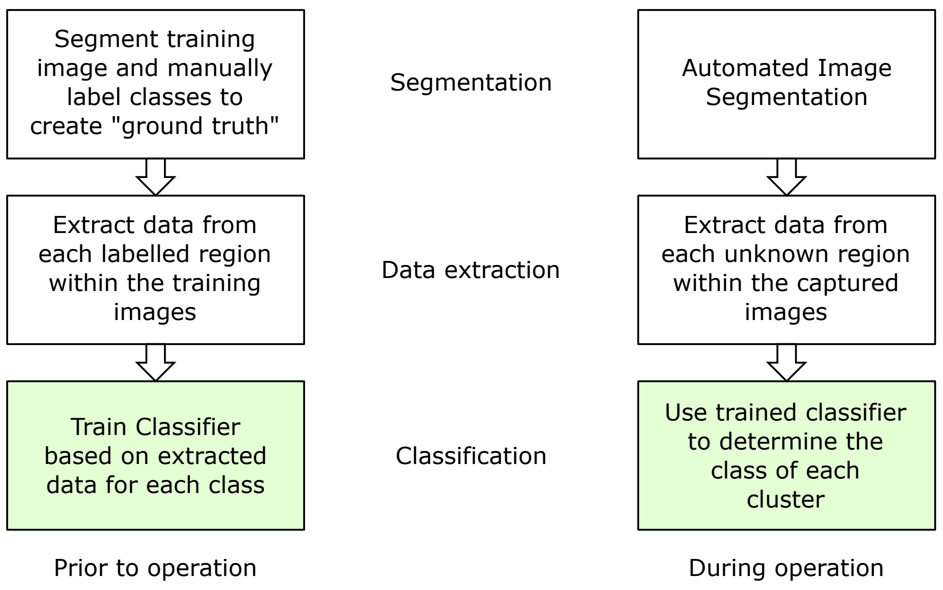

Segment training image and manually label classes to create "ground truth"
Segmentation
Automated Image Segmentation
Extract data from each labelled region within the training images
Data extraction
Extract data from each unknown region within the captured images
Train Classifier based on extracted data for each class
Classification
Use trained classifier to determine the class of each cluster
Prior to operation
During operation

Fig. 7.1 Stages of the semantic segmentation process, showing how the classification process is divided into a prior 'Training' stage as well as the 'Classifying' stage during operation.

Following on from data extraction, this chapter covers the final stage of semantic segmentation, in which the clusters are classified. Returning to the three stage process shown in Figure 7.1, it can be seen that classification is achieved in two parts; training the classifier, and using that classifier to determine the class of each cluster. However, before this is possible, the classifier itself must be selected. Despite having covered SVM classification in Chapter 6, the classifier required for the final result is quite different. This is due to the different types of information that must be incorporated within the final output. Essentially, the final classifier must be capable of working with three levels of information:

1. Raw data, such as discretised HSV
2. Indicators of class, such as horizon intercept category
3. Estimates of class, such as the Texture classification results

In addition to combining data from different sources, the classifier must also be capable of incorporating the suitability of each type of data, in reference to the output class. For example, if the texture classification result suggests that a cluster is asphalt, the classifier must include the understanding that painted surface markings are also highly likely, and that colour information should be used to differentiate. As such, a classifier which can interpret highly diverse data within context is required. Beyond simply relying on certain sources more than others, functioning without certain sources of data is also important. Although texture data is used as the primary source of data, some surfaces may provide insufficient data for a classification. Despite the likelihood that this represents an unknown, the rest of the data sources would still be available and therefore an estimate should still be produced.

Finally, the classification process presented in this chapter is intended to only work with visually obtained data. As stated in Chapter 6, fusion with other types of non-visual data tends to bias the visual results towards expected classes. Therefore, even though fusion with other sources of data could be achieved simultaneously with visual classification, it is considered more appropriate to first obtain a final visual classification result for each cluster. This result can then be compared with other data sources, in which ever manner is most appropriate.

This chapter contains the contribution of a representative probability calculation for texton classification data, specifically for converting a BDT based SVM classification into a probabilistic output, covered in Section 7.3.1.

7.1 Previous Work

Throughout much of this thesis, the previous work of Eaton (2015) [33] has been referenced, as many of the selected data extraction techniques were present in that paper, albeit in early iterations. However, the final classification approach used by Eaton (2015) has not been brought into this work, as the results proved to be unsatisfactory. The following section explores why these results were insufficient, and the progress made in improving upon them.

7.1.1 Attribute-Value System

As previously established in Chapter 6, Eaton (2015) [33] used both texture and discrete colour as data for classification. For texture, rather than provide a direct class estimate, the Bhattacharyya coefficient was used to compare each cluster to examples of each known class. The average Bhattacharyya distance between the cluster and class was then used as a measure of similarity. (Despite working well, this method was not pursued due to being extremely computationally inefficient, with more efficient distance metrics producing similar results). Data fusion between colour and texture was achieved through an Attribute-Value System (AVS). By manually defining the likelihood of each class being a specific colour, the Bhattacharyya distance was artificially multiplied to increase or decrease the probability that that class would be the closest. For example, the distance between a cluster and the 'yellow paint' class would have been artificially reduced if the cluster colour was yellow.

Including colour data within the classification process produced a more accurate result than texture alone, with the average correct classification rate rising from 65.6% to 74.5% per class, and 86.2% to 89.3% per cluster. However, as the approach fused data in a simplistic and naïve fashion, the colour data did not consistently improve each class. As shown in Table 7.1, the success varied depending upon the surface type being classified. For asphalt, the successful classification result was over 90%. However, classes which could not be easily defined by colour (such as buildings) only had around a 50% successful classification rate.

From the results of Eaton (2015), several conclusions could be drawn. One conclusion (for which a solution has already been implemented in this work) was that under-segmentation would nearly always produce a misclassification. In Eaton (2015), the segmentation approach aimed to produce as close to "single-object segmentation" as possible. To achieve this, reachability clustering was performed with a high threshold, which would occasionally group super pixels belonging to multiple classes. Despite only having a small proportion of each cluster from another class, it was found that even small errors in cluster borders could produce misclassification (or rather, clusters which contain more than one class cannot be classified correctly). As such, all subsequent work has

Manual	Automatic Segmentation Output as Percentage of Manually Segmented Classes								
Classification	Unknown	Asphalt	Grass	Sky	White paint	Trees/plants	Buildings	Yellow paint	Vehicle
Asphalt	0.02	**94.83**	0.62	0.23	0.65	0.42	1.74	0.2	1.28
Grass	0.02	24.41	**68.07**	0.01	0.09	2.79	0.8	0.9	2.91
Sky	0.11	0.11	0.01	**97.87**	0.15	0.16	1.32	0.18	0.1
White paint	0.04	18.16	0.89	0.4	**66.16**	0.37	8.35	3.1	2.52
Trees/plants	0.18	5.49	6	0.86	0.37	**61.91**	16.12	0.12	8.96
Buildings	2.51	2.26	5.97	3.04	3.47	12.12	**55.52**	0.08	15.02
Yellow paint	0.13	13.8	5.54	0.06	3.46	0.13	0.72	**74.93**	1.24
Vehicle	3.58	2.03	2.32	2.08	6.32	19.19	9.4	2.82	**52.27**

Table 7.1 Per-pixel classification results, using Texture and colour data with "naïve" AVS classifier for ten aerodrome images. The diagonal results highlighted in bold represent successful classification, in line with the manual classification of that cluster.

consistently aimed to produce a slight over-segmentation, so as to prevent multi-class clusters from being created.

Returning to the results of Table 7.1, the worst performing classes were those which appeared least regularly but had high internal class variance; i.e. Buildings, vehicles and plants/trees. As the data set has very few trees (due to the aerodrome used for testing having very few trees nearby) it was decided to remove this class entirely. Similarly, the vehicle class had limited data, as only a few other aircraft were present during testing. Attempting to use a small dataset to train a classifier should over-fit the class to this particular vehicle. However, the limited data available and the similarity in colour of the vehicle to nearby building produced poor results. Therefore, the decision was made to revert to detecting vehicles as generic collision risks. As buildings are static, their detection is not vital as they can be avoided from using map data and localisation. Therefore, map data is used as part of the guidance system, but is used as a source of information to aid visual classification. Despite this, the building class was retained during testing to determine if a new classification method could improve the result, even with a limited and highly variable data set.

Finally, for classification it was identified that a more coetmplex relationship between surface-type and data-type confidence was required. Due to the wide variety of data types, a probabilistic approach to data fusion was pursued; specifically using a Probabilistic Graphical Model (PGM).

7.1.2 Probabilistic Graphical Models

When selecting a classification approach for this work, PGMs were considered highly appropriate. PGMs are data fusion techniques which allow information from multiple sources to be combined to provide a final classification result. One of the biggest challenges in data fusion is combining multiple types of information, when the knowledge is originally expressed in a function unique to each data type. (e.g. colour is provided as a discrete section of colourspace, whereas texture is provided as an estimate of class). As the name suggests, PGMs use probability in order to fuse multiple sources of data together. The overall probability of a cluster being a certain class is then inferred from all sources of data, increasing the overall classification accuracy compared to what could be achieved with each individual source.

In addition, by representing the relationship between sources of data as a graph, the conditional relationship between sources of data can also be modelled, allowing confidence in data source to directly affect the classification result. As such, PGM offer significant advantages when compared to the AVS data fusion approach used in Eaton (2015) [33].

Within machine vision, PGM are commonly used for semantic segmentation, with MRFs, CRFs and BN being some of the most popular implementations. Both MRF and CRF use undirected graphs, in which the classification result can be altered multiple times during classification. For example, classification of a superpixel could be undertaken in isolation to produce a final result. These final results can then be compared to multiple local superpixels, with the possibility of altering the final result if another class is then more likely. As this form of classification can be applied at the pixel level, both MRF and CRF are highly appropriate for classifier led segmentation.

BN are also commonly used for classifier-led-segmentation, (such as [67] and [17]) however they use a Directed Acyclic Graph (DAG), in which information flow within the network always proceeds towards the final classification result. This makes BN more efficient, however they cannot incorporate information which depends on feedback from further down the classification path.

As discussed in Chapter 5, for this work segmentation has already been achieved in a separate stage. With cluster borders remaining static and no-relationship between neighbouring clusters included in the extracted data, there is no need to alter the classification result multiple times. As such, PGM for classifying pre-segmented images can use a simpler 'directed graph'. Work by Zhang et al. (2011) [126] discusses multiple PGM methods which could be used to classify superpixels. As PGM with undirected graphs are typically inefficient, methods which use directed graphs are more common for pure image classification. As such, Zhang et al (2011) conclude that without the need to perform the initial segmentation, a BN approach is both far more efficient, whilst allowing data to be incorporated more easily.

Compared to the original AVS data fusion approach, a BN should not only improve the classification performance but also provide a solution more robust to changing conditions. As such, a BN approach was adopted for data fusion and classification for all subsequent work. Since Eaton (2015), the use of the BN for cluster classification has been covered in two additional conference papers, both by *Coombes and Eaton* in 2016; [24] introduced a BN for data fusion of colour and texture data to provide a better result, followed by [25], which improved upon this result by introducing additional data types and soft (probabilistic) evidence.

This culminated in a journal paper (Coombes and Eaton (2017) [26]) in which a final BN implementation was established and an in depth review of the results was undertaken. As further work on the ATS focussed on the collision avoidance and localisation elements, the classification method has not been improved upon since this paper. As such, the results presented below are in line with [26].

7.2 Bayesian Networks

A Bayesian network is a PGM that represents a set of random variables and their conditional dependencies. The primary variables for a BN are those which can be directly observed; commonly referred to as ‘evidence’. Each of the independent data extraction techniques described in Chapter 6 as input as ‘evidence’ into the graph. Secondary variables can then be inferred from the ‘evidence’, using the conditional dependencies that link each variable.

An assumption is made in a Bayesian network that all variables are independent from one another, aside from direct dependants. The advantage of using this method is that an intuitive graphical model can be used to represent the whole structure. Provided sufficient evidence is available, a BN can be used to represent the knowledge of an expert. By using inference techniques, the network can effectively be asked questions about the probability of something happening.

When creating the BN, the relationship between input data and class can either be manually input, or obtained through machine learning techniques. As manually defined relationships impart human knowledge directly into the network, they should be used where appropriate. For example, the relationship between horizon intercept data and output class is simple for a human to input; (any ‘intercept’ clusters have a higher probability of being a collision risk). Attempting to extract this same relationship through data analysis would be needlessly complex. As such, manual input should be used where possible to ensure the network is simplified.

However, other parts of this network will need to be trained from manual classification data using BN machine learning techniques. This will help in determining many of the probability distributions which are used to fuse the information sources. As the data sets required will be very large, this will decrease the development time, as these distributions would alternatively need to be populated manually by an expert. In addition, directly linking data to classification result for complex evidence will dramatically increase the accuracy of the classification compared to what an 'expert' could achieve.

There are a number of further advantages in the use of BNs beyond the increase in performance. Due to the flexible, and probabilistic nature of BNs, both partial and uncertain information can be handled. For example, should the texture classifier fail to provide a result (e.g. if condensation formed on the inside of the camera enclosure, blurring the image) the other data sources will 'take over' and still provide a prediction of the class, albeit with a reduced confidence. In comparison, the AVS data fusion approach would fail completely.

In addition to a discrete class estimate, the final output from a BN includes a probability for each cluster belonging to the most likely class. As this final value incorporates all previous information, a simple threshold can be applied below which all clusters are simply considered unknowns. This provides a simplistic method of tuning the classifier in order to detect potential collision risks.

7.2.1 Generic Structure

The structure of a Bayesian network is similar to other PGM, consisting of a collection of nodes and edges. Each node represents a random variable, and each edge represents the conditional relationship between the connected nodes.

Unlike some other PGM, every Bayesian network is a Directed Acyclic Graph (DAG). DAGs enforce directional dependency, in which information starts at the 'evidence' nodes and permeates directly to the classification. This is done directly, through the use of directed edges (i.e. nodes cannot be mutually dependant upon each other and information only flows one way) and indirectly through being 'Acyclic' (additional nodes cannot form a path to bring data back to an earlier node). Within a DAG, there is no way to start at a node and follow a sequence of edges that eventually loops back to the same node again. The directionality is enforced to ensure that Bayes' theorem can be used throughout the network.

When information is fed into a node, a Conditional Probability Distribution (CPD) is required to 'interpret' the input, providing the probability of each potential output. As this is achieved using Bayes' formula it can be done sequentially, with the posterior distribution of each node used as the prior for the next. By defining the CPD for each node within the entire network, a full Joint Probability Distribution (JPD) can be calculated. A JPD represents the probability of the occurrence of every possible combination of states, across all the random variables in the network. This can be calculated using the chain rule shown below in Eq. (7.1), where A and B are random variables, and i is the number of discrete states of A.

$$P(B) = \sum_i P(B|A=i)P(A=i) \tag{7.1}$$

As JPD include every possible combination of states, they can become unmanageably large with only a small number of nodes. Instead, a more useful consolidation of a BN is the total probability for each variable. To calculate this, a probability distribution for a small subset of variables can be extracted from the JPD. This is achieved by summing all the probabilities in the JPD for each combination of variables that are not required, as shown in Eq. (7.2). This process is known as 'Marginalisation'.

$$P(A) = \prod_{x \in A} P(A|\prod_x). \tag{7.2}$$

At the conclusion of the network, a final classification distribution will be created, providing the probability that each cluster belongs to each class. Rather than draw directly on evidence, there are typically several layers between the observable data and the final result, as data is combined to provide a better result. As such, within a BN, only certain nodes will contain data which can be directly observed (i.e. evidence). However, a large benefit of BN is the ability to create nodes which are not directly based on evidence. The variables for these nodes are 'inferred' from observable data and the BN structure.

Using probability to infer an unobservable variable is known as 'Diagnostic reasoning'. The change from Causal reasoning (top to bottom), to Diagnostic reasoning (bottom to top) can be achieved using the Bayes Rule shown in Eq. (7.3); where $P(B|A)$ is the likelihood which will be obtained from the CPD of B, $P(A)$ is the prior, and $P(B)$ is the marginal likelihood, which is used to normalise the probability.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{7.3}$$

7.2.2 Scalability

Although these equations form the basis of the BN approach, directly using these equations for conducting inference is only sensible for small networks. BN for classification typical consist of multiple nodes, each with several states. For example, if a network consisted of just 6 nodes, each with 4 states, this would require 4096 entries in the JPDs.

Instead, methods such as belief propagation [84], or junction trees [27] are used to speed up calculating an exact solution. In using these methods, direct marginalisation of the joint distribution is not required. For an approximate solution (usually for extremely large Bayesian networks) various statistical sampling techniques can be used; such as Markov-chain Monte-Carlo sampling.

Horizon Class	Probability
Below	0
Intercept	1
Above	0

Texture Class	Probability
Asphalt	0.29
Grass	0.05
Sky	0.08
White Paint	0.2
Yellow Paint	0.14
Red Paint	0.18
Building	0.06

Table 7.2 Example of hard evidence (left) and soft evidence (right) for use in a Bayesian Network. Although the probabilities in both tables sum to one, only the soft evidence provides any indication of confidence in the data.

7.3 Bringing Evidence into the Bayesian Network

As each node within a BN works using Bayes' theorem, both the input and output are in the form of a probability distribution. This applies not only to the nodes, but also to the BN as a whole. As such, each type of data used within the network must first be converted into a probabilistic form. Although any form of probability distribution can be used, to make calculations simpler, it is typical to work with either discrete data or Gaussian distributions within a BN.

As stated in Chapter 6, the primary data sources for classification are cluster colour and texture. Due to the texture data extraction method providing a direct estimate of class, it is difficult to represent texture data using a Gaussian distribution. Therefore, a discrete BN implementation has been chosen. This is also useful for the inclusion of additional data, such as the relative position of the horizon line, which is already discretised. As the method of discretisation is unique to the data type, the following section will describe how the data is discretised for use in the BN, (if required).

Aside from discretisation, the data must either be submitted as 'Hard' or 'Soft' evidence. Hard evidence is suitable for data which is known to be certain. For example, for the 'Horizon' intercept data is best submitted as 'hard evidence'; having only three mutually exclusive classes of *above*, *below* or *intercept*. By comparison, soft evidence allows uncertainty to be represented within the network. For a classification based result, such as texture, the additional information informs the network of how 'confident' the results are, allowing weighting to be applied. (The techniques used for solving and conducting interference on a Bayesian network for soft evidence are based on the work in [16]).

The difference between hard and soft evidence can be seen in Table 7.2. On the left, horizon data is simply submitted with a winning class, whereas the texture data on the right is submitted as a distribution. If hard evidence were to be used for texture data, the asphalt class would have simply been assigned the entire probability, despite the fact that several other classes also have a high probability of being the correct class. As such, in addition to discussing discretisation, a method of converting the data into a probability distribution will also be established in the following section.

Fusion between data sources will be achieved using the concept of 'sub-networks'. Although each source of evidence will provide data in a probabilistic form, the states may differ widely from the final classification (e.g. the horizon intercept states are $above$, $intercept$ and $below$, rather than $grass$, $asphalt$, sky etc). As such, each sub-network is intended to associate the input data with the probability of that cluster belonging to each class. A final node will then be used to combine the probabilities from each sub-network into a final class estimate. A simplistic overview of the Bayesian structure, showing only the sub-networks, is shown in Figure 7.2.

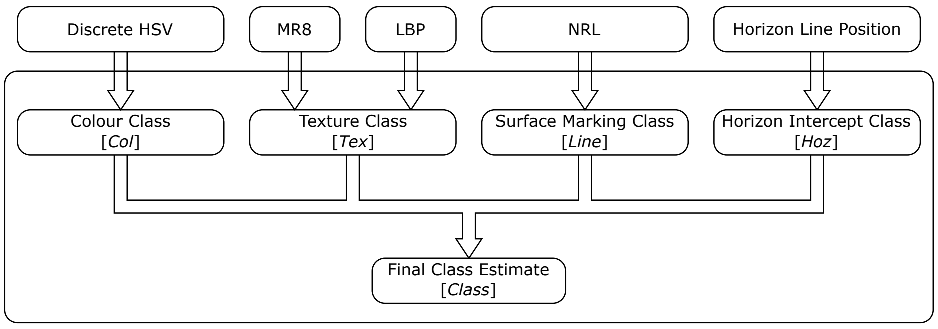

Discrete HSV
MR8
LBP
NRL
Horizon Line Position
Colour Class [*Col*]
Texture Class [*Tex*]
Surface Marking Class [*Line*]
Horizon Intercept Class [*Hoz*]
Final Class Estimate [*Class*]

Fig. 7.2 Simplistic Overview of Bayesian Sub-Network Structure

In order to complete the network, the Conditional Probability Distributions (CPDs) need to be determined. For each node, the CPD states the marginal probability of each variable with respect to the other variables within that node. Unlike the intuitive network structure, some numerical parameters are harder to elicit using human expertise. For example, despite the significant decrease from 16 million to 2161 colours for HSV colour classification, there are still far too many discrete colours for manual conversion. Therefore, where suitable, parameter estimation techniques will be used to calculate the CPDs. As large data sets are common for problems which require classification, the use of Parameter estimation for Bayesian networks is well established. Many different techniques are proposed in literature but for this work Maximum Likelihood Estimation (MLE) has been selected, as it has established use for image classification, having already been demonstrated to improve performance in texture based classification, specifically for skin detection [95].

MLE is designed to work with a small sample of a complete population. Despite using many training images, it is impossible (or at least highly impractical) to record ever possible type of surface relevant to each class. As such, the training data set can only be considered a small percentage of the total population. However, assuming that the samples are distributed much like the full data set, MLE assumes that the observed data represents the most likely results, as they have actually been observed. The parametric values for the model are then determined by maximising the likelihood function, i.e. the model is made to fit the observed data as well as possible. The application of MLE to BN is explained in detail in [62].

In order to minimise complexity, each CPD is trained within it's sub-network, reducing the number of examples required for each training set. Therefore, a training set of 100 manually classified images are used to determine each CPD.

7.3.1 Texture Data

From the example shown in Table 7.2, it is clear that asphalt has the highest probability of being the correct class. If texture data were to be used in isolation, there would be little to gain in submitting soft evidence as it would be easier to simply identify which class was most likely. However, as the BN fuses data from multiple sources, this probabilistic data can be essential. For example, if the associated colour data for this cluster were to be overwhelmingly yellow, it would be more likely that yellow paint is being observed. If texture data was taken as hard evidence, this would already have been discounted. As such, texture data will be added to the network as soft evidence.

In order for this to be possible, the texture classification output must be converted from a distance function into individual class probabilities for each cluster. As established in Section 6.3.9, for both MR8 and LBP feature descriptors, the texture classification is achieved using SVMs within a BDT. There are two major difficulties in extracting a probability distribution from this result:

1. SVM classification distance is not probabilistic.
2. Multiple classes were grouped together during each vote.

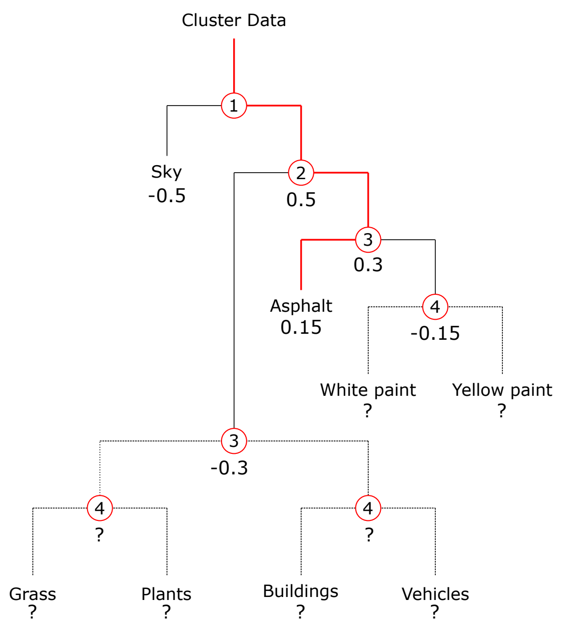
Beginning with the first problem, drawing probability directly from an SVM is difficult. For all SVMs, increased distance from the hyperplane does not linearly correspond to increased confidence of being the winning class. Work undertaken by Platt (1999) [85] recognises this, and suggests a method of using an additional trained function to convert the output of an SVM into probability. However, this assumes that the SVM is isolated, and is used to directly estimate probability between two classes. Figure 7.3b is intended to demonstrate how each SVM vote works within the same feature space when using texture data. As such, a multi-class SVM BDT is used to establish regions within the feature space which correspond to each class. (As these regions have extremely high dimensionality, SVM is far more efficient than attempting to define the boundaries of each region.)

Each vote can be considered as a line dividing the texture feature space. Each time the space is bisected, the side of the line on which the cluster falls helps to narrow it's class. Typically when using a BDT, probability is extracted from each vote, and Bayes rule is used to track the flow of probability through the tree. However, as the distance between the cluster data and each hyperplane are directly comparable, cumulative probability is not appropriate. (Despite being further from the cluster, classes which appear higher in the BDT (i.e. those which are most distinct) would receive an artificially higher probability). Instead, it is more accurate to preserve the distance to each voting hyperplane, and assign probability in a single step once voting is complete.

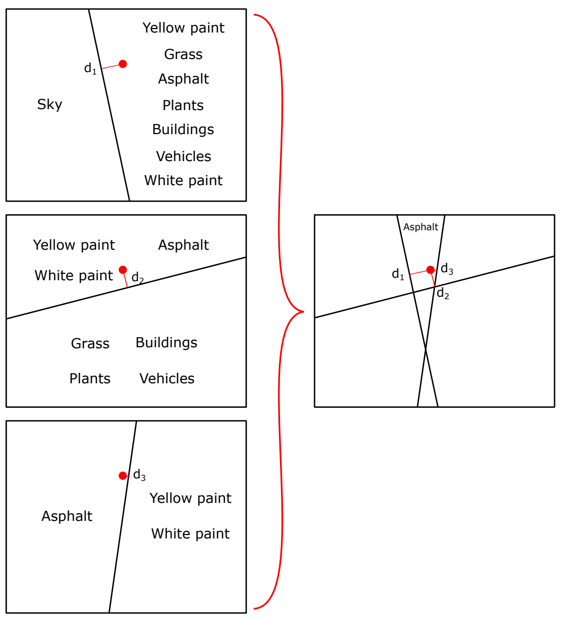
The second difficulty in using the texture data is that the BDT method groups multiple classes together. Although this quickly determines the final result (by minimising the number of 'votes' required overall), it fails to provide a 'texture distance' for every possible class. As shown in Figure 7.3, the final classification result is found through only 3 votes, with most of the potential classes having no data associated within them, beyond the group level.

When building the BDT, each voting layer is built using maximum dissimilarity, with the two groups selected such that the distance between their centres is maximised. Although the distance to each member of the losing group will differ, the maximum potential likelihood of the cluster being each class will be the distance in the loosing vote. (i.e. further dividing the feature space can only increase the distance between each class and the cluster data). Therefore, the last vote in which each class appears is found, and the distance of the entire subset is used for each individual class with that group. As the more likely classes are compared more closely as votes progress down the BDT, the overall precision loss is considered negligible.

As recovering an exact probability for each class is extremely complex due to the voting method, this work instead aims to produce a 'representative probability', based on the distance from the hyperplane. In order to represent distance as a probability, the distance is assumed to be proportional to the likelihood of being class (simply assuming that



(a) SVM voting within the BDT.



(b) SVM voting within feature space.

Fig. 7.3 Examples of SVM voting, both within the BDT and the Texture feature space. For each vote, the distance from the hyperplane into the class is expressed below each class in the BDT. Losing classes receive negative values as the data point is the wrong side of the hyperplane.

the cluster distance from each hyperplane is indicative of the confidence of belonging to that class). However, prior to this, the directionality of the distance to each hyperplane must be accounted for.

For each vote, the loosing group is assigned the distance from the hyperplane in the negative direction, aside from the final result, which is assigned a positive. This raw result can be considered the “SVM Distance from Hyperplane”. As the winning class may be close to the final hyperplane, the significance of actually being on the correct side of the hyperplane should be recognised. As distance is being used as the metric for conversion, a positive/negative distance is not useful. Instead, the winning result should have a distance of zero, to demonstrate maximum confidence. As such, the maximum value (i.e. the winning distance from the hyperplane) is subtracted from all other results, providing an adjusted absolute distance between the cluster and each class. This will be referred to as the “Adjusted Distance from Hyperplane”.

The adjusted distance from the hyperplane can now be converted to Likelihood [41]. For this, a Gaussian distribution is considered appropriate, as it quickly saturates to low probability when the hyperplane is far from the cluster data point. As the number of

Class	SVM Distance From Hyperplane	Adjusted Distance From Hyperplace	Likelihood	Representative Probability
Asphalt	0.15	0	0.7979	0.3411
Yellow Paint	-0.15	0.3	0.6664	0.2848
White Paint				
Grass	-0.3	0.45	0.5322	0.2275
Plants				
Buildings				
Vehicles				
Sky	-0.5	0.65	0.3427	0.1465

Table 7.3 Example of converting Texture SVM distance to Representative Probability

dimensions is known, and the results are normalised based on cluster size for comparison, the maximum texture distance can be calculated. This allows a standard Gaussian Probability Density Function (PDF) to be created with $\mu = 0$ and $\sigma = 0.5$. By applying the PDF to each adjusted distance, the "Likelihood" is obtained as shown in Equation 7.4, where D is the adjusted distance.

$$L(Class) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(D_{Class}-\mu)^2}{2\sigma^2}} \tag{7.4}$$

However, as the final probability distribution must sum to one (for compatibility with the BN), the final stage is to normalise the results, with respect to the sum of all probability. This is then known as the 'Representative Probability", which is highly representative of the actual probability, but is calculated with a much lower computational cost. This is then ready to be provided to the BN. An example of converting from raw SVM distance to the representative probability distribution is shown in Table 7.3, based on the voting undertaken in Figure 7.3.

$$P(Class) = \frac{L(Class)}{\sum_{i=1}^{n} L(n)} \tag{7.5}$$

7.3.2 Combined Texture Node

As stated in Section 6.3.3, two different texture descriptors have been used for this work, such that the strength of one method accounts for a weakness in the other. As MR8 samples from a large area, it should benefit from the additional information when classifying large clusters, but risks including too much data from neighbouring clusters when attempting to classify small clusters. By contrast, as LBP is extremely localised, is is typ-

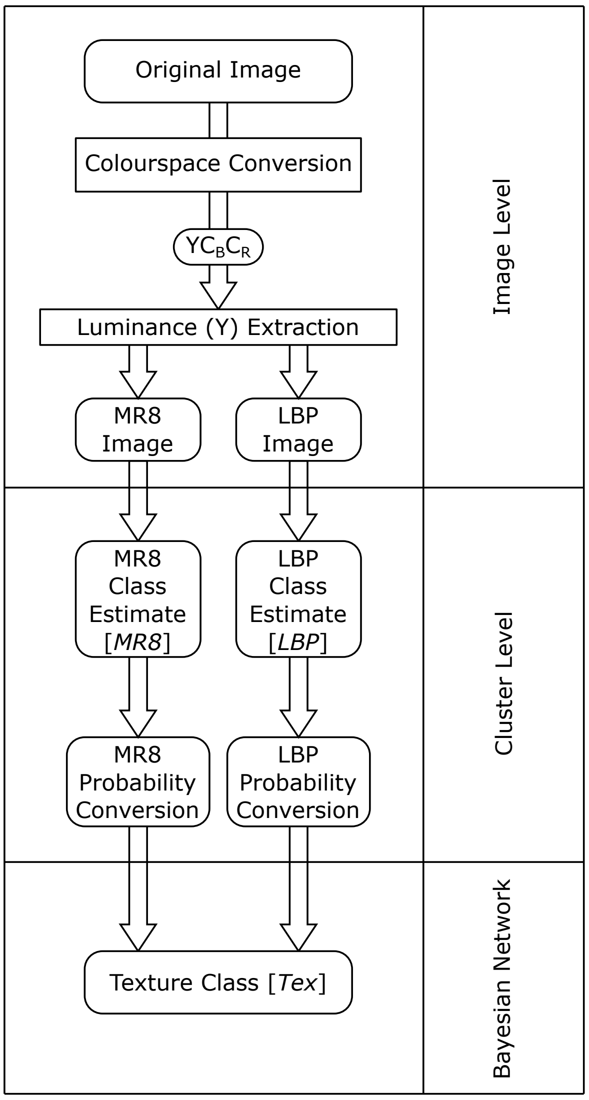


Fig. 7.4 Representation of the entire Texture classification process

ically more accurate on small clusters. However, the small sample size decreases the expected accuracy overall.

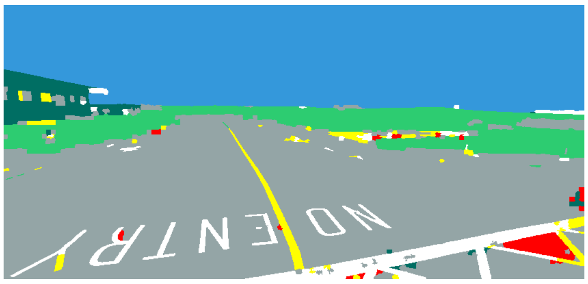
Throughout the prior stages, the MR8 and LBP texture descriptors have been processed separately. However, for use in the BN, MR8 and LBP data are combined together for the final classification. Although both sources of data could be passed to the final classifier independently, a 'Texture classifier' node (Tex) is established which fuses the data before the final node. This allows the entire texture classification process to be assessed and trained separately, with the relationship between MR8 and LBP explored without the influence of other sources of data. The entire texture classification process can be seen in Figure 7.4.

Data fusion within the texture node (Tex) could be achieved by simply averaging the LBP and MR8 result for each class. However, this would fail to include the knowledge that cluster size influences the relative success of each classifier. Instead, the size of the cluster will be used to proportionally weight the contribution of each classifier, based on the accuracy of each technique when compared to manually classified images in the training set. The correlation better texture descriptor accuracy/weighting and cluster size could be manually included in the CPD of the combined texture class node (Tex). However, determining how to bias these results would require a great deal of manual effort. Instead, these differences in performance can be captured more accurately by using parameter estimation techniques. For this work, every cluster in the training set has undergone manual classification, providing a 'complete' data set. This allows MLE to be used. Using MLE, a training set of 100 manually classified images are used to determine the CPD $P(Tex \mid MR8, LBP)$.

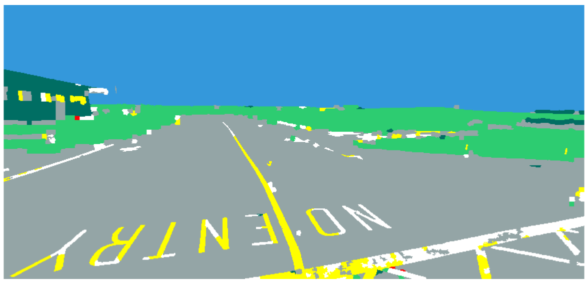
Figure 7.5 depicts the classification results achieved using BDT based SVM within the BN. Figures 7.5a and 7.5b show the individual result of using from the MR8 and LBP nodes, based on the highest probability being the winning class. At a pixel level, this achieves results of 93.5 % and 92.19 % respectively. For comparison, Figure 7.5c shows the output from the [*Tex*] node, in which MR8 and LBP have been combined. The winning class within [*Tex*] satisfies the condition:

$$T_i = \underset{Tex}{\operatorname{argmax}} P(Tex \mid MR8, LBP) \tag{7.6}$$

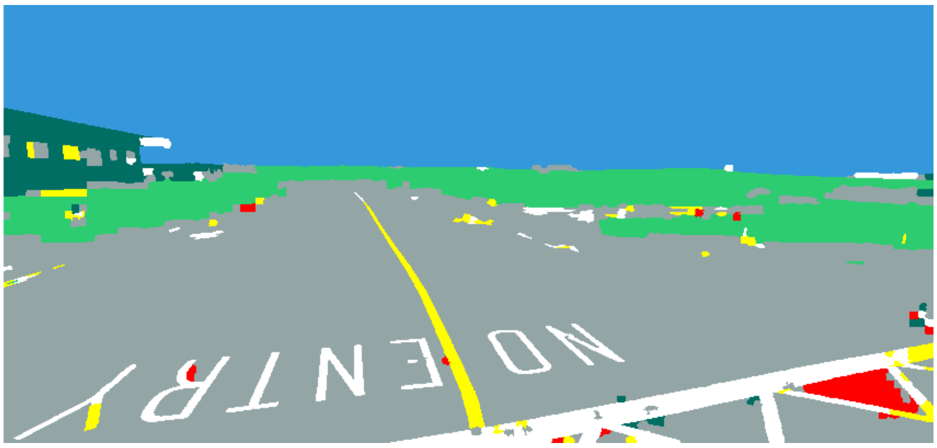
Using the combined classification produces an improved result of 94.1 %. Although this is only a small improvement, the improvement in surface marking classification in Figure 7.5 demonstrate that the concept is working as intended.



(a) MR8 Only Classification



(b) LBP Only Classification



(c) Combined Texture Classification

Fig. 7.5 Comparison of texture classification results for LBP, MR8 and Combined Texture classification nodes.

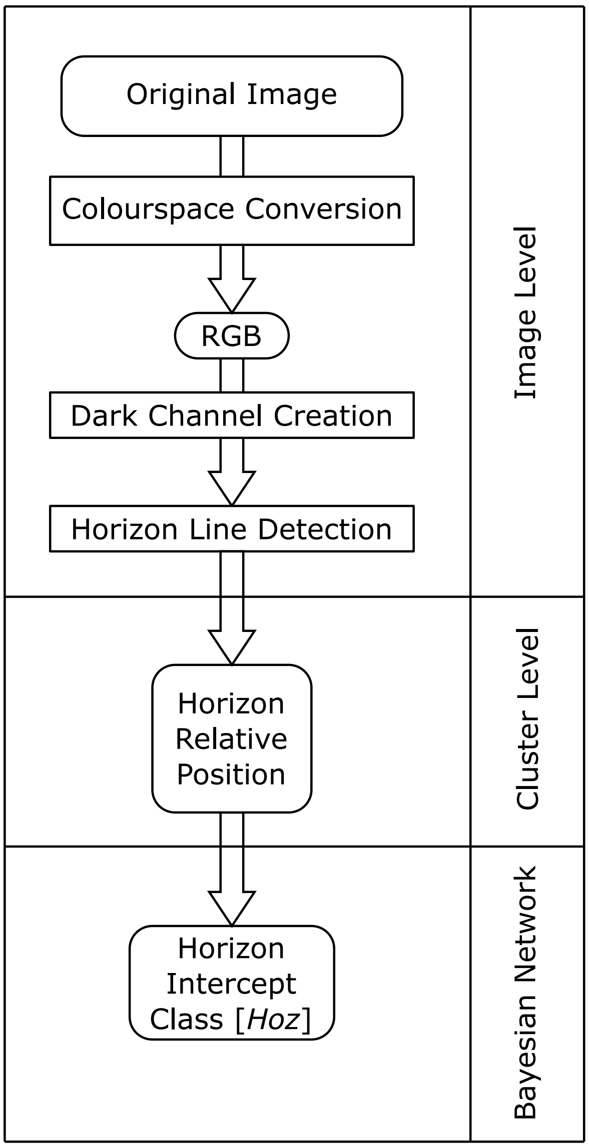


Fig. 7.6 Representation of the entire Horizon Intercept classification process

7.3.3 Horizon Intercept

In comparison to the effort required to extract a texture classification, the horizon intercept classification is far more straightforward. As the remaining BN sub-networks all follow a similar approach, and the Horizon Intercept sub-network is the least complex, it is introduced first to establish the concept.

As described in section 6.5, the position of each cluster relative to the horizon line has already be discretised into only three potential states: $Above$, $Below$ and $Intercept$. As all three states are mutually exclusive, for each cluster the sum of the input data is always

	Asphalt	Grass	Sky	White Paint	Yellow Paint	Red Paint	Building
Above	0.1	0.1	0.4	0.1	0.1	0.1	0.1
Below	1/6	1/6	0.0	1/6	1/6	1/6	1/6
Intercept	0.1	0.1	0.1	0.1	0.1	0.1	0.4

Table 7.4 CPD for Horizon Intercept Sub-Network

one, satisfying the BN requirement. Shown in Figure 7.6, the entire Horizon Intercept Bayesian sub-network consists of a single node $[Hoz]$. The 'Horizon Intercept Logic' is then applied to the network within the $[Hoz]$ node's CPD. For each cluster, the probability of being each class, based on the horizon intercept data alone, is shown in Table 7.4.

For clusters with the $Above$ state, the probability of that cluster being any class other than Sky is drastically reduced, as objects are unlikely to pose a collision risk when entirely above the ground. Conversely, for clusters with the $Horizon$ state $Below$, the probability of that cluster being Sky is reduced to zero, to prevent false classifications of sky on the ground. Finally, if the $Horizon$ state is $Intercept$, all classes aside from $Building$ are dramatically decreased. Although the horizon line should be entirely below the sky area and therefore small sky clusters could be captured in the intercept class, the failure for the reachability clustering to merge this cluster into the greater sky suggests it is not a typical sky cluster. Despite this, the presence of such a cluster at the horizon implies that whatever object is represented by this cluster is likely a very long distance from the UAS, and therefore does not pose significant risk.

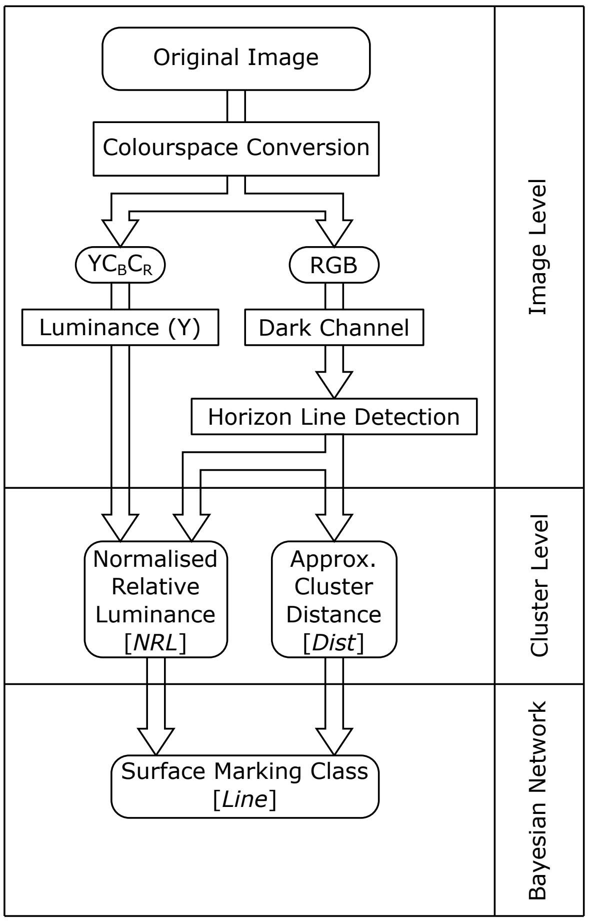


Fig. 7.7 Representation of the entire Surface Marking classification process

7.3.4 Surface Markings

Due to the benefits of accurately extracting surface markings for localisation, a separate sub-network has been created specifically to aid in their classification. As shown in Figure 7.7, the primary source of information for Surface Marking classification is NRL, for which the extraction process is detailed in Section 6.4.3. In order to redefine the luminosity value for each cluster on the ground (relative to the maximum luminosity value on the ground), the position of the horizon line is also required. Therefore, unlike the other sub-networks, the Surface-Marking Classification sub-network draws from multiple sources of information. Should the camera be positioned such that the horizon is not visible, both Surface Marking classification and Horizon Intercept classification will depend on inertial measurements from the IMU aboard the UAS.

Prior to use within the BN, the NRL data needs to undergo discretisation. In Section 6.4, the discretisation of HSV data was achieved using 2161 states. By contrast, just three states were found to be sufficient for NRL: $High$, $Medium$ and Low. Initially it was intended that only two states would exist ($High$ and Low), separating surface markings from other ground clusters through a single threshold. Obvious surface markings should be captured within the $High$ state and all other clusters should fall within the Low state.

However, the quality of surface markings is highly variable, with worn and faded lines often sharing similar NRL values to other classes. As extending the $High$ state to include these values would lower the confidence in the classification, a third state was introduced. This $Medium$ state, cannot be considered as indicative of surface markings as the $High$ state, and therefore will have a lower probability of representing surface markings within the CPD. The values used to define each state are shown in Table 7.5, having been determined empirically from test images.

NRL states	
High	$NRL_i > 0.77$
Medium	$0.65 < NRL_i < 0.77$
Low	$NRL_i < 0.65$

Table 7.5 NRL discrete states

Despite NRL being a clear indicator of surface markings in the foreground, atmospheric effects can alter the NRL values of distant clusters. On clear days, Rayleigh scattering scatters blue light more than red, lowering the NRL values of objects in the distance. Conversely, on overcast/rainy days, the presence of water droplets in the air introduces haze/fog, which scatters all light wavelengths equally, shifting the colour towards white. As photons from objects in the distance must pass through more haze, the NRL value for all clusters is increased with distance from the camera.

As the image data set was captured entirely on wet, overcast days, NRL values can be observed to increase towards the horizon. Consequently, classes which would have Low NRL states in the foreground gain $Medium$ or $High$ states when far from the camera. As the distance to these clusters is fairly large, even if potential collision risks are miss-classified as surface markings there is minimal risk to the aircraft. However, as surface markings are some of the few terrain features common to most aerodromes, they are extremely useful for localisation. With specific surface features (such as bends or junctions) being hundreds of metres apart, localising the aircraft requires that lines be correctly identified even at range. Therefore, the correction of false positives is a necessary step.

In order to correct for the atmospheric effects, the NRL state must be fused with the

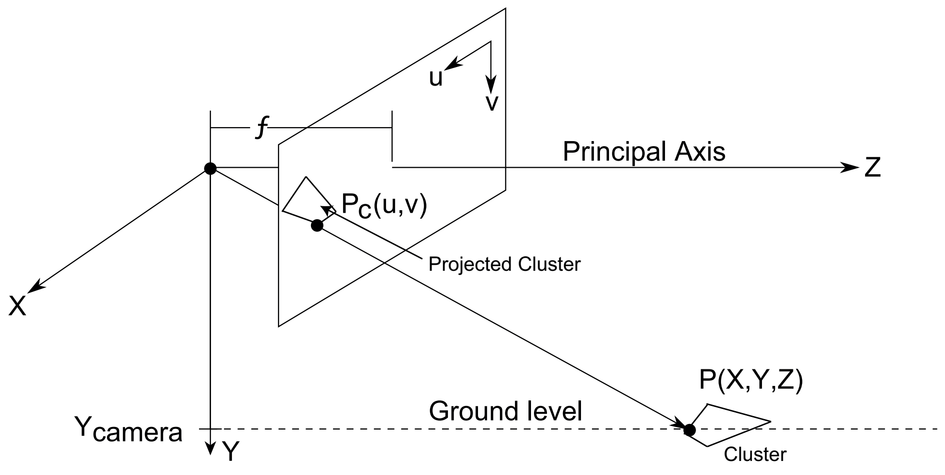


Fig. 7.8 Pinhole camera model used for depth estimation

estimated cluster distance from the camera. As only distant clusters are affected significantly, precise distance estimation is not required. Therefore, for simplicity of concept, distance to cluster is approximated using the pinhole camera model. Using the camera's focal length (f) and height above the ground (Y_{cam}), similar triangles can be used to map between the 3D position of the point $P(X,Y,Z)$ and the position of the point within the image $P_c(u,v)$, as shown in Figure 7.8. Simple trigonometry can then be used to find the ground distance D_c, as stated in Eq. 7.7.

$$\begin{aligned} Z &= \frac{f.Y_{cam}}{v} \\ X &= \frac{u.Y_{cam}}{v} \\ D_c &= \sqrt{Z^2 + X^2} \end{aligned} \tag{7.7}$$

The estimated distance from camera (D_c) must also be discretised to be used within the BN. The discretised distance ($Dist$) is similar to NRL, using just three states: $Close$, Mid and Far, as shown in Table 7.6. As both NRL and $Dist$ are intended solely to improve the classification of lines, both are combined within the same sub-network of the BN, as shown in Fig. 7.7. The sub-network concludes with the Surface Marking Class node, which uses the variable $Line$ to represent the probability that each cluster is a surface marking. $Line$ has two states, true (T) and false (F), which represent the probabilities of a cluster being a surface marking class. The CPD of the $Line$ node is shown in Table 7.7.

[$Dist$] states	
Close	$D_c < 20m$
Mid	$20\text{m} < D_c < 55\text{m}$
Far	$D_c > 55\text{m}$

Table 7.6 Discrete states for estimated cluster distance from camera [$Dist$]

NRL state	Dist state	True	False
Low	Close	0	1
Medium	Close	0.75	0.25
High	Close	0.9	0.1
Low	Mid	0	1
Medium	Mid	0.55	0.45
High	Mid	0.8	0.2
Low	Far	0	1
Medium	Far	0.2	0.8
High	Far	0.6	0.4

Table 7.7 CPD for surface marking classification node - $P(Line|\text{NRL}, Dist)$

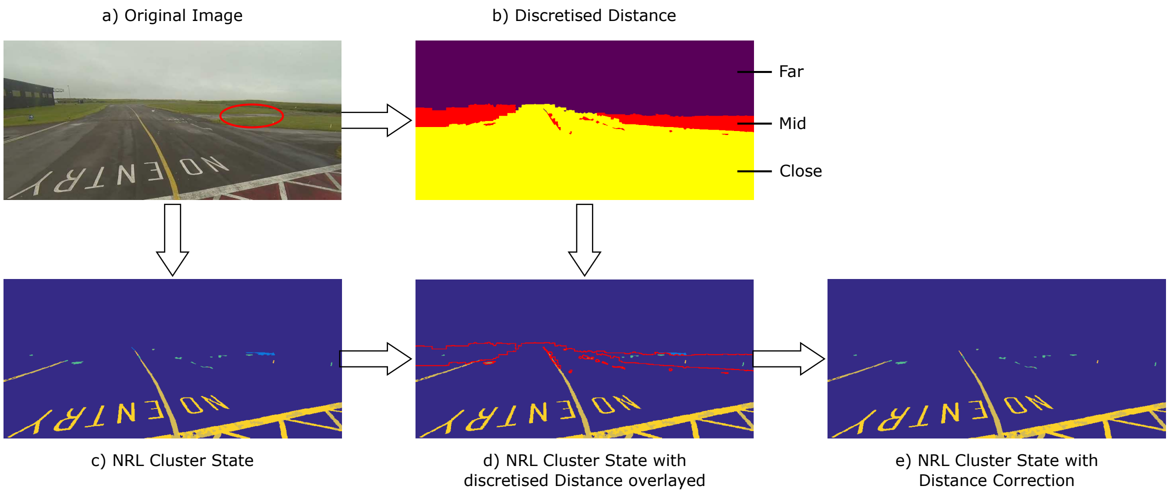

a) Original Image
b) Discretised Distance
Far
Mid
Close
c) NRL Cluster State
d) NRL Cluster State with discretised Distance overlayed
e) NRL Cluster State with Distance Correction

Fig. 7.9 Demonstration of using $Dist$ to improve NRL classification result. In (a), the original image features some distant asphalt which is given a $Medium$ NRL state due to damp conditions and distance from the camera. By extracting the relative distances, shown in (b) and overlaid in(d), the final classification has correctly lowered the clusters NRL state to Low in (e).

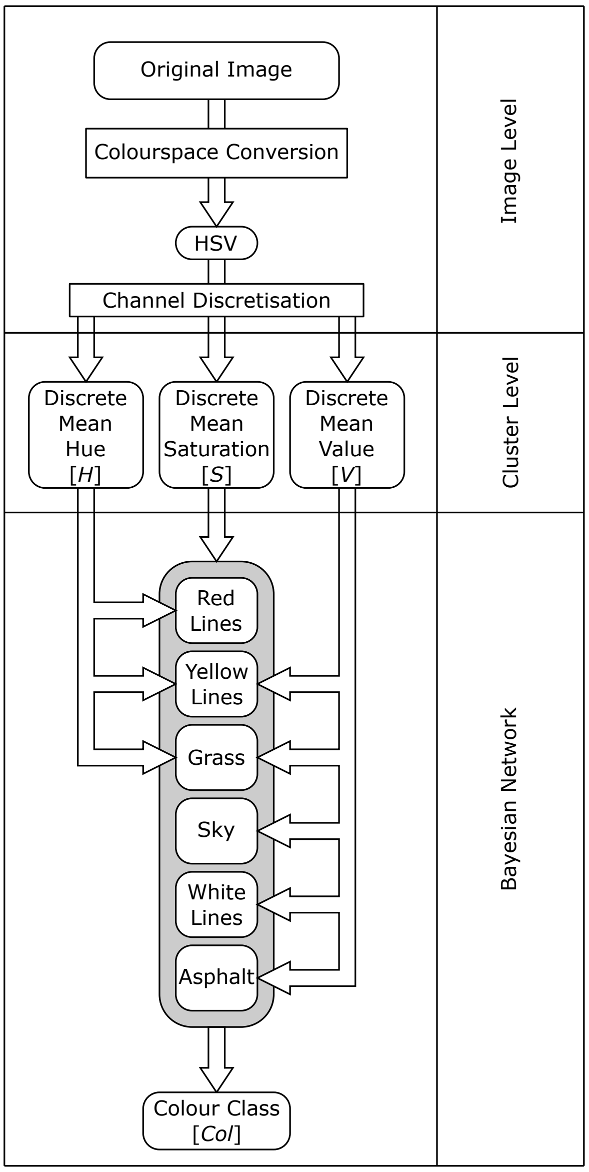
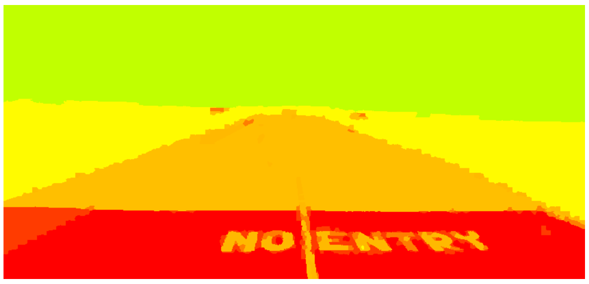


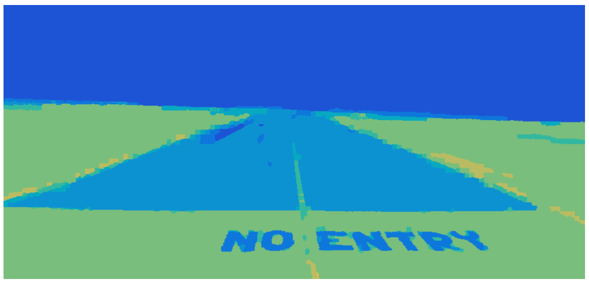
Fig. 7.10 Representation of the entire Colour classification process

7.3.5 Hue, Saturation and Value

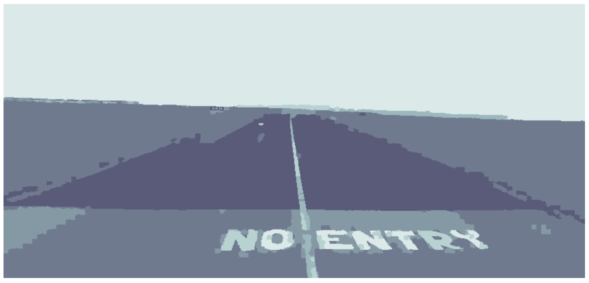
Compared to the simplicity of the single nodes used for the other classification sub-networks, the HSV "Colour Classifier" sub-network is more complex. Shown in Figure 7.10, the HSV sub-network provides a classification estimate using colour data alone. Unlike texture, where data can be directly related to a single class, multiple classes can share the same colour information. As such, the likelihood of a cluster belonging to each class must be determined independently, resulting in an individual node for each potential class. The only exception is the "Building" class, as building colour is too variable



(a) Discrete Hue



(b) Discrete Saturation



(c) Discrete Value

Fig. 7.11 Discrete individual H S V channels for example aerodrome image

to determine correlation. Therefore, only the other six possible classes are represented. Each node has just two discrete states: true (T) and false (F).

The process of colour extraction and discretisation has already been covered in Section 6.4. From the original 16 million possible colours, the colour space has been simplified down to just 2161, in order to allow a between association between colour and class. Conversion to probability is achieved by submitting the colour data as hard evidence within the three colour channels.

As discussed in Section 6.4.2, all three colour channels will not be useful for every class. Instead, only relevant information is passed through the sub-network. The edges in Figure 7.10 indicate which classes are conditioned using which inputs. $Saturation$ is present for all classes as the difference between colourful/grey is useful in all cases. $Value$ is nearly as useful as the brightness of a cluster is indicative of its class. The only class which does not use *Value* during classification is 'Red surface markings', as the *Value* range has been found to be too great to be meaningful.

Figure 7.11a shows the effects of equalising *Saturation* and *Value* for every pixel in the image. When only *Hue* varies, the usefulness of this colour channel is highly apparent. For clusters where *Saturation* is high, a similar representation is common. For example, red surface markings are clear to observe. However, when the saturation is low, *Hue* has no meaning. This can be seen in the unusual colours shown within the asphalt and white surface markings. As such, although *Hue* is useful, it can only be of use in clusters with high saturation. The *Hue* colour channel is used to differentiate regions with high *Saturation* and *Value*. Only surface markings (red and yellow) and grass use *Hue* during classification.

As all three colour channels are influenced by light levels and atmospheric conditions, the colour range of each class varies dramatically, even during a single day. Therefore, it is extremely difficult to define typical levels for each class node manually. Instead, using the same methodology as used for the texture node, MLE has been used to train each of the CPDs.

7.3.6 Colour Class Node

The Colour class node Col is a hidden node which simply combines the individual true/false probabilities into a single node. This simplifies the network, making it easier to observe the output of the colour classifier, as all classes can be compared in a single node. In addition, this also makes the CPD of the final $class$ estimate node much simpler, as it will have only a single parent. As this final fusion is easy to understand, for this node the CPD has been completed manually.

Using the sky class as an example, there are three potential outputs from the hidden node for each class. As the colour of a single cluster *could* be indicative of belonging to several classes, the number of classes which could match on colour alone must be taken into account. If the cluster colour is such that only this class (i.e. sky) is likely, the probability that the colour derived class is sky, given the sky node is true and all others are false, is one. This is shown in Eq. (7.8).

$$P(Colourclass^{sky}|asphalt^F, grass^F, sky^T, white^F, yellow^F, red^F) = 1 \tag{7.8}$$

Equally, if this class has no association with the cluster colour, then the relevant class parent will be false and the probability will be set to zero, as shown in Eq (7.9).

$$P(Colourclass^{sky}|asphalt^{F}, grass^{T}, sky^{F}, white^{F}, yellow^{F}, red^{F}) = 0 \tag{7.9}$$

Finally, if several classes are possible based on colour alone, the probability is uniformly distributed between them. This is shown in Eq.(7.10), where there are two potential classes which each receive a probability of 0.5.

$$P(Colourclass^{sky}|asphalt^{F}, grass^{T}, sky^{T}, white^{F}, yellow^{F}, red^{F}) = 0.5 \tag{7.10}$$

An example of the classifier output is shown in Figure 7.12. In this figure, the top image shows a typical aerodrome scene which has undergone discretisation in HSV. The class probabilities of each cluster are calculated from $P(Colourclass|H, S, V)$, which is the marginal probability distribution of $Colourclass$ with H, S, V entered as evidence. The winning class is selected using Eq 7.11. Where multiple classes are equally likely, the first node in sequence is selected.

$$c_i = \arg \max_{ColourClass_i} P(ColourClass_i|H_i, S_i, V_i) \tag{7.11}$$

Using only the HSV colour classifier, the percentage of correctly classified pixels in Figure 7.12 is 95.6 %. It can be clearly seen that the largest source of error is the misclassification of white surface markings as sky. This is due to the two classes sharing the same discrete colour and performing classification without any additional context. By simply factoring the relative horizon position, this error could be removed. An additional error has been introduced by having no possible class for building. As such, the building has been classified as asphalt with a probability of 0.5778, due to similarly low *Saturation* and *Value*. Despite this, as colour classification is only intended to add the texture classification process, the accuracy of these results confirms that colour is a useful addition to the BN.

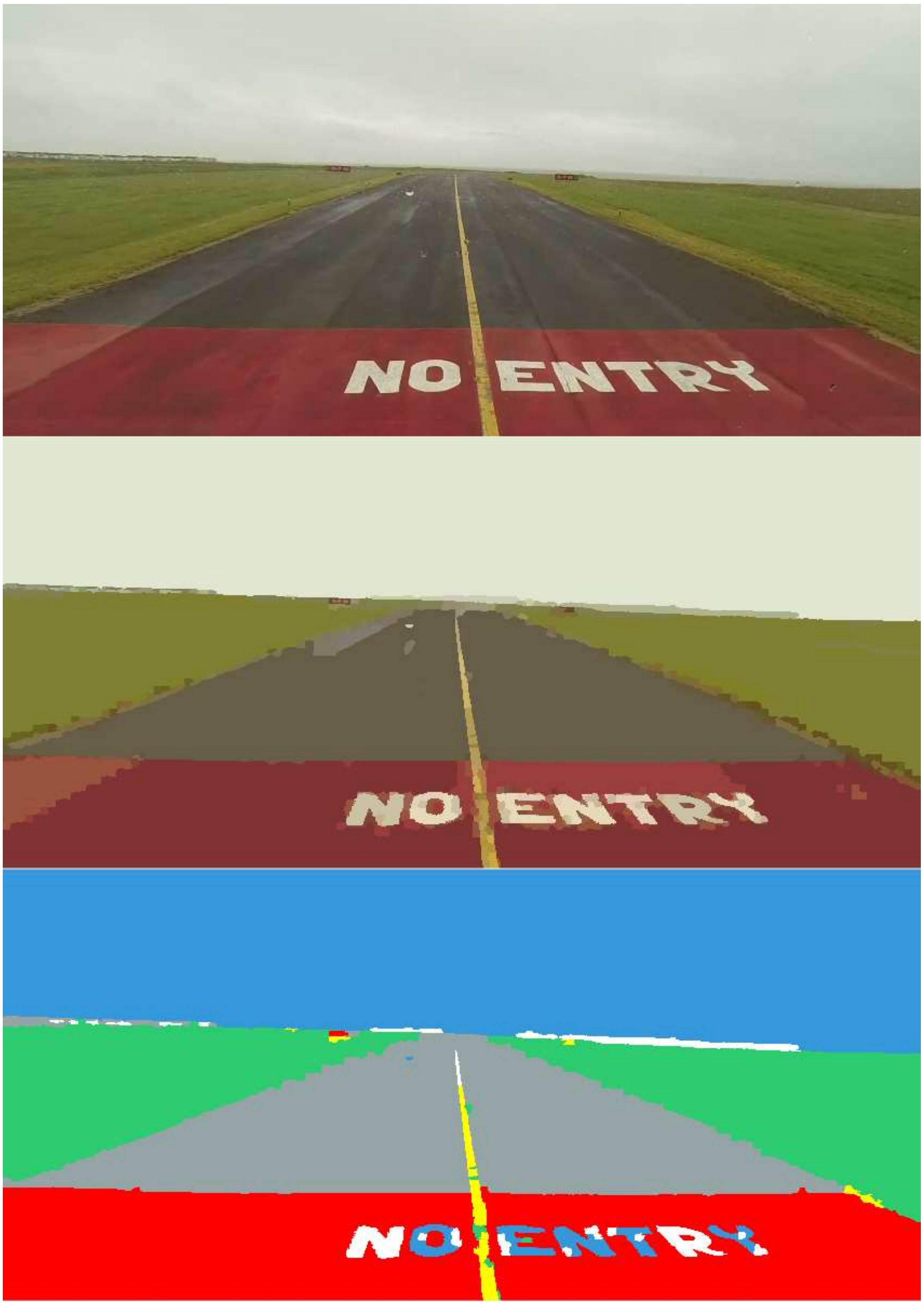


Fig. 7.12 Example of an image progressing through colour-only classification. The topmost image shows the original 24-Bit colour representation. The middle image shows the Discretised HSV colour representation, with only 2161 colours. The bottom image shows the trained Bayesian network output, using only colour classification.

$[Col]$	Colour Class	$[H]$ $[S]$ $[V]$	Discrete Mean Hue Discrete Mean Saturation Discrete Mean Value
$[Tex]$	Texture Class	$[MR8]$ $[LBP]$	MR8 Texture classification estimate LBP Texture classification estimate
$[Line]$	Surface Marking Class	$[NRL]$ $[Dist]$	Normalised Relative Luminance Estimated cluster distance from camera
$[Hoz]$	Horizon Intercept Class	$[Horizon]$	Relative horizon position

Table 7.8 The four sub-networks of the Bayesian network, with their associated input evidence.

7.4 Complete Bayesian Network

For each individual cluster in the original image, data is extracted and provided to the BN, as displayed in Table 7.8. At the conclusion of the previous section, the BN consisted of four separate sub-networks, each providing independent probabilities that the a cluster belongs to each class. To complete the BN, the output of the four sub-networks must be combined. Section 7.3 discussed each of the sub-networks; establishing their strengths, weaknesses and intended roles within the BN. Using this information, the four sub-networks can be fused using manually determined logic and therefore the CPD for the Final Class Estimate $[Class]$ has been created manually.

As the output of each sub-network is a direct estimate of class, further data-type conversion is not required. The outputs of all the sub-networks are multiplied together within each class, using experimentally determined weighting in order to achieve the best result. Combining the elements of classification together, the entire classification process is now defined, as shown in Figure 7.13.

One of the main motivations for using a BN approach is the ability to include probability within each stage of the classification process. Not only does this allow the fusion of widely different data types, but it also provides confidence in the output from each sub-network, and the final result overall. From the final BN node $[Class]$, the final output is a probability of the cluster belonging to each class, with the highest probability indicating the most likely class. This is summarised in Equation 7.12, where c_i is the class assigned to cluster i.

$$c_i = \arg\max_{Class_i} P(Class_i | H_i, S_i, V_i, MR8_i, LBP_i, NRL_i, Dist_i, Horizon_i) \tag{7.12}$$

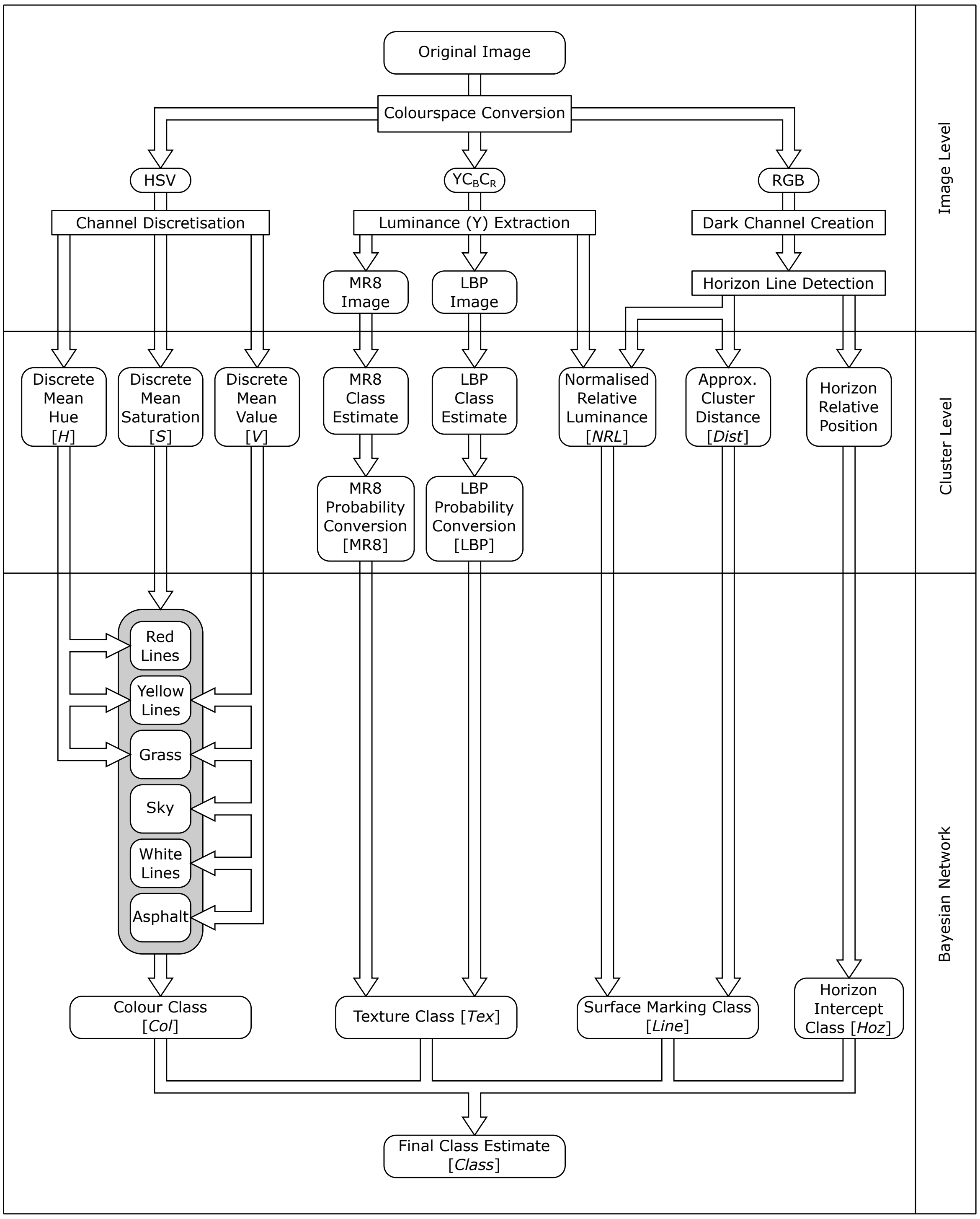


Fig. 7.13 Representation of the entire classification process

7.5 Unknown classes from uncertainty

As established in Chapter 4, attempting to detect every possible object that could appear within an aerodrome environment is highly impractical. Instead, the intention of this work is to detect and classify terrain features, inferring the presence of other objects (i.e. collision risks) from regions within an image which cannot be classified. Therefore, recognising when a cluster is not classified is just as important as determining the most likely class.

Returning to the BN, the final result is the probability of each cluster belonging to each class. As established in Equation 7.12, the overall winning class can be determined by simply selecting the class with the greatest probability. However, if a winning class is always declared regardless of the confidence in classification, this classification method fails to include the concept of 'Unknown' classes, essential for collision risk detection.

In cases where the cluster class is 'Unknown', the combined result of the BN should produce a low probability of being any of the known classes. Therefore, as the final winning class is associated with a probability of actually being that class, 'Unknown' clusters can be identified when confidence in the winning class is low. As the probability of the winning class will always be between zero and one, a simple threshold can be applied. Equation 7.13 represents the method used to determined unknown classes, where (C_i) is a cluster, $Unknown$ is an unknown collision risk state and U is the probability threshold. If the maximum class probability for a cluster does not exceed U, it will be classified as unknown potential collision risk.

$$\begin{cases} c_i = Unknown \text{ for } \max_{Class_i} P(Class_i) < U \\ c_i \neq Unknown \text{ for } otherwise \end{cases} \tag{7.13}$$

7.5.1 Parameter Sensitivity

For any BN, the final classification result is entirely dependant on the parameters used to weight the internal elements. Even between dissimilar classes, small changes to parameters can effect the confidence in the result significantly, or even change the 'winning' class. If a small change in internal parameters alters the final result, the BN is said to be highly 'sensitive' to parameter tuning.

For this Bayesian network, the internal parameters were ascertained through 'trial-and-error', comparing the output to manual classification results. As the test set used was limited by the amount of images that could be manually classified in a reasonable time, it is likely that the final parameters are not optimal. However, the design of this BN is intended to reduce the sensitivity. Typically, extremely precise tuning is only required for two situations:

- Where the final result is equally similar to multiple classes.
- Where the final result is dissimilar to all known classes.

When classes are very similar (such as surface markings and asphalt), the final output can be sensitive even to the precision of the parameters used. To account for this, the overall design of the BN is specifically intended to 'resolve disputes' between sub-networks based on the classes. By experimentally confirming a good response to known classes within each sub-network, the overall parameters used to integrate the sub-networks together do not need to be as precise to produce a correct result. Furthermore, for any result which is dissimilar to all known classes, the inclusion of the 'unknown' class limits the impact of parameter precision; as results below a threshold are simply discarded, there is no need for highly accurate parameter tuning. As such, the parameters within the overall BN are less sensitive due to the sub-network structure and inclusion of the 'unknown' class.

7.5.2 Testing Unknown Risk Detection

To confirm that a 'confidence-based' approach is a suitable method of detecting collision risks, the first requirement is that unknown objects are reliably classed as 'Unknowns'. In order to demonstrate this, the upper half of Figure 7.14 depicts a common aerodrome scene, in which the UAS cannot continue along a taxiway due to another vehicle blocking its path. As the vehicle occupies the taxiway directly ahead of the UAS, it represents an actual collision risk and therefore its detection is critical. The lower half of Figure 7.14 shows the per-cluster results of classification for the same scene, using the full BN. To detect unknown objects, the confidence threshold (U) has been set to 0.5, with 'Unknown' clusters depicted in orange.

From Figure 7.14 it can be seen that the majority of the vehicle has been classified as an unknown, although some elements have been classified as asphalt. Looking specifically at the outline of the clusters, the fact that the vehicle has received multiple classifications indicates that reachability clustering has not produced a object-specific segmentation. This is due to the dissimilarity of certain sections when compared to the vehicle as a whole. Although most of the vehicle is dark green in colour, sections of the vehicle are visibly very different, such as the roof and the windows. The top of the vehicle has simply been misclassified as asphalt, likely due to being a small cluster very close to the horizon.

Although the rear window has also been classified as asphalt, this cannot be considered a misclassification. Looking at the original image reveals that asphalt can be seen when looking directly through the vehicle. Therefore, the cluster classification is correct, albeit not useful. As transparent objects are extremely difficult to detect using a camera, the assumption is made that detecting the rest of the vehicle should be sufficient. As the lower-most clusters have been correctly labelled, depth estimation can be achieved using the point at which the vehicle meets the ground. Therefore, despite some error, the majority of the vehicle has been labelled as a collision risk. As additional collision risks within the scene have also been classified as unknowns (specifically the marking boards to each side of the taxiway) this method is capable of detecting unknown objects.

However, not all unknowns are collision risks. The largest unknown object in Figure 7.14 is the Irish sea, which is visible just below the horizon. As there is no 'ocean' class, this has correctly been identified as an unknown. If large bodies of water were very close to the taxiway (which is uncommon), they would be best avoided using map data. As small bodies of water tend to represent puddles on the taxiway, they have been included within the training data set and should be classified as asphalt. Additional unknowns include the yellow taxiway stop bar, at which the other vehicle is currently waiting. The main reason for this is that the clusters are very small at this distance, lacking sufficient data for confidence in the final classification. As the UAS moves closer (with over 30 meters range between the camera and these clusters) the increase in cluster pixel count should help to correct this misclassification.

Closer to the camera, there are several clusters around the edge of the taxiway that have a low enough certainty to be classified as unknowns. As grass will inevitably grow to overlap the taxiway edge, superpixels which contain both asphalt and grass are more difficult to cluster, resulting in isolated small clusters which are difficult to classify. However, some of the patches in Figure 7.14 are much larger. Reviewing this particular section of taxiway, the original taxiway surface appears to have been slightly wider. Although the centre of the taxiway has been replaced, the original asphalt at the borders is slowly being reclaimed by the grass. As such, these clusters are unique terrain features not found elsewhere, nor in the training set. They are correctly identified as unknown due to the atypical texture which is not comparable to normal grass. Although no specific class exists for this situation, as grass is not considered a navigable surface for this aerodrome it

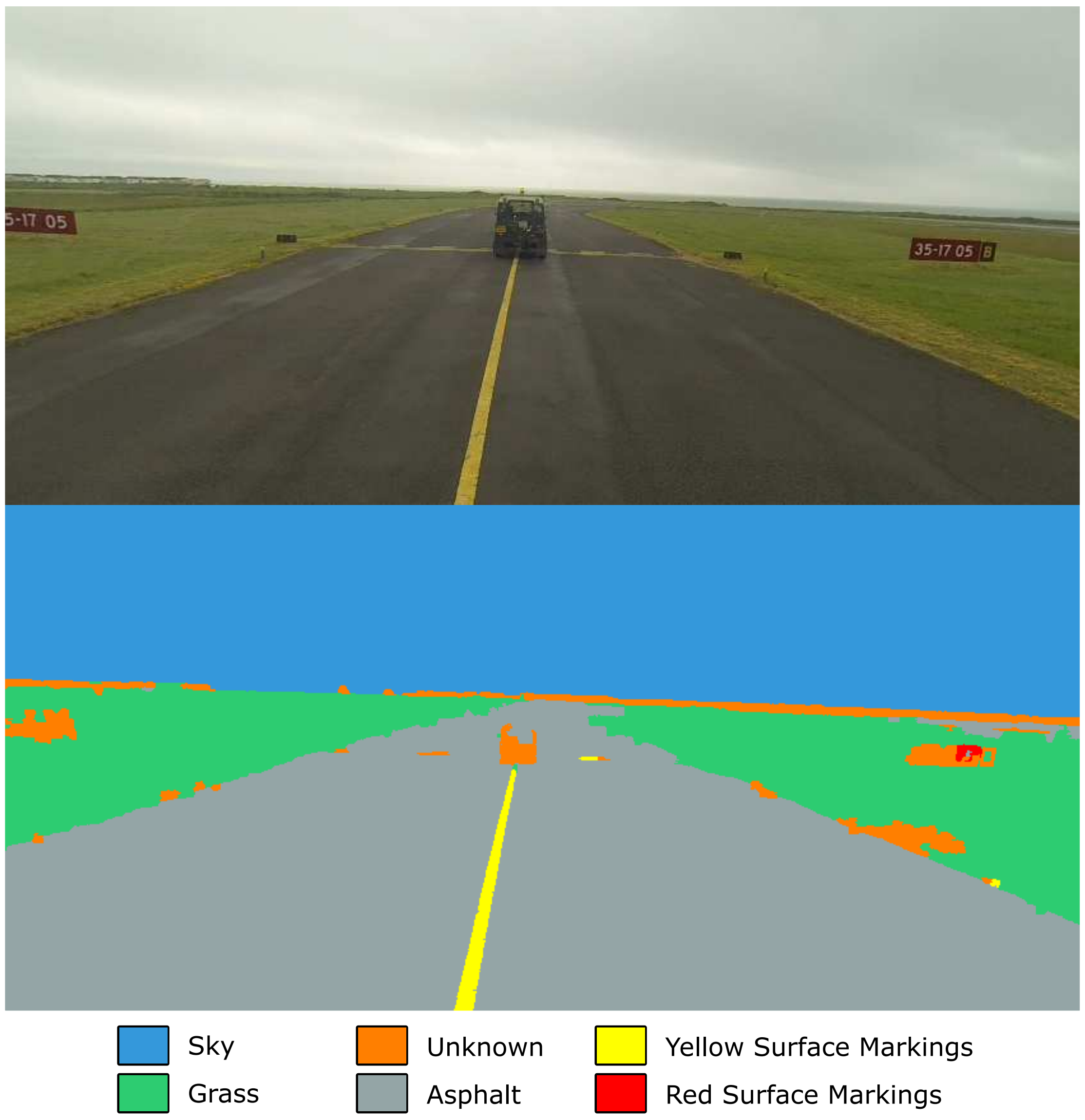


Fig. 7.14 Example of ‘Unknown’ object detection through classification confidence. The uppermost original aerodrome image shows an ground vehicle positioned around 30 metres ahead of the camera at a stop bar. The bottom image shows the most likely classes, with unknown collision risks detected via low probabilities of being any one class.

is unlikely that the UAS would need to venture onto the grass. As such this form of miss-classification is not expected to pose any risk to a taxiing UAS. If a much large training set were available for both the colour and texture classifiers, then it is likely these borderline cases could be resolved, however the eventual combination of visually detected risks and aerodrome map at a later point in the ATS should be enough to permit safe navigation.

7.6 Conclusions

This chapter has covered the final element of the classification process, in which the extracted data is fused and a final classification assigned to each cluster. Beginning with a summary of previous work, the difficulties of fusing different types of information were presented. Assessing why a previous AVS based classification approach was insufficient, the conclusion was drawn that data fusion must not only fuse disparate data types, but must also recognise the relative strengths of the individual data, on a per-class basis. As a probabilistic approach allows practically any form of data to be combined for classification, a BN was selected, in line with the previous publications [24], [25] and [26].

As BNs require evidence to be submitted in terms of probability, the conversion from the original data type was investigated for each data source. Although most sources were simply discretised and submitted as hard evidence, texture data required a more complex approach. As texture data undergoes 'pre-classification', the raw data provided to the BN is already an estimate of class, which forms the basis of the greater BN. However simply selecting the winning class prevents the associated confidence in the classification from being used. As multiple classes can have similar textures, the decision was made to submit texture data as soft-evidence.

However, as both MR8 and LBP classifications were undertaken using BDT based SVM, the grouping of classes resulted in an incomplete confidence set for each class. Therefore, representative probability was generated, based on the distance between the cluster data and the SVM hyperplane, for the last group in which each class was present.

With all evidence converted into probability, the final Bayesian structure was introduced and the concept of sub-networks was explored. Although some nodes were manually defined, other sources of data were too broad for human interpretation, with parameter estimation techniques (specifically MLE) used to obtain the CPD. By designing each sub-network to output the 'per-class' probability, the final node logic could be manually entered into the CPD.

Based on the final result, the winning class was determined based on probability, with the most likely class being selected. Where confidence in the winning class was low, (specifically below 50%) the object was considered to be an 'Unknown' class. As unknown classes are essential for the concept of generic object detection. The ability to detect risks was briefly explored, with promising results. Further results from the full classification process, including the detection of generic risks, will be explored in greater detail in the next chapter.

Chapter 8

Semantic Segmentation Case Study

This chapter presents a case study into the results of the combined segmentation and classification process. Before it is possible to use the classification approach as part of the ATS, the validity of the final output must be assessed. In order for the assessment method to be credible, the data used must be representative of what would be encountered in actual use. Therefore, this Chapter begins with a description of how the test data was acquired.

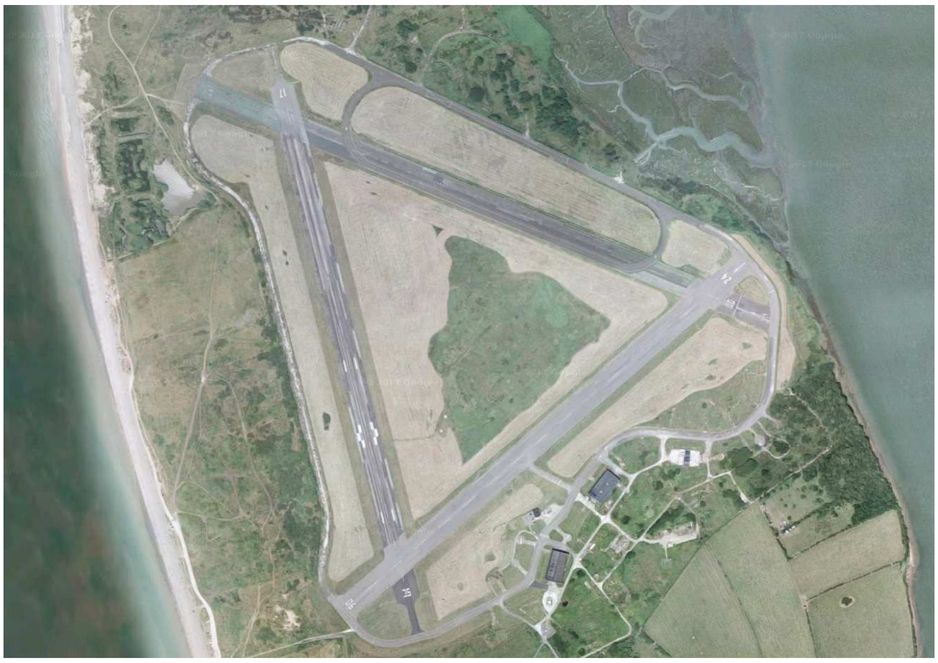


Fig. 8.1 Satellite Image of Walney Island Airport, from which the test data for the case study was acquired. Imagery ©2017 Infoterra Ltd & Bluesky, Landsat / Copernicus, Data SIO, NIOAA, U.S. Navy, NGA, GEBCO. Map data ©2017 Google.

8.1 Data Acquisition

8.1.1 Location

To ensure the test environment was as representative as possible, data was collected from an active aerodrome; specifically Walney Island Airport in the UK (also known as Barrow Island Airport). Shown in Figure 8.1 [45], Walney Airport is a small aerodrome located on an island in the Irish sea. Originally built as a Royal Air Force (RAF) base, the aerodrome is now privately owned by BAE systems, allowing for full control of the test scenario.

The aerodrome map data for Walney Island Airport is shown in Figure 8.2 [109]. Despite originally having three runways, the northernmost runway (Runway 12) and the associated taxiways are now entirely disused, and have fallen into disrepair. As such, the

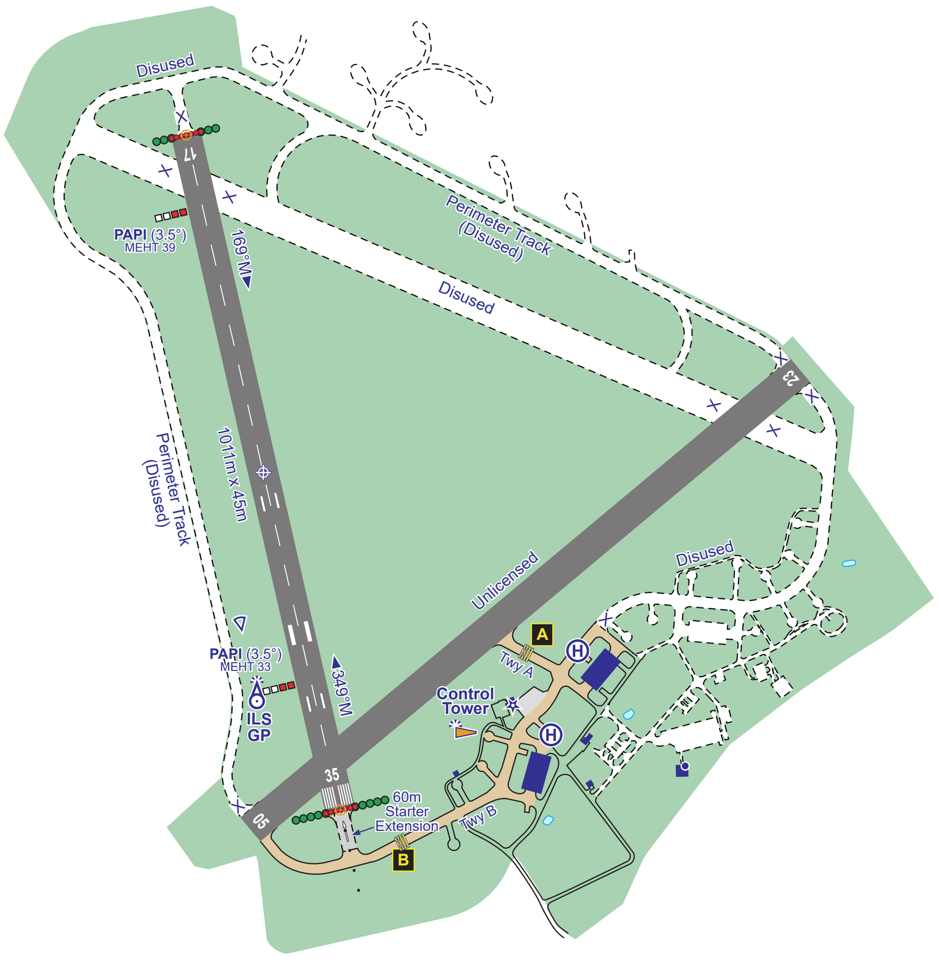


Fig. 8.2 Extract from Aerodrome map of Walney Island Airport. As the disused sections are not viable for taxiing, all data was collected from runway 17 and taxiways A and B. Original map sourced from the Civil Aviation Authority.

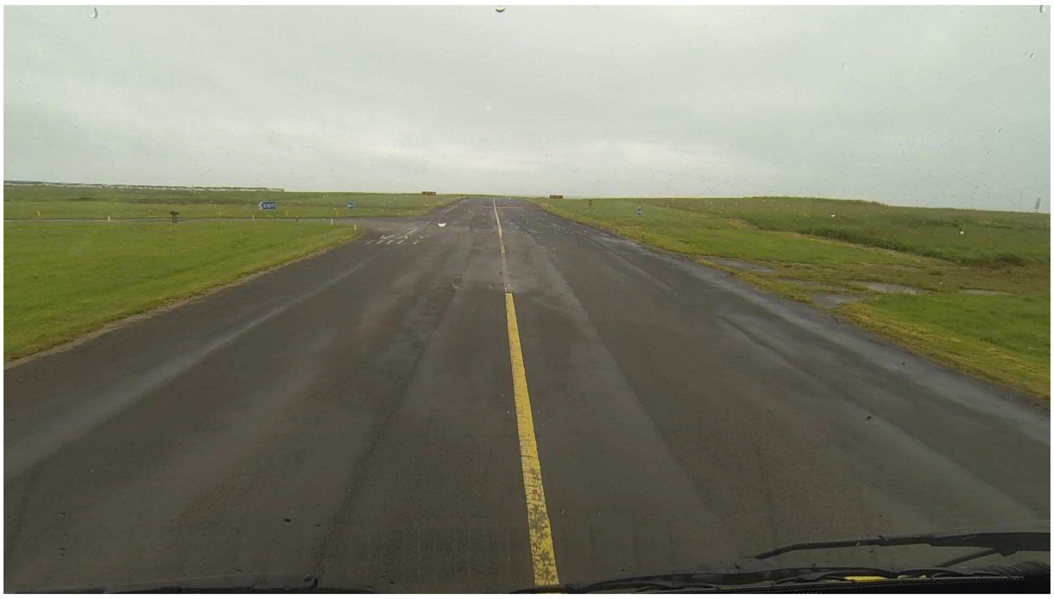
northern side of the aerodrome has not been used for testing. The southernmost runway (Runway 05) is still maintained but is considered ’unlicensed’ and is no longer used during commercial operation. When the aerodrome is in use, Runway 05 is primarily being used as a large ‘taxiway’, linking the hangers to the remaining runway via a more direct route. The westernmost runway (Runway 17), is the only active runway and therefore has been used for all runway image capture. Finally, at the far south end of the aerodrome, a small number of hangers and other buildings are linked via a section of curved taxiway, which is also used by ground vehicles.

Although small, the aerodrome offers a high-variety of terrain features, such as surface markings and asphalt. Due to the long operational history, the runways and taxiways within the aerodrome vary in accordance with the era in which they were last overhauled. For example, Runway 05 has not been active for several decades, resulting in the surface markings being both worn and out-of-date (the runway is missing the "precision-instrument" landing markings associated with ILS, which are present on Runway 17). For taxiways, some of the surfaces near to the hanger area are relatively new, with clearly defined surface markings similar to those found at newer aerodromes. However, just a short distance away the surface of the same taxiway becomes aged and worn, with inconsistent surface textures where repairs have been made. As such, Walney airport provides a wide variety of surface types, in excess of what is found at newer aerodromes, providing a significant challenge to the classifiers.

8.1.2 Conditions

As Walney Island airport is wholly owned by BAE Systems, it was possible to gain exclusive use of the aerodrome for testing. However, although Walney Island is not a large commercial aerodrome, there are regular flights and many other aerodrome users which limit the amount of time available and increase the cost of testing. As such, the aerodrome dataset was created over the course of a single day, (on the 7th August 2015).

Although the short testing window has inevitably limited the range of conditions captured during the test, weather and lighting conditions did vary quite significantly, reducing the risk of over fitting. Testing began at the conclusion of a rain shower, with the initial footage obtained in damp conditions, with puddles and water-droplets visible to the camera; as shown in Figure 8.3a. Although much of the testing was then conducted in overcast conditions, clearer skies had emerged by the end of the day, allowing a significantly different data set to be obtained; as shown in Figure 8.3b.


(a) Damp conditions at the start of testing.

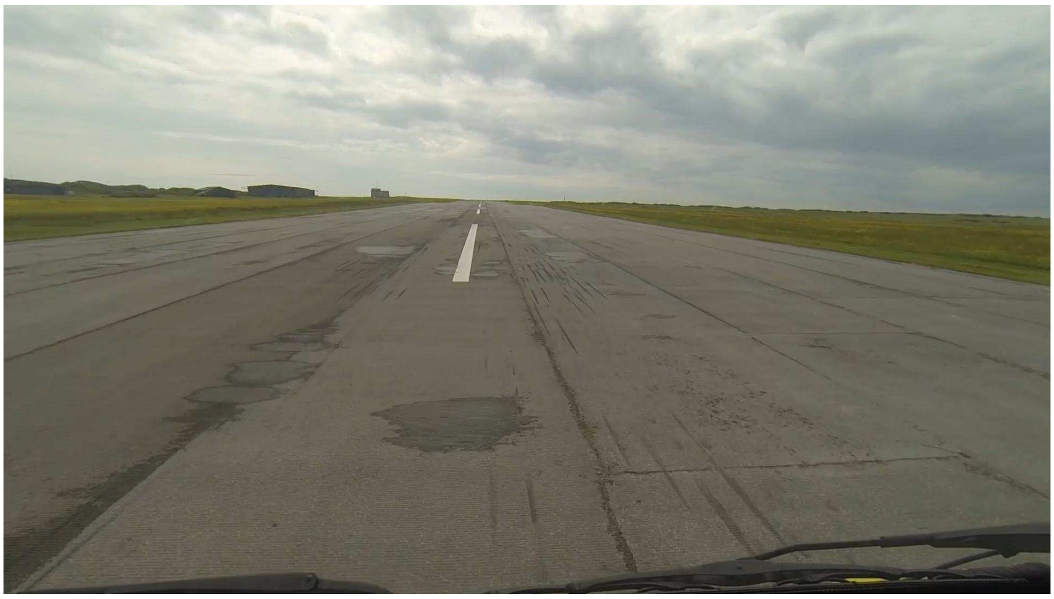
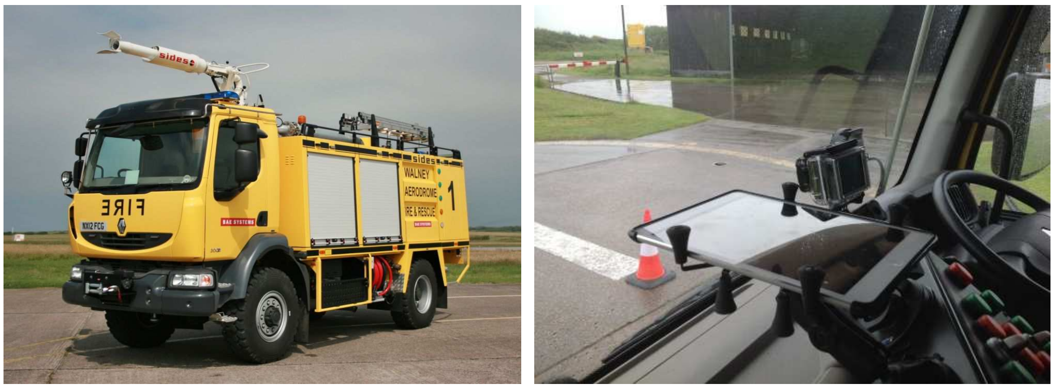

(b) Clear conditions at the end of testing.

Fig. 8.3 Comparison of testing conditions at the beginning and conclusion of the test day

As overcast and clear skies differ in both colour and texture, this provided a much better cross sample of the sky class, making sky classification more robust. As the horizon detection algorithm is also used to provide additional weighting to sky classification, even this limited data set should be sufficient to classify sky clusters. For all other classes, by using image features such as texture and separating colour components using the HSV colourspace, the effects of illumination changes should be minimal. Therefore, the wide



(a) Test vehicle used for data capture. (b) Camera set up within test vehicle.

Fig. 8.4 Test vehicle and camera set up used for data capture.

range of in-class surface types combined with the varied weather conditions has provided a highly challenging and realistic data set.

Although the amount of data within this training set is insufficient for classification to be robust at any other aerodrome, the data collected here could form part of the training set for the eventual system. As the ideal set would consist of data collected at a great number of aerodromes (throughout the course of a day and in various weather conditions) this data is highly appropriate.

8.1.3 Equipment

Despite the intention to use the ATS on the BAe Jetstream surrogate UAS, for practical purposes testing was achieved using a terrestrial vehicle. To replicate both the height of the Jetstream camera as well as the 'rolling motion' typical of aircraft turning when on the ground, the aerodrome fire engine was used, as shown in Figure 8.4a. A benefit of using this vehicle is the inclusion of a roof mounted GPS receiver, replicating the set up found on most UAS.

To ensure a representative image was captured, the camera height and mounting position were similar to that on the Jetstream, with the camera mounted on vehicle centreline, as shown in Figure 8.4b. The camera selected was a GoPro HERO3 Black, which is well suited to this task; primarily designed to be used for dynamic motion video in outdoor conditions, it features a very short exposure time to eliminate blur, and a large dynamic range which allows the vehicle to head towards the sun without over-exposure. The output video was captured at a resolution of 1920 by 1080 pixels at 30fps. As already mentioned in a previous chapter, the effects of lens distortion were rectified using the manufacturers software, allowing both horizon line estimation and depth extraction to function without issue.

To provide additional data for later chapters, the GPS position and inertial sensor readings were recorded, in addition to the video footage. As the fire engine lacked any form of inertial sensors, a tablet computer was mounted directly below the camera to capture this information, in addition to the GPS position from the receiver on the vehicle's roof.

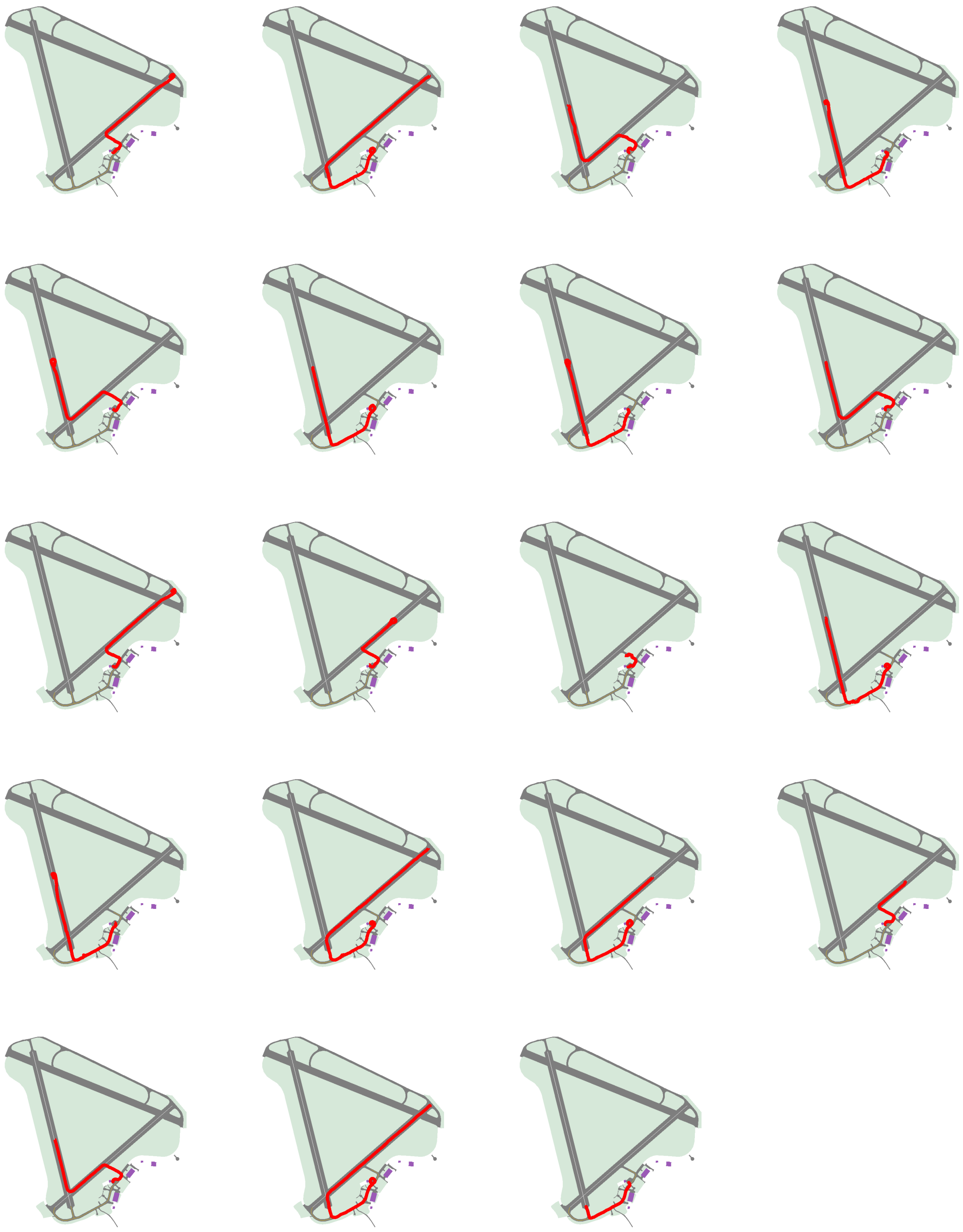


Fig. 8.5 GPS position data (shown in red) for each of the nineteen test runs, overlaid onto the map of Walney Island Airport. Aside from the 'disused' runway and taxiway which were not accessible, the entire aerodrome has been sampled for testing.

8.1.4 Methodology

Testing was conducted in a series of ‘runs’, most of which were intended to mimic a typical taxiing activity for an aircraft. To include as much footage as possible, a small number of runs took unconventional paths around the aerodrome. Figure 8.5 show each of the nineteen routes taken by the vehicle during testing, with the entire aerodrome being covered.

In addition to moving the camera vehicle, other vehicles were moved about the aerodrome to provide test footage for collision risk detection. As no aircraft were available, a ground vehicle was used as a substitute, as previously described in Section 7.5. As ground vehicles are typically smaller than aircraft and are quite common on taxiways, this was considered an acceptable substitution.

8.2 Comparative Data

After the data acquisition was complete, additional work was required to ready the data for classification. This included the creation of ground truth images, as well as deciding upon the format of the results used for this chapter.

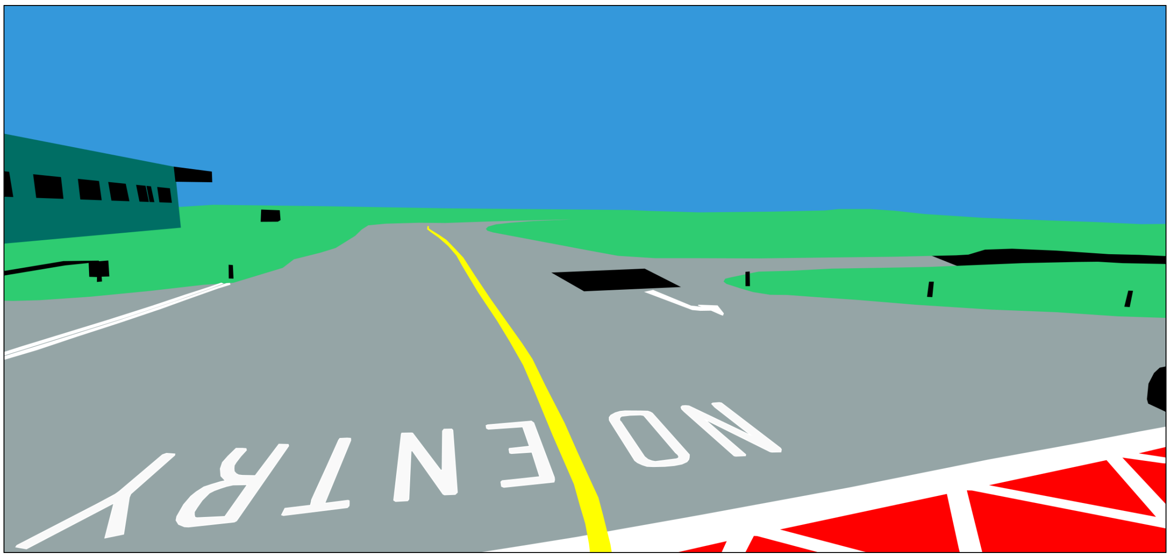
8.2.1 Ground Truth

After the data acquisition is complete, a final stage is required before the data can be used. In order to train the classifiers (and for comparison against the classifier results), the raw data must be associated with its relevant class. To achieve this, ‘Ground truth’ images need to be created. This process begins by selecting a suitable range of images from the entire set. From the nineteen taxiing runs, a smaller data set of 100 images was extracted:

- Fifty images were selected entirely at random
- Fifty images were manually chosen to ensure sufficient examples of each class would be present in the data set

As a human should achieve a near optimal result, each of the 100 hundred images underwent manual segmentation and classification. To avoid introducing data which could hinder classification, only regions which belonged to expected classes were included. Although the segmentation of the training images could be achieved automatically (using SLIC segmentation and reachability re-clustering) training is best achieved using clear examples of each class. To prevent including data which did not improve the classifier (such as sampling from regions of asphalt with moss, which incorrectly associates green with asphalt)Any region whose contents did not belong to an expected class were simply labelled as ’unclassified-regions’ and not included in the training process. A typical ground truth image is shown in Figure 8.6.

The one hundred image pairs (the selected images and their associated ground truths) serve two purposes; training and testing. As such, an 80:20 split has been adopted, with 80% of the images used for training (both MR8 and LBP texture classifiers, and the full BN). As the methods of training have already been established in Chapters 6 and 7 they will not be explored here. The remaining 20% of the images which have not been used for training are used for validation and provide the results for this chapter.



⬜ Asphalt	🟦 Sky	🟨 Yellow Lines	🟩 Building
🟩 Grass	⬜ White Lines	🟥 Red Lines	⬛ Not manually classified

Fig. 8.6 Example of 'Ground truth' image used for classifier training, for the original aerodrome image shown in Figure 8.7.

8.2.2 Result Format

To evaluate the entire classification process, the results from the complete BN are compared to the manually created ground truth images. As the BN should provide a significant improvements over the previous texture only methods, the classification results from the individual MR8 and LBP texture classifiers are also presented.

Although each of the other sub-networks within the BN also provide information on each class, they cannot be considered standalone estimates. Due to many of the results providing equal weighting to multiple classes, direct comparison between the output of the each sub-network and the final BN result is difficult. As these additional sub-networks are responsible for the change in result between the texture classifiers and the complete BN, any improvement in the output is assumed to be due to the additional sources of information.

Using the twenty images specifically reserved for testing, the results will be compared in numerous ways; including per class, per cluster, per pixel and the overall classification performance. However, prior to this, a single image will be used to demonstrate each stage of the classification process, and assess how they impact the final result.

8.3 Single Image Classification Review

In order to accurately assess the overall performance of the classifiers, data needs to be drawn from a large number of images. However, this form of mass data cannot demonstrate the data progression between each stage. Therefore, an example image will be presented here and used to demonstrate the majority of the classification process; assessing how each stage impacts the final result.

To be able to determine the accuracy of each classification stage, the example image has been selected from the twenty images within the data test set so that ground truth is available. However, as these images were selected at random, many depict parts of the aerodrome which are limited in the number of classes present. Therefore, an example image has been specifically selected which contains regions of all known classes, as shown in Figure 8.7.

8.3.1 Segmentation

Although this chapter mainly studies the output from each of the classifiers, the segmentation process is also of great importance. As all classification results are based on the initial segmentation, errors introduced at the earliest stage will remain in the final classification result. As stated in Chapter 5, separate segmentation and classification stages were primarily implemented due to the need to segment unknown objects, which can be difficult to achieve using classifier led segmentation. However, by using separate stages,

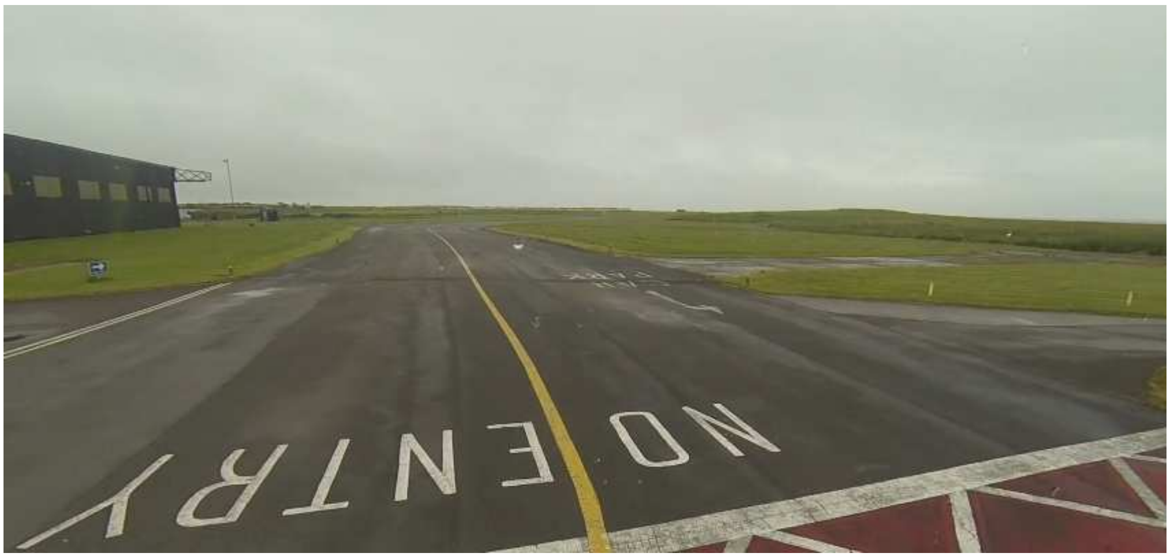


Fig. 8.7 Example image used to compare classifier results.

there are effectively two different types of error which occur: misclassification and segmentation error. Segmentation error occurs when a cluster contains multiple classes. This may be a result of reachability clusters erroneously combining superpixels from different classes, or may occur at the superpixel level, if the boundary between two regions is sufficiently gradual.

The segmentation results for the example image are shown in Figure 8.8. (Although the segmentation process begins with a superpixel over-segmentation, it is difficult to draw strong conclusions from superpixels alone. Therefore only the final segmentation result, produced by reachability clustering, is shown). The superpixel segmentation and reachability clustering approach was chosen as it is capable of precise cluster border extraction, whilst also maximising the size of each cluster provided to the classifier. As such, the expected output should be large continuous clusters, with precise boundaries. However, as throughout the entirety of this work, the reachability clustering threshold has been set to produce an over-segmentation, in order to ensure that under classification does not occur.

From Figure 8.8 the effects of the oversegmentation are clear, as many additional clusters exist when compared to the human defined ground truth, shown in Figure 8.6. Although this works well in preventing segmentation error through reachability clustering, it cannot prevent superpixels from forming on imprecise regions between borders. As such, a number of smaller clusters have formed within the image between the taxiway

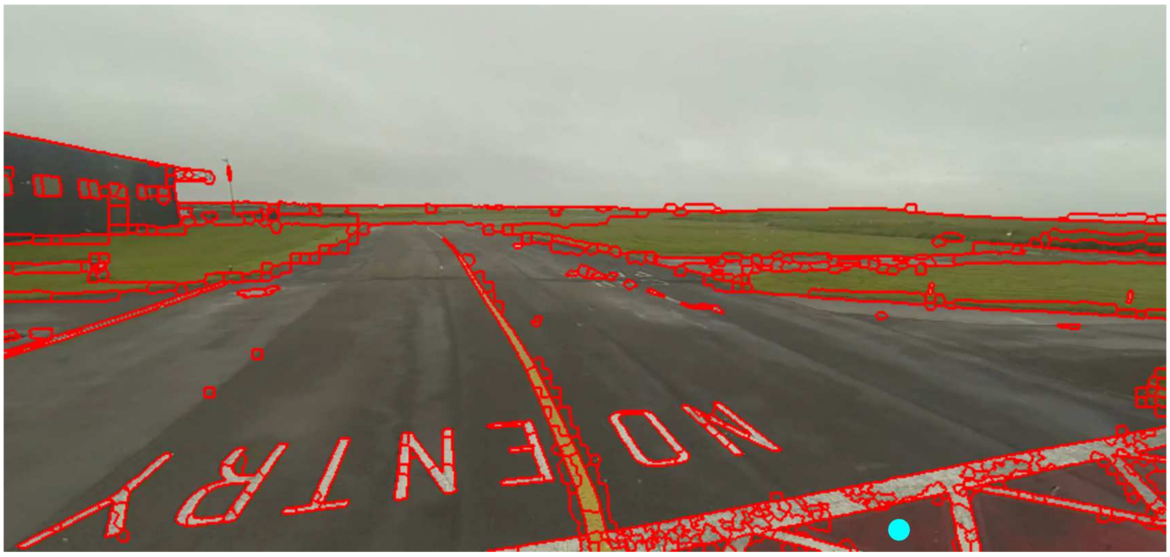


Fig. 8.8 Example image having undergone superpixel segmentation and reachability clustering.

and the surrounding grass. Depending on the content of each superpixel, these may be classed as taxiway, grass or unknowns in the final result.

Despite this, it can be seen that there is minimal segmentation error where the underlying features in the image are clear, with cluster boundaries closely following the surface markings. Although the surface markings are also over-segmented, the original border lines within the image are well preserved.

For this image, the final segmentation is considered acceptable, with only a very small number of clusters containing any form of segmentation error. Although the end results cannot be deduced from the segmentation stage alone, the superpixels bordering the taxiway are the most likely to be falsely classified as 'unknowns'. As these borders are visibly imprecise, as well as being both distant and not in the direct path of the UAS, they should have minimum effect on the vehicles taxiing (especially as the errors are likely to be rectified as the camera gets closer).

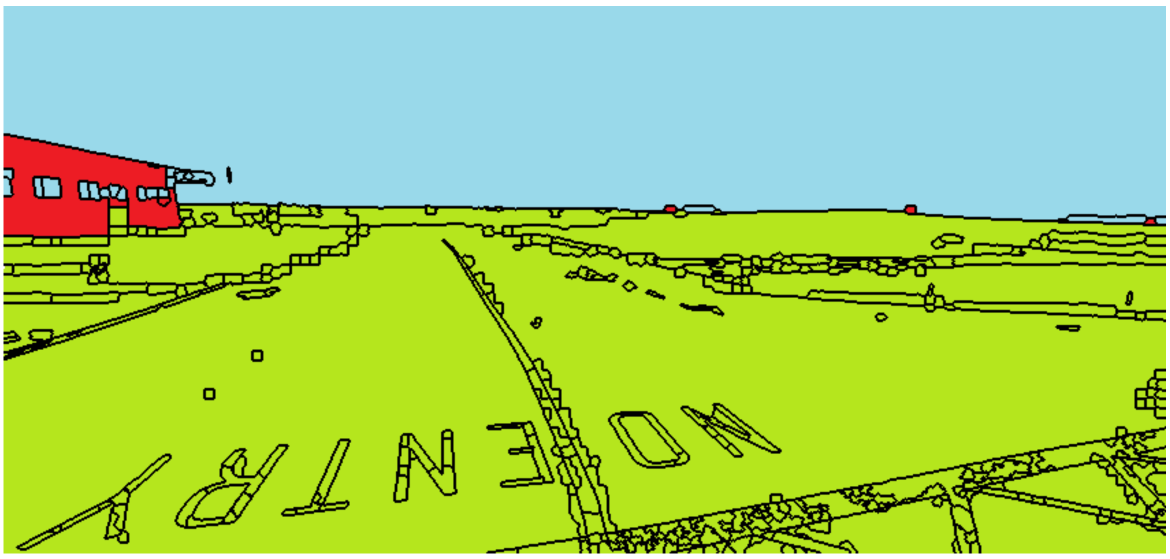
Following on from segmentation, each of the data extraction techniques are applied to the image. Of a final note, the cyan dot in the bottom of Figure 8.8 has been placed to highlight a particular cluster, for which the classification results will be discussed.

8.3.2 Horizon Intercept

One of the most straightforward forms of data to assess is the Horizon Intercept Class [*Hoz*]. As the horizon considered to be is a twenty pixel tall band, only clusters which lie on both sides of that band (or wholly within it) receive the *Intercept* category. For the example image, only a few clusters fall into this category. Figure 8.9 shows that the *Intercept* category includes the building at the left side of the image, and three small clusters on the horizon. Reviewing the original image in Figure 8.7, these results are considered to be nearly entirely accurate.

Clusters which have somewhat erroneous classification include the windows on the building to the left of the image. As the windows within the building rise out of the horizon band but not below, they are considered sky. However, as the bottom of the building is entirely classed as *Intercept*, this should have minimal effect. Finally, although one of the clusters identified on the horizon is a building, the other two are not. However, as these objects are very far from the camera, there is not need for precise classification. As such, the horizon detection results are unlikely to be a source of misclassification.

As the example cluster (highlighted in cyan in Figure 8.8 is far from the horizon line, the horizon intercept classification of *Below* is correct.



🟩 Below 🟥 Intercept 🟦 Above

Fig. 8.9 Example image having undergone horizon intercept categorisation. The *Above* category is shown in light blue, the *Intercept* category is shown in red and the *Below* category is shown in pale green.

8.3.3 Normalised Relative Luminance

As with the horizon intercept data, the evidence provided to the Surface Marking Class [$Line$] is also categorised into three states, based on the brightness of the clusters. NRL data was specifically intended to help detect white and yellow surface markings. As such, Figure 8.10 can be considered a near ideal example, with nearly all yellow and white markings segmented from the other clusters using NRL values alone. Although small clusters within the surface marking are occasionally categorised as Low, this is due to the wear on the paintwork producing a broken surface, which is far less reflective than surface markings should typically be.

The only other objects which have received $High$ or Mid values for [NRL] consist of a puddle (which is highly reflective and does not pose a risk) and reflective markers defining the edge of the taxiway. As [NRL] is strongly associated with surface markings in the BN, the presence of 'upright' objects which share high [NRL] values is of some concern. However, as the markers have different colours and textures (as shown in Figures 8.11 and 8.12) they are not classified as surface markings in the final result.

Of note, the example cluster (highlighted in cyan in Figure 8.8) has received a Low [NRL] classification, despite being a surface marking. This is due to red surface markings having low reflectivity and therefore very low luminance when compared to the white

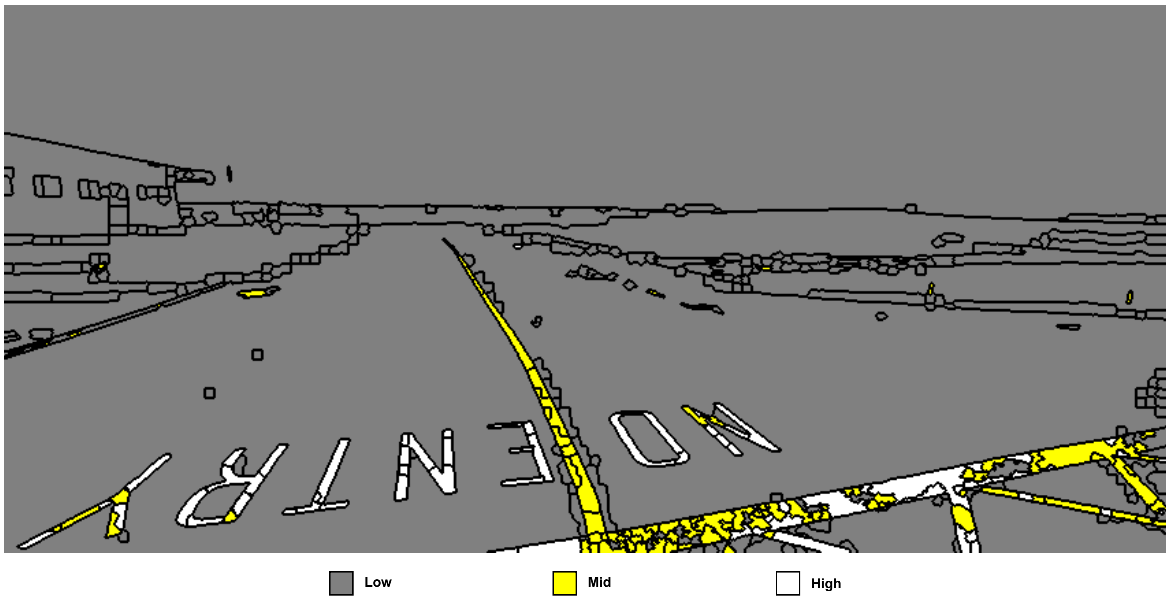


Fig. 8.10 Example image having undergone [NRL] data extraction. The $High$ category is shown in white, the Mid category is shown in yellow and the Low category is shown in grey.

and yellow aerodrome paint. As low variance is typical of red surface markings, this has been taken into account within the final classification node [$Class$] of the BN. As such, it should have no impact on the final classification result.

8.3.4 HSV colour

The HSV Colour Classification [Col] is the first sub-network to provide an actual estimate of class. As such, the final results are directly comparable to the ground truth image shown in Figure 8.6. However, even without the ground truth for comparison, the relative strengths and weaknesses of the colour class estimation are apparent from a brief inspection of Figure 8.11.

Overall the image has been mostly classified correctly, with elements such as the grass border, taxiway and sky, all appearing as expected. However, significant misclassification has also occurred. Towards the top left of the image, the entire building (shown in Figure 8.7) has been misclassified as 'asphalt'. Although this does seem like a significant issue, it is to be expected. As in-class variation was too great, the 'building' class has not been included within the colour classifier. As such, misclassification is inevitable, and for this reason colour data does not effect the building class estimate in the final combined node of the BN.

The other significant misclassification within the image are the white surface markings which form the words 'NO ENTRY'. Unlike the building class, white surface markings can be associated with colour (as the name would suggest) and therefore should be classifiable using HSV data. However, although other white lines are correctly classified, only

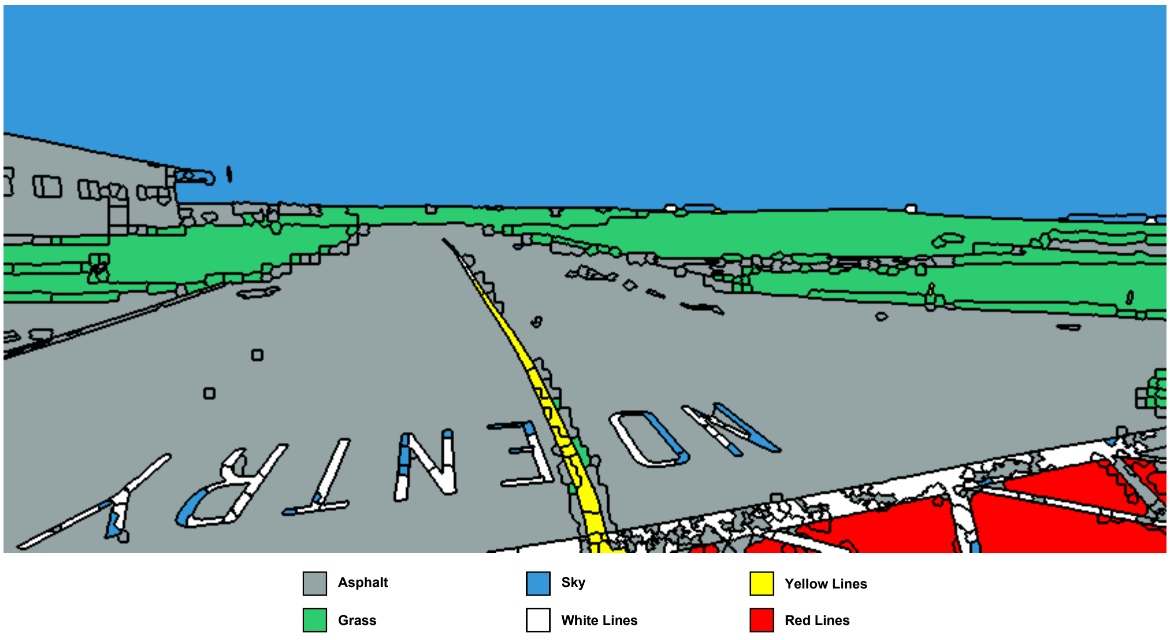


Fig. 8.11 Results of colour classification sub-network [Col] when applied to the example image shown in Figure 8.7.

around 65% of the pixels within the lettering are labelled as white lines, with the rest being classified as either sky or asphalt.

Although the over-segmentation does limit the amount of data provided by each cluster, as the colour data is simply the mean value of all pixels, the size of the cluster should not have significant impact on the end result. Instead, the main problem is that overcast skies are so similar to the surface markings that the two are indistinguishable on colour alone. As colour cannot be directly linked to each class, the regions of colour space which correspond to each class overlap. As this would have been vastly time consuming for a human to manually input, parameter estimation (i.e. MLE) was used to establish the CPD which provides the final estimate of class. As such, it is very likely that multiple class have very similar probabilities for many of the 2161 possible colours.

During the development of the BN, the difficulty found in using colour data to classify surface markings (which would seem to be apparent on colour alone) was the primary motivator for the development of NRL, and one of the reasons for the inclusion of horizon data within the BN. Looking ahead, the misclassification is resolved in full BN result (shown in Figure 8.13). As the lettering is well below the horizon line, the sky class will receive a strong reduction in probability, with the contribution from the $[NRL]$ ensuring that the letting is classified correctly.

Of note, the example cluster (highlighted in cyan in Figure 8.8) has been correctly classified as a 'Red Surface Marking'. As this colour is the primary method of distinguishing the surface markings (aside from NRL) this data will be highly influential in the final result.

8.3.5 Combined Texture

As established in Chapter 6, texture is the primary source of information for classification, with the addition of other data intended to refine the result. Due to the comparative benefits of LBP and MR8 texture classifiers having already been assessed in both Chapters 6 and 7, the results of the combined texture node are presented here for brevity. As the two sources of texture data are combined within their own texture sub-network, this allows a direct comparison to the results of the other sub-networks previous discussed above.

Unlike the colour classifier, the combined Texture Classifier node [*Tex*] has been trained using data from all classes, including buildings. Comparing the texture based results in Figure 8.12 to the colour based result in Figure 8.11, the most obvious improvement is that the building has been (mostly) classified correctly.

In addition, despite colour data being the obvious method of differentiating the coloured surface markings, the texture-only classification actually outperforms the colour classifier in this task, for both the white lettering and yellow taxiway centreline. This is mainly due to these particular surface markings being well maintained; with fresh paint having a strong and well defined texture which is clearly sufficient to distinguish the surface type. However, although this is successful, it should be considered a case of over-fitting

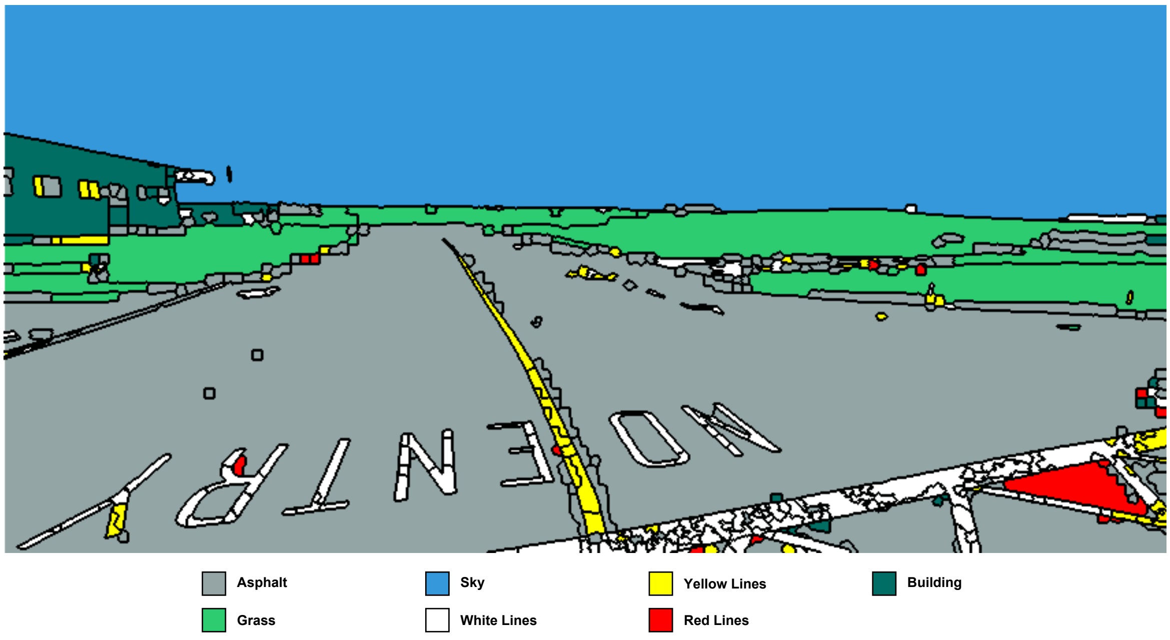


Fig. 8.12 Results of combined texture classification sub-network [*Tex*] when applied to the example image shown in Figure 8.7.

as other aerodromes are unlikely to have the exact same paint. Therefore, the NRL and colour data should be more useful for generic terrain identification for paints with different textures.

This is immediately apparent when returning to the example cluster (highlighted in cyan in Figure 8.8). Unlike the white and yellow surface markings, the red-surface marking has been misclassified as asphalt. Although the yellow centreline and white lettering are new, the red surface markings are very old and worn, resulting in a very similar appearance to asphalt; resulting in the texture misclassification. As such, the overall result will depend upon the final CPD to correctly associate the surface type based on colour rather than texture.

8.3.6 Final BN Classification

Figure 8.13 shows the final BN classification results, based on the original image shown in Figure 8.7. As with colour and texture data, comparison to the Ground Truth image (shown in Figure 8.6) can be used to determine how accurate the result is. However, in addition to each of the expected classes, the results also include clusters of 'unknown' class, where confidence is not high enough to determine an outright winner. As these are of particular importance, they will be reviewed using an additional figure below.

Focusing on classes with have received an estimate of class, it is clear that many of the clusters which were misclassified in the individual sub-networks have been correctly classified using the full BN. For example, the building in the top left of the image was completely misclassified in the colour classification. However, as colour data has no effect on the probability of the cluster being a building, the texture and horizon intercept data were sufficient for the building to be correctly classified. Furthermore, the white lettering on the taxiway has received a far better classification, although a small number of clusters are still misclassified.

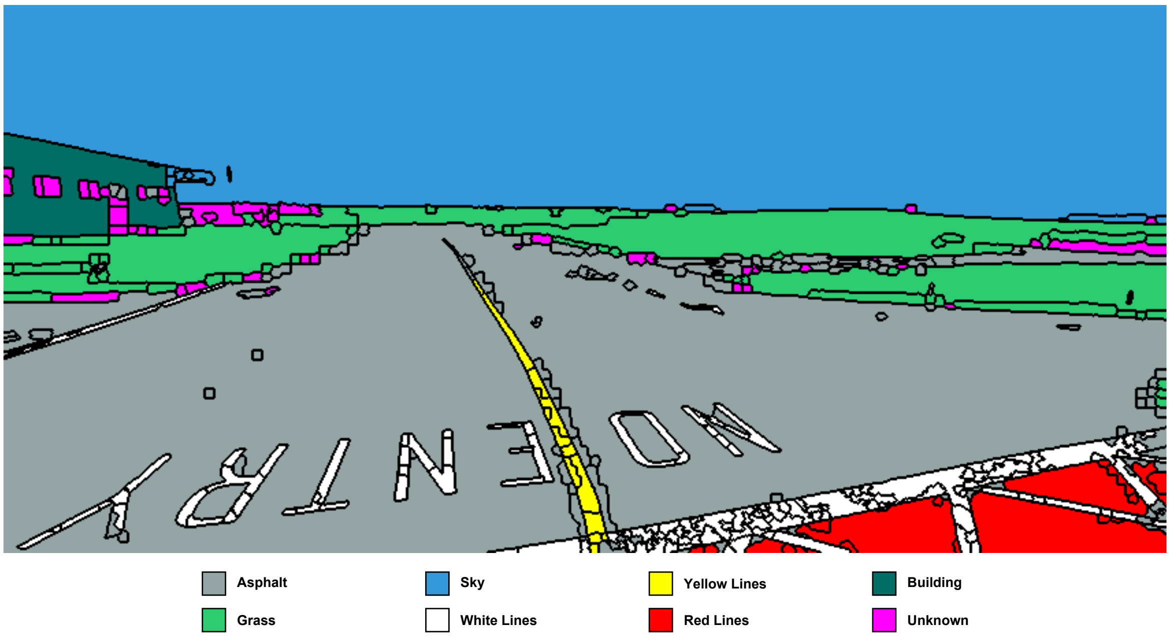


Fig. 8.13 Results of full Bayesian network [$Class$] when applied to the example image shown in Figure 8.7.

Returning to the example cluster (highlighted in cyan in Figure 8.8) the final BN result can be assessed in more detail. Moving through the sub-networks:

- The Horizon intercept state [Hoz] identifies the cluster as being $Below$ the horizon. As such it has a zero probability of being a sky cluster, but all other classes remain equally likely.
- The Surface Marking Class [$Line$] has an NRL state of Low. When combined with the distance state of $Close$, the cluster receives a near-zero probability of being a white or yellow surface marking (as shown in Table 7.7).
- The Texture Class [Tex] provides a direct probability of belonging to each class. Although the surface is painted, the region is extremely worn with the surface more indicative of asphalt. As such, the combined texture classification approach identifies asphalt as having the highest probability of a match, albeit with only with only 52.92% confidence.
- Finally, the Colour class [Col] is strongly weighted towards being a red surface marking, based on hue, with a very high probability of 0.9592.

Despite the texture classifier identifying asphalt as the most likely class, the BN class has been correctly identified as 'Red-surface marking'. As the main contributors to this result are colour and texture data, the posterior distributions for the Colour Class sub-network and Texture Class sub-network, are shown alongside the final BN probability in Table 8.1.

Within the CPD for the final node [$Class$], each of the surface markings are heavily weighted towards the results of the colour classifier. As the colour classifier identified that the cluster was most likely to be red surface marking (with a very high probability of 0.9592), this has a strong effect on the final results; with a 76.04% probability that the final result is a red surface marking, compared to just 19.05% for asphalt.

At this point, the benefit of using soft evidence becomes clear. Although colour data was the primary reason for the correct classification, had the winning class from the texture classifier been submitted as hard evidence, the texture sub-network would have provide a 100% probability of the result being asphalt. Although weighting within the final node could still have adjusted the result based on colour, a near doubling in texture confidence (from just 52.92%) would have made it far less likely to correctly identify the cluster.

Class	Colour	Texture	Final Class
Asphalt	0.0408	0.5292	0.1905
Grass	0.0000	0.0271	0.0008
Sky	0.0000	0.0225	0.0000
White line	0.0000	0.1043	0.0204
Yellow Line	0.0000	0.1036	0.0203
Red Line	0.9592	0.0932	0.7604
Building	-	0.1202	0.0000

Table 8.1 Marginal posterior distribution for HSV colour sub-network ($Colour$), the combined texture sub-network (Tex) and the complete BN ($Class$) for the highlighted cluster shown in Figure 8.8.

8.3.7 Discrepancies between Classifier and the Ground Truth

Unlike the highlighted cluster, a number of the other clusters within the example image did not match their expected class from with the ground truth image. Figure 8.14 highlights the clusters where ground truth and the BN results differ. As is apparent from this figure, there are several different reasons for the discrepancies.

One reason which is not an issue with the classification process, is the presence of regions of uncertain class within the original image. For example, the surface markings on the right side of the taxiway (near the very centre of the image) are at such an oblique angle to the camera that the edges of the clusters are difficult to define. To avoid intro-

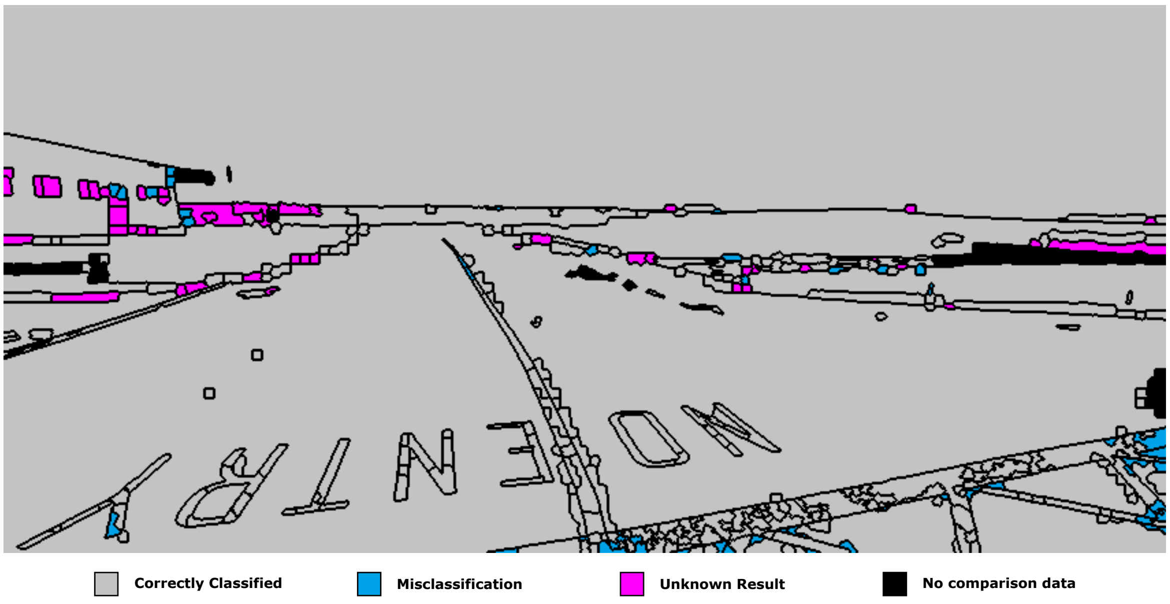


Fig. 8.14 Discrepancies between the Bayesian network result and the Ground Truth image

ducing data from the wrong class into the training process, the entire area was labelled as as ‘unclassified-region’ and not used. As such, there is no method of judging the classification result either correct or incorrect; therefore clusters without a manual class for comparison are simply shown in black in Figure 8.14.(Despite this, the classifier has identified these clusters as asphalt in Figure 8.13, which would seem to be an appropriate result.)

The second cause of discrepancy is uncertainty, where clusters with less than 50% confidence in the final result have been categorised as ‘unknown’. These ‘unknown’ class clusters are shown in magenta in Figure 8.13. As generic object detection relies on unknown clusters, it is important that clusters which *are* an unexpected collision risk fail to gain the necessary confidence to be labelled as a class. At the same time, as the collision risk detection strategy is to avoid all unknown clusters, false labelling terrain clusters as collision risks would have a direct influence on the course of the UAS. Therefore it is vital to assess if the ‘unknown’ classification of each cluster can be assumed valid.

From Figure 8.13 it can be seen that a significant number of clusters have been classed as unknown, almost entirely in the upper half of the image. The main source of unknowns within the image is the building in the upper left. As this is a collision risk, parts of the building which were not sufficiently close to the building class should be identified as unknowns. For this building in particular, each of the windows has been labelled an unknown. This is almost certainly due to the windows not being included within the ‘building’ training set. However, as all parts the building represent an unknown and therefore a collision risk, this classification is correct.

A small number of unknowns have also occurred very close to the horizon line. Although simply labelled as ‘grass’ in the ground-truth, when examined in detail there are numerous distant objects close to the horizon. Although this was detected in the segmentation process, the size of each superpixel is too large to individually contain such distance objects. Although this could be considered misclassification, the extreme distance between the camera and the object not only makes the actual class difficult to determine, but also implies that these objects will never impact the safety of the taxiing aircraft. As such, these unknowns are acceptable.

Finally, unknown results have occured along the border between the taxiway and the grass. This is mainly due to inconsistencies in the border, where grass and taxiway have been included with a single cluster, due to a segmentation error. This is less likely to occur closer to the camera, due to a more obvious border between grass and asphalt. As the grass/asphalt border is a common case, but has minimal effect on the taxiing process, it is not considered an issue. Furthermore, as the segmentation is undertaken independently on each frame, without a strong underlying feature it is unlikely that the segmentation error will persist, with unknown clusters only becoming collision risks once they have been consistently identified.

The final discrepancies are misclassification, in which the BN has provided a confident result in direct contrast to the manually selected class. Although there are no unknown results on the taxiway surface for this image, there are clusters which have been misclassified. Depending on the exchange of object type, this has the biggest potential for danger, as the ATS may have entirely missed a collision risk.

However, looking at this image, the majority of mis-classifications occur in the highly over-segmented surface markings. As with the example cluster, the surface markings are so worn as to be difficult to classify correctly. As the discrepancies are all between navigable types of ground, this issue is not considered too great and could likely be resolved with a larger dataset. As Walney Island Airport has only a limited amount of surface markings (other than the centre line), very few images within the training set contained red from which to train.

Overall, the segmentation and classification of this image is suitably accurate for taxiing. The vast majority of clusters (and pixels) have been correctly identified, with the BN showing significant improvement over each of the internal sub-network results. Finally, this image is viewed in isolation, whereas the collision risk system will process multiple images over time. In doing so, the small discrepancies should be resolved with the help of 'neighbouring' frames.

8.4 Full Dataset Results

Having reviewed the application of the entire classification process to a single image, the overall accuracy of the BN can be more thoroughly established by using the full test data set.

8.4.1 Cluster error and pixel error

In the example image above, error was most easily discussed by referring to the individual clusters created by the segmentation process. This would seem reasonable as classification is performed on a per-cluster basis. However, as clusters can be drastically different in size, the significance of the final result can be lost if only the cluster based result is reviewed. For example, a image could only contain two grass clusters; one taking up most of the terrain and one which is just a single superpixel. If the single superpixel were to be misclassified, the per cluster accuracy of the result would be only 50%, which greatly skews the perceived quality of the results.

When referring to segmented images, there are two way to statistically represent the result; per cluster or per pixel. Table 8.2 shows the 'per cluster' and 'per pixel' error for each of the results applied to the entire data set. Before any numeric review is undertaken, simply comparing the cluster error percentage to pixel error percentage shows that clusters conceal the actual result. As smaller clusters are more likely to have difficulties in classification whilst being more numerous, the results can appear to be extremely poor, despite the majority of the image having been classified correctly. Therefore, the classification results for the rest of this section will be provided in terms of pixels.

8.4.2 Comparison of Classifiers - Total Errors per Classifier

Table 8.2 shows the relative performance across the entire data set for both the texture-only classifiers, in addition to the full BN. A comparison between the results of the BN

Classification Method	Pixel Error	Cluster Error
MR8	5.12%	57.67%
LBP	6.29%	52.81%
Bayesian network - hard evidence	1.48%	19.82%
Bayesian network - soft evidence	0.72%	13.40%

Table 8.2 Comparison of classification error between the MR8 and LBP texture classifiers and the full Bayesian network.

when using hard evidence instead of soft are also shown.

Beginning with the comparison between the two types of evidence, it is clear that using soft evidence within the BN has significantly improved the final result. Reiterating the conclusion drawn from the single cluster example in Section 8.3.6, by submitting the texture class as the confidence in the result, the texture node $[Tex]$ is typically far less dominant within the overall BN. This allows other sources of data, such as colour, to improve the final result. As such, all comparison to the BN going forward will refer to the full BN using soft evidence.

As expected, the Bayesian network shows a significant increase in classification accuracy when compared to the texture only methods. The largest per cluster improvement in result is between the MR8 classifier and the full BN, with a 44.27% decrease in error. As the same improvement in pixel terms is only 4.4%, the conclusion can be drawn that the main impact of the BN is in the improved classification of smaller clusters. This is corroborated by the LBP results; with a 39.41% increase in cluster classification corresponding to a 5.57% increase in pixel performance. Although slight, the difference between the results suggests that fewer clusters contributed to the BN improvement over the LBP, supporting the concept that the highly localised LBP feature, should be better at identifying smaller clusters when compared to MR8.

8.4.3 Comparison of Classifiers - Total Errors per Class

Aside from the influence of cluster size, it is difficult to determine much about the relative accuracy of the classifiers from the overall error rate alone. As such, tables 8.3, 8.4 and 8.5 show the pixel level classification results for each class, for the MR8, LBP and BN classification results, respectively. Within these tables, each row represents the ground truth classification of each pixel within the twenty image test set. The columns then define the response of each classifier, with the highlighted cells representing the percentage of correct classification.

Beginning with asphalt, all three classification methods produce extremely strong results, with LBP having the highest error of 4.5%. Although painted surface markings are more visibly similar, grass was the most common misclassification for all three classification methods. As grass and asphalt as visibly dissimilar, almost all error in this class can

		MR8 Texture Classification Result						
		Asphalt	Grass	Sky	White Line	Yellow Line	Red Line	Building
Manual Classification	Asphalt	98.971	0.487	0.000	0.169	0.330	0.010	0.033
	Grass	10.068	87.900	0.000	0.971	0.870	0.000	0.191
	Sky	0.000	0.000	98.924	1.076	0.000	0.000	0.000
	White Line	3.546	0.032	0.000	77.260	19.162	0.000	0.000
	Yellow Line	2.540	0.000	0.000	2.115	95.345	0.000	0.000
	Red Line	86.534	0.000	0.000	0.838	0.873	9.388	2.267
	Building	2.177	0.000	0.000	0.000	0.175	0.000	97.648

Table 8.3 Class-based comparison between the MR8 texture classification results and the manually created ground truth.

		LBP Texture Classification Result						
		Asphalt	Grass	Sky	White Line	Yellow Line	Red Line	Building
Manual Classification	Asphalt	95.493	2.656	1.301	0.420	0.064	0.003	0.063
	Grass	3.459	94.473	0.000	1.784	0.250	0.000	0.033
	Sky	0.414	0.566	98.897	0.013	0.010	0.000	0.099
	White Line	4.218	1.649	0.000	86.482	7.632	0.000	0.019
	Yellow Line	5.353	1.694	0.000	16.955	75.998	0.000	0.000
	Red Line	86.469	11.261	0.000	2.004	0.104	0.085	0.077
	Building	2.157	0.082	0.000	0.245	0.397	0.000	97.119

Table 8.4 Class-based comparison between the LBP texture classification results and the manually created ground truth.

be assumed to be the result of segmentation errors along taxiway borders. As such, the BN has only slightly improved upon the results of the individual texture classifiers, as it also depends directly on the segmentation results.

As expected, a near identical situation exists for grass, where asphalt is the class most likely for grass-clusters to be misclassified as. Of some note, unlike asphalt, there has been around a 10% improvement between the MR8 classifier and the final BN. This indicates that the texture of grass is better detected by the LBP, and that fusing texture with colour provides a very good final result.

For all three classification methods, the sky class is by far the most well classified; having results of 98.9% correct for both MR8 and LBP, and increasing to 99.6% for the final BN. This highly accurate result is due to the sky representing a huge proportion of each image, with segmentation between ground and sky usually highly accurate. The slight errors which remain are most likely due to discrepancies between the borders in the automatic segmentation compared to the manual ground truth.

Considering all three types of surface marking together, tables 8.3 and 8.4 re-affirm that texture data alone is insufficient for classification, with poor results for white and yellow surface markings, and extremely poor results for red surface markings. For both MR8 and LBP, the white and yellow surface markings are more likely to be misclassified as each other than as asphalt, with 19% of white lines being misclassified as yellow lines by MR8, and 16% of yellow lines being misclassified as white line by LBP. This is to be expected, as the texture of paint is sufficiently distinct to differentiate it from asphalt, but is not distinct enough to determine the colour. By contrast, due to the poor state of red surface markings, they are almost entirely misclassified as asphalt, with only 9% of MR8 classifications being accurate, compared to far less than 1% for LBP. As the paint has worn away, the underlying asphalt becomes more apparent, eliminating any distinguish

		Full Bayesian Network Classification Result							
		Asphalt	Grass	Sky	White Line	Yellow Line	Red Line	Building	Unknown
Manual Classification	Asphalt	98.976	0.543	0.000	0.044	0.012	0.010	0.000	0.41
	Grass	1.600	98.327	0.000	0.000	0.014	0.000	0.000	0.06
	Sky	0.042	0.000	99.554	0.000	0.000	0.000	0.000	0.04
	White Line	0.897	0.000	0.005	95.317	0.767	0.228	0.000	2.78
	Yellow Line	0.063	1.259	0.000	2.051	93.647	0.000	0.000	2.98
	Red Line	2.489	0.100	0.000	0.013	0.000	97.384	0.000	0.014
	Building	3.952	0.000	0.000	0.000	0.000	0.000	96.048	0.000

Table 8.5 Class-based comparison between the full BN classification results and the manually created ground truth.

features. As asphalt is the more common class, it's feature space is better defined and therefore more likely to be selected when working with texture alone.

With poor surface-marking texture-classification results prompting the inclusion of other types of data, the improvements shown in Table 8.5 are as expected. By including NRL and HSV within the BN, the final results are significantly improved, increasingly classification of white, yellow and red surface markings to 95.32%, 93.65% and 97.38% respectively. Although red surface markings are still misclassified as asphalt for 2.49%, the quality of the surface markings within the test set makes further improvements unlikely.

For yellow and white surface markings, high NRL values helps to isolate the classes. However, differentiating between the two colours depends on primarily on Saturation of HSV data, resulting in the the two classes still sharing slight misclassification between each other. However, rather than misclassification, the most common results for white and yellow surface markings, (aside from correct classification) within the BN is 'unknown'. This is due to clusters having low confidence in the texture result, with the added information from colour and NRL insufficient to bring the confidence back over the threshold. As these are likely to be small clusters which do not adhere to the actual surface contents, they should only appear in individual frames and not affect the taxiing overall.

Finally, the building class has received a remarkably high classification result for all three classification methods. Although buildings should be easily separable from other classes on texture alone, the overall range of potential material should introduce a greater degree of error, with more clusters being identified as unknowns. As all data has been acquired from a single aerodrome, it is likely that 'over-fitting' has occurred; with the small number of buildings sampled creating a texture feature space which would not be representative of conditions at other aerodromes. Despite this, as stated in Chapter 7, the building class was primarily included within the BN to assess the ability of the classifier to deal with a highly varied class. As buildings are static collision risks and can therefore be avoided using position data alone, the overall impact of the over fitting is minimal.

8.5 Conclusions

This chapter has applied the classification framework to a dataset acquired from a real-world aerodrome, with images captured from positions representative of a taxiing aircraft. Although the test environment was small, sufficient examples of each class were available for training. Selecting one hundred images from the entire dataset, manual ground truth images were created, with eighty images used for training the classifiers, whilst the remaining twenty were used to create the results above.

The BN framework has been shown to near universally improve classification performance when compared to the individual texture classifiers. By including the confidence of the texture classification result, as opposed to just the winning class, the overall classification performance improved by around 5%. Unlike similar methods of achieving this result (such as ANN), the BN process is deterministic. As such, the flow of information through the BN can be easily monitored and verified; as required for safety critical aircraft systems. As the BN allows for further extension, additional sources of information could be included going forward, such as cluster adjacency. (However, as cluster adjacency requires a multi-pass approach, it was not included here as a significant modification to the BN would be required).

The greatest limitation of this study is the test set; consisting of a single aerodrome and with images captured during the course of a single day. Although some variance in weather and lighting was incorporated, it was insufficient to have a major consequence in the test results. To produce a more universal system, data would ideally be sourced from multiple aerodromes, in varying conditions and seasons. However, due to restrictions of time and cost involved, this has not been possible.

As the training and test data are sourced from the same dataset, over-fitting is unavoidable. However, the basic principles of the BN approach have been validated, a larger data training set should be all that is required. Therefore, although the classifications results may have been over-fitted to the conditions, the output is considered sufficiently accurate for further interpretation.

Chapter 9

Depth Exaction and Collision Risk Localisation

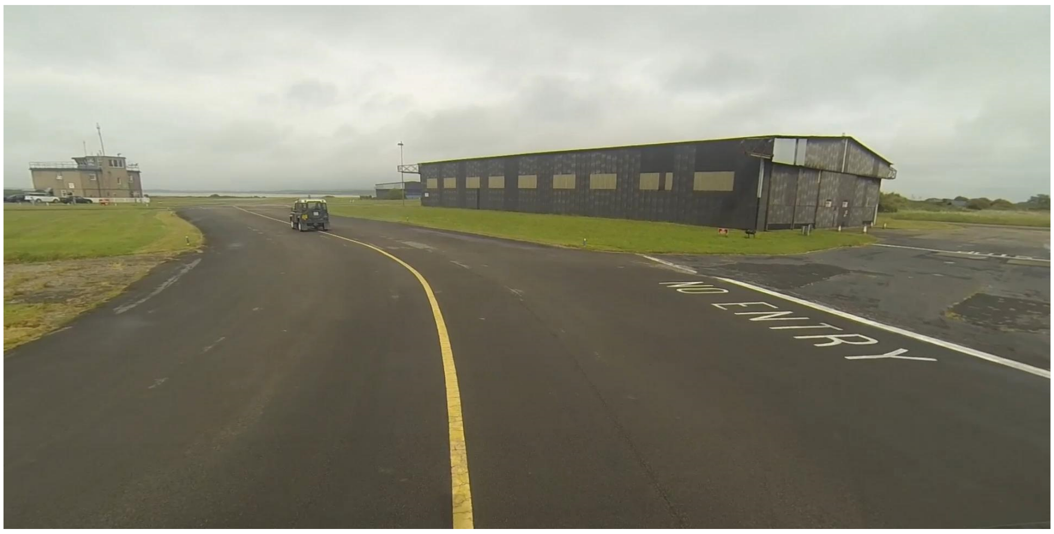
At the conclusion of the classification stage, each pixel within the captured image has been assigned a state; either as a known class useful for localisation, or as an unknown class useful for collision avoidance. Despite the great deal of processing required to reach this stage, the data within the image is still not directly usable. Although the presence of collision risks within the image can be determined, without range or bearing data the ATS has knowledge of where collision risks are actually located. Similarly, although terrain features have been identified, they have no context to aid in localisation. As such, this chapter focuses on converting the image data into more useful information.

9.1 Class Amalgamation

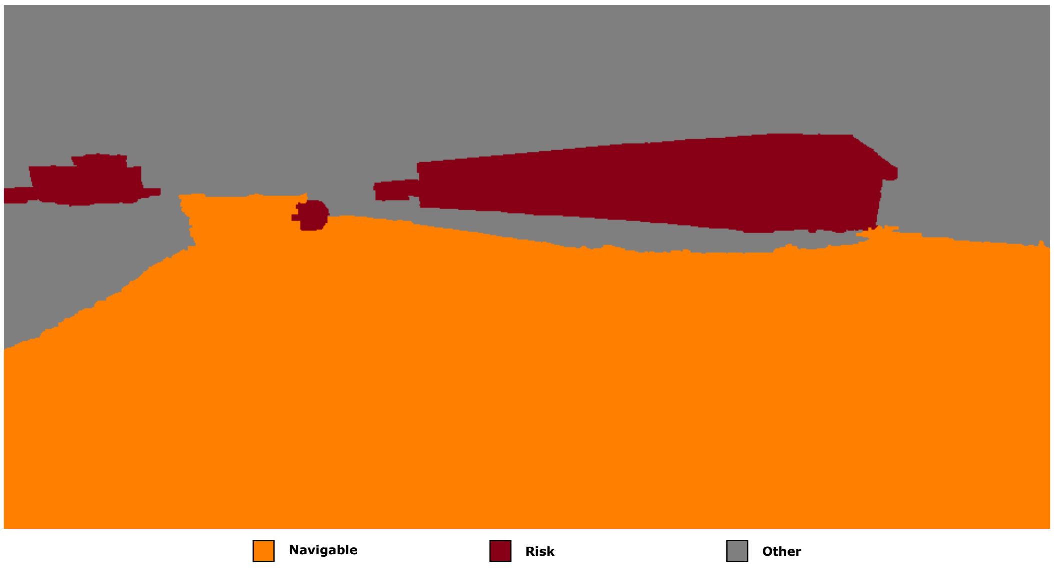
Prior to determining the real-world position of each cluster, the data obtained from classification can be simplified to make processing more straightforward. As many of extracted classes are no-longer relevant by themselves, it is possible to combine the classification output into new 'amalgamated' classes. Specifically for collision risk detection, only 2 classes are required:

- **Risk Class** - 'Unknown' and 'Building' classes are combined into a class which tracks all clusters which could present a collision risk.
- **Navigable class** - 'Asphalt' and all three 'Surface Marking' classes are combined into a single class which tracks the extents of the navigable terrain.

Although not directly part of collision risk detection, the 'Navigable' class is useful as it represents the area in which the area can manoeuvre. As the limits of navigable space may be defined by non-risk clusters (such as grass) the navigable surface is important when attempting to confirm a route is valid (leaving the navigable area should not occur in standard operation conditions). Finally, the remaining classes, ('Sky' and 'Grass') are simply not considered within the collision risk process. An example of the amalgamated classes is shown in Figure 9.1b.



(a) Original image with collision risks and taxiway



(b) Amalgamated classes for collision risk detection.

Fig. 9.1 Comparison of original aerodrome image with amalgamated classes.

9.2 Temporal Smoothing

Following on from class amalgamation, all potential collision risks are stored within a single class. However, this final class is only based the instantaneous classification of all objects present in the input image. As no data is exchanged between frames, both segmentation and classification occur on a per-frame basis. Therefore, exact borders of objects will vary over time, even for static objects when the camera is not in motion.

The significance of this variance depends on the size of each cluster. Large clusters,(such as taxiway surfaces), remain consistent as small changes in border position have little effect on the cluster overall. However, smaller clusters are far more likely to vary over time. As shown in column (b) of Figure 9.2, the shape of each cluster depends on the underlying superpixels.

If an unknown cluster represents a moving collision risk, the changing position of the cluster combined with varying borders can make the boundaries of the object inconsistent, even within just a few image frames. This makes collision risk detection far more difficult. If range and bearing estimates were extracted from individual frames, the position of risks will rapidly change. Therefore, to limit the influence of inter-frame variance, Multiple-Frame Averaging (MFA) has been introduced. This form of 'temporal smoothing' is intended to minimise the influence of single frame misclassification, whilst better defining the boundaries of collision risks.

9.2.1 Multi-Frame Averaging

Methods which combine data from multiple sequential image frames to produce an improved output are commonly referred to as MFA. This form of filtering is often used in machine vision for noise reduction, where repeatedly sampling a scene can help to remove instantaneous artefacts in any one frame. This is very commonly used to remove rain from video footage for films and television [60]. Multi-frame sampling can also be used for background subtraction [34]. Rather than attempting to remove anomalous data, background subtraction works by averaging multiple frames to determine the static components of a scene, allowing moving objects to be segmented on even the most cluttered of backgrounds. (As what is not the background must be the foreground, this can also be referred to as foreground detection).

For this work, MFA is applied differently as there is no need to perform background subtraction; all regions within each image which are not considered to be a collision risk are already a known value within the Navigable image. Instead, it is used only to increase certainty that a collision risk exists through repeated observation. Rather than isolate the background of an entire image, it seeks to isolate the 'core' regions of collision risk clusters.

The temporal smoothing process begins by creating a separate 'risk' image for each frame. This consists of a binary image in which every pixel in each risk cluster is labelled as a 1. Morphological dilation is then applied, with the intention of connecting disconnected elements of the same objects (as shown in column (c) of Figure 9.2). Although this slightly expands the borders of detected risks, this can only make the risk larger and therefore should never decrease the safety of the UAS. Temporal smoothing can then be applied by comparing the current frame to previous observations. Various methods of MFA exist:

- Direct Frame Difference - Comparing only neighbouring sequential frames.
- Running Average - Average the value of each pixel over several previous frames.
- Running Median - Median value of each pixel over several previous frames.
- Running Gaussian Average - One dimensional Gaussian smoothing over time.

As the purpose of temporal smoothing is to remove instantaneous errors, directly comparing neighbouring frames is insufficient, making Direct Frame Difference unsuitable. (Therefore, in addition to selecting a method, the number of frames over which to sample must also be decided and will be further discussed in Section 9.2.2). Furthermore, as the input image is binary, a median filter can only ever produce a binary result. As this is equivalent to an average filter with a fixed threshold, a running median filter was not used.

Gaussian averaging is well established as producing a good result in colour images [72]. However, as with all MFA methods, Gaussian smoothing is typically intended to be applied in post-process, with results about the current frame drawn from both future and previous frames. With only previous results to draw from, the Gaussian approach would be half a bell curve if applied to only the latest frame. This would bias the data to suggest that the most current footage is also the most accurate. Whilst this would be the best approach for removing noise from colour footage, this use case is highly difference. For collision risk detection, a risk spotted in an earlier frame may be missed in the most current and each frame within the running set should be given equal weighting. Therefore, a running average is considered to be appropriate for this scenario.

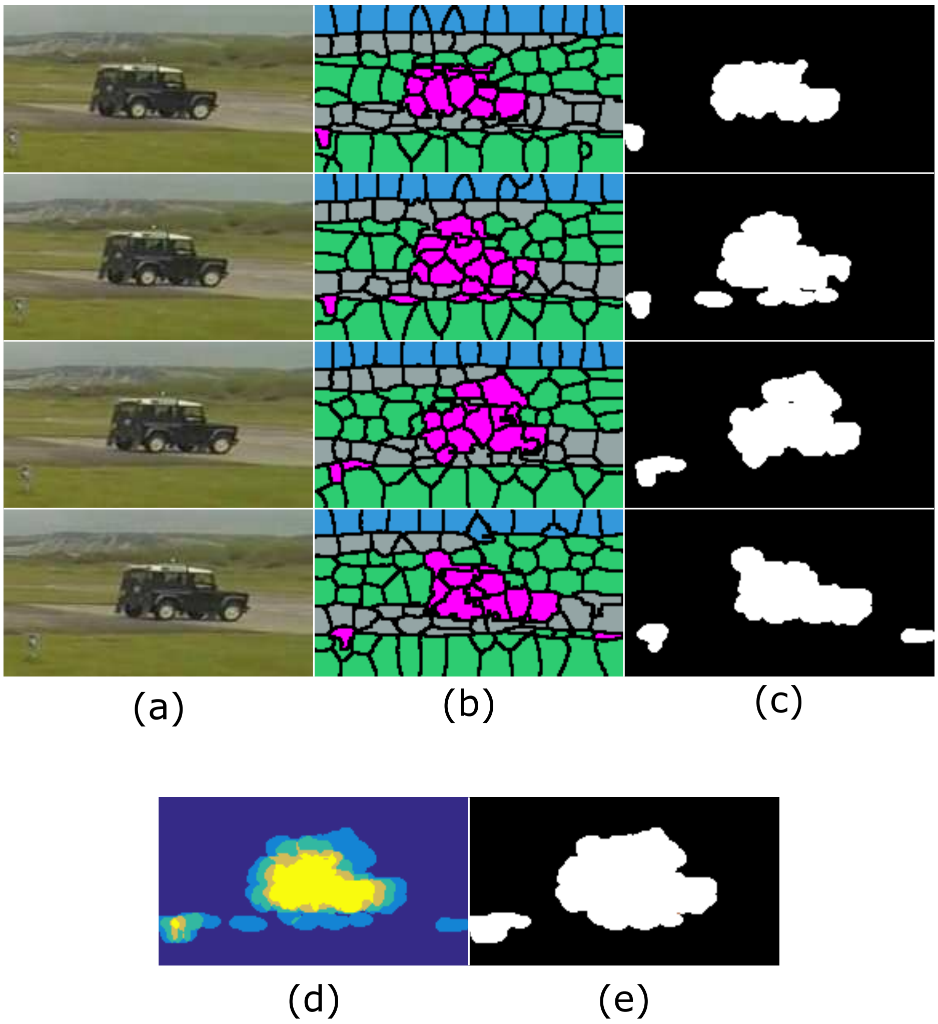


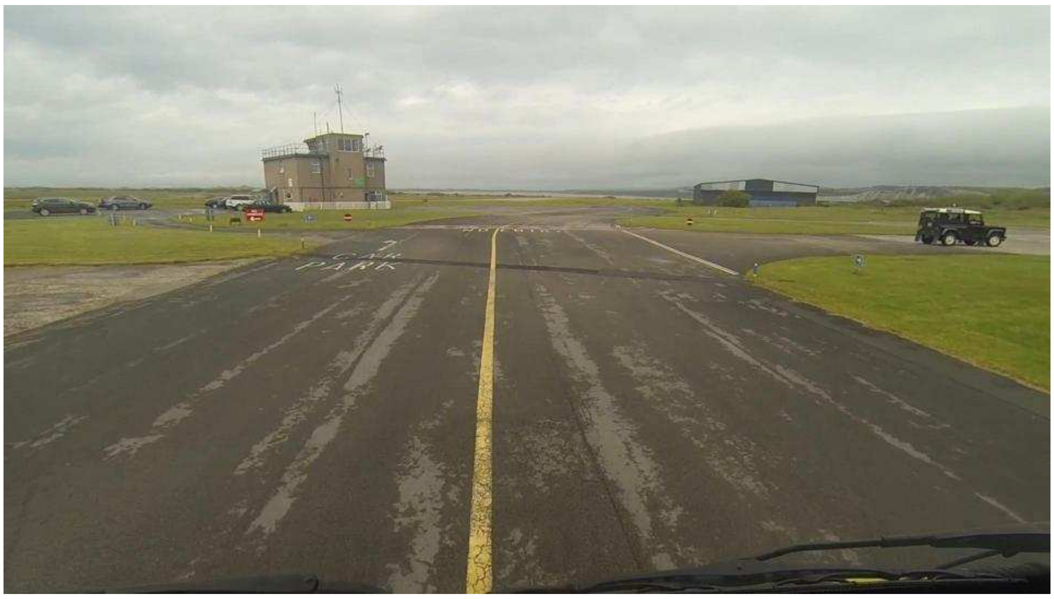
Fig. 9.2 Stages of temporal smoothing, showing: (a) the original image, (b) the classified image with superpixel borders overlayed, (c) the 'risk' layer after class amalgamation, (d) the 'rolling risk' layer and (e) the temporally-smoothed collision risk cluster.

- To incorporate data from multiple frames, a 'running risk image' is created, in which the previous N-risk images are summed together. (To ensure the information is entirely relevant, all clusters older than N frames are discarded.) As the risk layer is a binary image, the value of each pixel number within the rolling risk layer is a direct count of how many frames that pixel has been identified as collision risk (as shown in image (d) of Figure 9.2).
- By determining a suitable threshold, only clusters which have appeared enough times within the last N-frames are considered actual risks; other clusters are considered to be noise.
- Using the newly extracted 'cores', connected components are again extracted, using a binary version of the entire rolling risk image in which all non-zero pixels are considered. This ensures that even if only a part of the object has been classified in enough frames, the entire suspected cluster is extracted (as shown in image (e) of Figure 9.2).

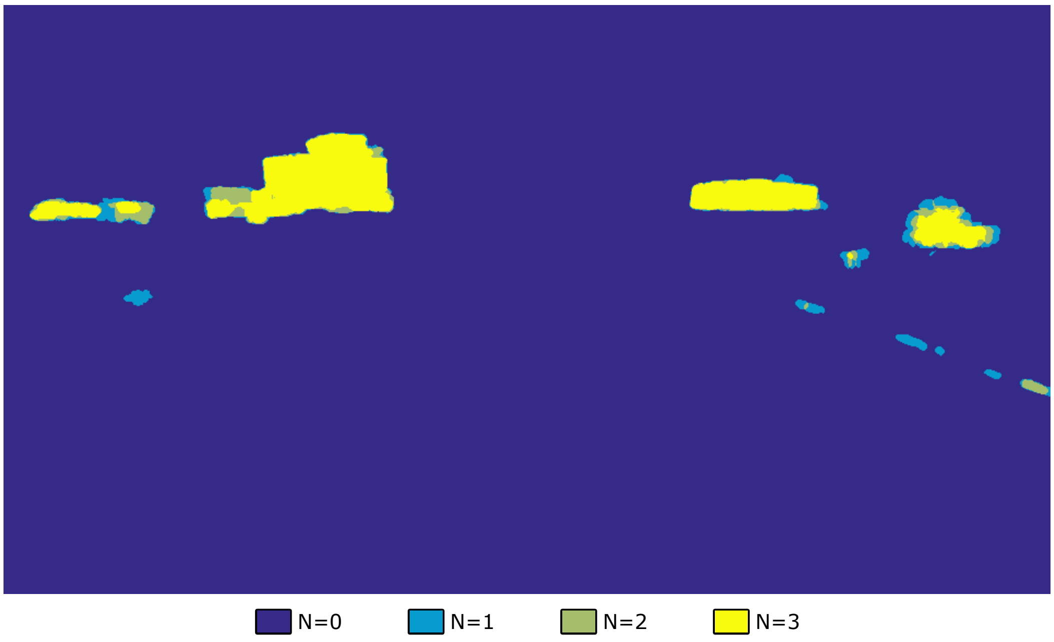
Figure 9.3 shows the results applied to a full image. As both the camera and the other ground vehicle are in motion, there is significant variance in the position of the cluster from frame to frame. Despite this, a clear 'core' can be observed, allowing the position of the risk to be determined more consistently.

9.2.2 Risk Frame-Buffer Size

Previously referred to as N, the number of frames used within the running risk image (i.e. the image buffer size) has a large effect on the final result. If too many frames are used, rapidly moving objects are unlikely to reach the threshold so will not be considered risks. Conversely, if too few frames are used, single frame misclassification noise may introduce false 'risks' into the scene. Therefore, selecting the appropriate number is critical.



(a) Original Image when approaching collision risks



(b) Temporally smoothed collision risks, where N indicates the number of frames for which a risk has been observed.

Fig. 9.3 Comparison of original aerodrome image with temporally extracted collision risk image. The added motion of the vehicle results in a less clearly defined risk.

Determining an acceptable number of frames is simplified by the fact that the camera can move. As the UAS is in motion, each frame will have a maximum expected change, restricting the amount of frames which can be used. Although the maximum forward velocity of the UAS could be used to define the expected change, the borders of objects change with respect to their distance from the camera. As such, until the objects are close, the rate of change will be low. Instead, using the rotational velocity of the aircraft is a better indication of how many frames an object can be expected to remain in a similar position within the image.

Observing aircraft taxiing around an airport, it was concluded that the forward speed of an aircraft affects the radius of the turn, but not the rate of turn itself. By watching a taxiway corner at East Midlands Airport in the UK, it was observed that smaller general-aviation aircraft were able to make a 90° turn in approximately five seconds. However, the size and role of the aircraft can affect this, with large airliners taking between 20 to 30 seconds to make the same turn. Despite not being large, UAS are remotely guided and thus are unlikely to be making the rapid taxiing manoeuvres common to general aviation pilots. Therefore, 15 seconds for a 90° turn was considered appropriate.

This results in a turn rate of 6 degrees per second. The camera captures images at 30 frames per second, and has a 94.4 degree horizontal field of view. With 960 pixels in the horizontal image dimension, during rotation static objects within the image should differ at most by 8 pixels per frame. As a collision risk smaller than 40 pixels in width is unlikely to be an imminent threat to the UAS, a four frame buffer was chosen to be the maximum allowable. As such, the temporal smoothing uses four frames to create the 'rolling risk' image, with a minimum of three appearances required for a cluster to be declared a risk.

9.3 Depth Extraction

After temporal smoothing has been applied, the clusters should be consistent enough for depth extraction to be performed. The term 'depth extraction' relates to the fact that when a camera captures an image, the visible scene undergoes a transformation from three dimensions down to two. Depth extraction then seeks to restore the third dimension to the image, by mapping each pixel to a point in 3D space.

9.3.1 Methods of monocular depth extraction

In Chapter 7, a simplistic method of distance estimation (depth extraction) was introduced to help combat the influence of atmospheric effects on the detection of surface markings. As NRL itself is broadly discretised, a precise measure of distance was not required. Therefore, the pin-hole camera model was used. This provided a simple estimate which was then further grouped into either near, mid or far categories. Although these results proved to be sufficient for roughly grouping clusters, the actual estimate of distance was quite poor. As such, a more reliable method of determining the distance to 'risk' clusters is required. Although cameras lack any inherent depth data, they are commonly used to track objects in 3D space. There are two primary methods for achieving this: stereo-vision and in-image context.

Stereo-vision is the concept of combing multiple image from known positions to create a depth map, without the need for classification. As previous covered in Section 4.3.1, this can either be achieved using multiple cameras at fixed distance from each other, or using a single moving camera with a known path. As the UAS hardware has been defined as a single forward facing camera, multi-camera stereo is not possible. In addition, as the act of moving the UAS to detect collision risks is inherently dangerous, this is also not a viable option. For these reasons, and for the reasons described in more detail in Section 4.3.1, a stereo based system is not viable.

In-image context is a far broader set of techniques used to elicit information from a scene. Monocular depth extraction relies on being able to extract 'depth cues' from the image; features from which scale and distance can be estimated. As lines and corner points alone provide little context, this usually requires classification of entire objects which can be isolated from their surroundings. For example, processing an image containing a car on a road would begin by separating the car from the background. If the physical dimensions of the vehicle are known, the size of the vehicle within the image can be used to infer distance. As with stereo vision, the concept of known object detection was already covered in Section 4.2. Due to numerous difficulties in isolating vehicles and the immense data set required to detect all possible objects, the generic method of object detection through terrain classification was chosen instead. As all risk clusters are detected using a generic method, the dimensions of each object are wholly unknown and therefore depth estimation through object size cannot be used.

Distance estimation through terrain classification is a fairly new method of eliciting depth information. Just as the effects of atmospheric scattering are proportional to the objects distance from the camera (as discussed in Chapter 7) similar contextual clues can be extracted from clusters to provide an estimate of depth. Techniques such as [93] have demonstrated the ability to extract depth maps from monocular images through comparison to known data sets, essentially classifying the image into distance clusters based on known data, with neighbouring superpixels used to smooth the final result. As the data used here is already broken into superpixels, a similar application could be viable. However, although this contextual approach was shown to perform well at producing a 'relative' depth map, it was less successful at producing an absolute depth map. As collision risks at an aerodrome are typically the only object within view, a relative distance map is difficult to incorporate. In addition, as this method is highly dependant on texture, different weather conditions can render the method unusable.

As such, the contact point between the object and the ground is again determined to be the most reliable source of contextual information, linking the camera to the collision risk. Assuming that the ground is flat and that the risk clusters within the image lie on the ground, the intersection of objects within the 2D image space can provide context to where they lie in 3D. This is essentially a plane-to-plane mapping, from the image plane to the ground plane, using the perspective of the camera to determine the results. As such, the methods which use this approach are commonly referred to as Inverse Perspective Mapping (IPM). Despite using principles similar to the pinhole camera model, IPM represents a far more accurate mapping and should always provide a more accurate result.

9.3.2 Inverse Perspective Mapping

When an image is taken, each object within the camera's field of view is *mapped* onto the image plane as a cluster of representative pixels. In converting between 3D and 2D, the exact location of the representative pixels is determined by the effects of perspective. As the name suggests, an *Inverse* Perspective Mapping is intended to 'reverse' this process, returning pixels to their 3D coordinates from when the image was taken.

As objects have the potential to be anywhere in the scene, restoring all pixel data is impossible. Instead, an IPM can only be established for pixels where the original object had a known relationship with the camera. Despite this limitation, IPM has many applications; with by far the most common being for Self Driving Cars (SDCs), where removing the effects perspective can assist in lane marking detection. As such, there is already great precedent for the use of IPM from a forward facing monocular camera, mounted on a moving vehicle. As IPM is simply a concept, there are numerous ways in which it can be implemented. For this work a combination of solutions has been applied.

The concept of IPM for road vehicles was first introduced by Bertozzi et al. (1998) [15] and the equations from this paper are still in general use. The IPM process begins by defining two euclidean spaces: $W = (x, y, z)$, the 3D world space and $I = (u, v)$ the 2D image space. Although information can be mapped in either direction, only the $I \rightarrow W$ mapping is of interest. The general concept of IPM is shown in Figure 9.4.

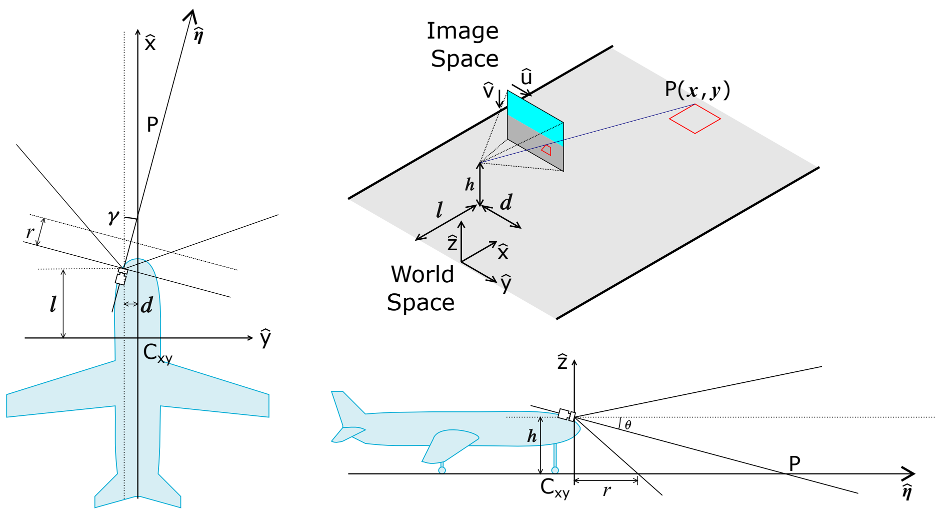


Fig. 9.4 Inverse Perspective Mapping overview, showing the physical distances and angles required to map a point in image space onto the ground plane in world space.

In order to calculate an IPM the following parameters are required:

- Half the angular aperture of the camera $alpha$.
- The resolution of the image $(m \times n)$.
- The position of camera in world co-ordinates (l, d, h).
  - Where h is the height of the camera above the ground plane and l and d can be set to zero when working in camera local co-ordinates.
- Attitude of the camera: pitch angle θ and yaw angle γ.
- The image co-ordinates of the point being mapped (u, v).
  - In contrast to the more typical one-based co-ordinate system, zero-based co-ordindates are used here to simplify the equations.

As all of these vairables are known, the (x, y, z) position of Point P can then be determined using Eq. (9.1):

$$\begin{cases} x(u,v) = \frac{h}{tan\left[(\theta-\alpha)+u\frac{2\alpha}{n-1}\right]} \times cos\left[(\gamma-\alpha)+v\frac{2\alpha}{m-1}\right]+l \\ y(u,v) = \frac{h}{tan\left[(\theta-\alpha)+u\frac{2\alpha}{n-1}\right]} \times sin\left[(\gamma-\alpha)+v\frac{2\alpha}{n-1}\right]+d \\ z=0 \end{cases} \tag{9.1}$$

Originally, the intention was to directly replicate the IPM method used by Bertozzi et al. without modification. However, one drawback of this early method was the assumption that the yaw and pitch angles of the vehicle were fixed, with the only offset due to the mounting angle between the camera and vehicle. As such, the estimated distance would vary during acceleration and cornering. Later work by Kheyrollahi et al. (2012) [58] endeavoured to solve this using visually determined features to estimate yaw and pitch. In Kheyrollahi et al. (2012), the World space coordinate system was relative to the road centreline, with the two sides of the road used to detect a vanishing point at the horizon. The position of the vanishing point was then used to calculate pitch and yaw.

As calculations for this work are performed relative to the camera, the yaw angle does not require calculation. However, an accurate estimate of pitch is useful to minimise the effects of braking and cornering, in addition to the general flexibility of aircraft undercarriage. Rather than perform an addition estimate of vanishing points, the visually derived horizon position (from Chapter 6) is used instead.

As with temporal smoothing of the risk clusters, the horizon line position is also smoothed. As the horizon line is stored using the straight line equation, the temporally smoothed horizon line can be calculated by averaging the y-intercept and gradient over the last 4 frames. The y-co-ordinate of the vanishing point y_{vp} is then assumed to occur at the intersection of the horizon line and the centre of the image, as shown in Figure 6.33. The pitch of the vehicle relative to the horizon can then be estimated using using Eq. (9.2):

$$\theta = \arctan(1 - (\frac{2y_{vp}}{n})) \tag{9.2}$$

Although this estimate is only valid for small angles, the horizon line detection algorithm within this work is already limited. Should the visually derived horizon differ significantly from the inertial sensor readings, the horizon is assumed to be obscured and an inertially derived horizon line is used instead. For use here, when the horizon is data is no longer available, the inertial pitch angle is used directly.

9.3.3 Detection range and error

An unavoidable limitation of all visual distance estimation methods is range. As objects become more distant from the camera, their associated clusters decrease in area and move vertically in the image plane towards the horizon. As every object within an image is made of pixels, the absolute range is determined by the ability of the classifier to discern the object. Just as fewer pixels are available to define the object, there are also fewer pixels available to define the difference in object position. As such, the functional range is the distance at which results can be considered accurate; which is dictated by the resolution of the image.

Due to the oblique angle and low height of the camera, significant differences in distance can be represented by just a few pixels. As the camera is attached to a moving vehicle, the accuracy of depth estimation is further reduced based on the stability of the camera. Despite the use of the horizon line to attempt to mitigate the effects of vibration, even small differences in camera attitude can have a significant impact on the distance estimate. Although collision risks objects can be detected at a long range using the classification approach, without confidence in the distance estimate the position data is not useful for either localisation or collision avoidance.

The use of IPM for SDC is similarly range limited. Although the majority of publications choose not to specify their maximum range, 28 metres was chosen as a functional limit by Gang et al. (2000) [54]. More recent work has selected a similar limit, with a system specifically designed to monitor road traffic, classifying objects at 30m as 'distant' [108]. As aerodromes are far larger in scale than roads, relevant collision objects are often further than 30 metres ahead, requiring a longer detection range. Therefore, a maximum viable 'depth' range has been defined.

As the accuracy of the IPM is discrete, the distance relative error can be calculated by determining the equivalent distance estimate (and therefore the error) if the cluster was offset by a small number of pixels. Figure 9.5 shows the estimated maximum IPM error over distance, for two different pixel offsets.

To determine these errors, a representative image was created with a flat and centralised horizon line. Using the same camera parameters as the physical camera, for each pixel below the horizon, the associated world-space distance was calculated. As cluster border precision is likely to be the largest source of inaccuracy, the relative difference between results was found if the cluster border was shifted by either 1 or 5 pixels. (This is considered the 'maximum' error as the pixel offset was always directed towards the horizon line. However, larger pixel offsets will obviously produce larger errors).

From Figure 9.5, the expected trend can be seen; with a consistent cluster border shift

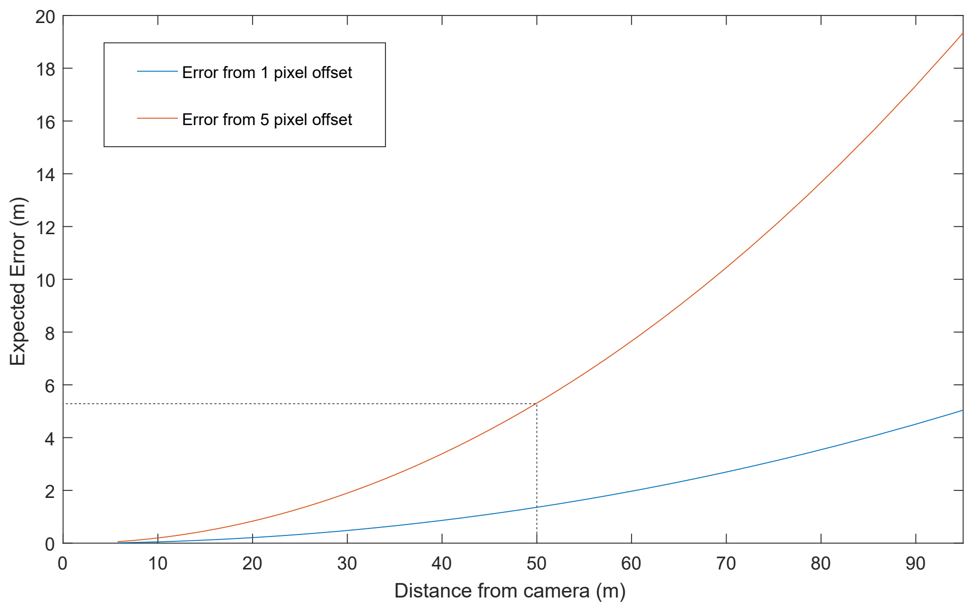


Fig. 9.5 Distance Estimation Error due to Inaccurate Cluster borders - calculated using the IPM formula and assuming cluster borders are 'shifted' upwards to create the maximum error.

resulting in larger distance estimate errors, as the camera moves away from the object. As the error is not linear, smaller distances see almost no error with a single pixel variance, where as the same border offset can cause a 20 metre position shift when the object is 90 metres away. In order to select an appropriate maximum detection range, two separate issues must be balanced:

- The accuracy of objects further from the camera cannot be relied upon, with collision risks potentially moving rapidly between frames.
- Collision risks must be detected with sufficient distance to allow the UAS to stop.

At the most basic, collision risk detection is used for detecting risks which will cause the UAS to stop. Therefore, the maximum detection range should allow both detection and braking to occur. As aircraft taxiing speeds are not regulated, the exact speed at which the UAS will travel is unknown. Instead, taxiing speed is typically limited by the size of the aircraft; as smaller aircraft often gain sufficient lift at lower speeds to lose control, they must taxi slower than larger aircraft. For most small aircraft, 20kts ($\sim 10ms^{-1}$) is the maximum taxiing speed. Although braking distance is vehicle specific, as the equivalent braking distance for road vehicles is 14 metres (at $10ms^{-1}$), a 50 metre distance is considered an acceptable maximum range for both detection and braking.

As highlighted on the graph, 50 metres also represents the approximate distance at which a 5 pixel shift will result in only 10% error. Although still large, a 5 m position error can be likely be resolved by filtering over time.

9.3.4 Nearest point approximation

In order for an IPM to provide an estimate of distance, the assumption is made that all collision risks are in contact with the ground. However, as the actual identity of each collision risk is unknown, the exact points of contact are not defined. To avoid having multiple distance estimates for a single object, IPM are commonly applied to bounding boxes, rather than the outline of the object itself. This is especially common in SDC research (both [120] and [102] use bounding boxes to represent other vehicles, despite very different overall approaches). Using the lower edge of a bounding box as the contact point with the ground ensures that the minimum distance to the camera is associated with the entire object, rather than just the closest point.

Despite this benefit, the applicability of a bounding box approach for the aerodrome environment is debatable. Although the tracking of vehicles would benefit from a single

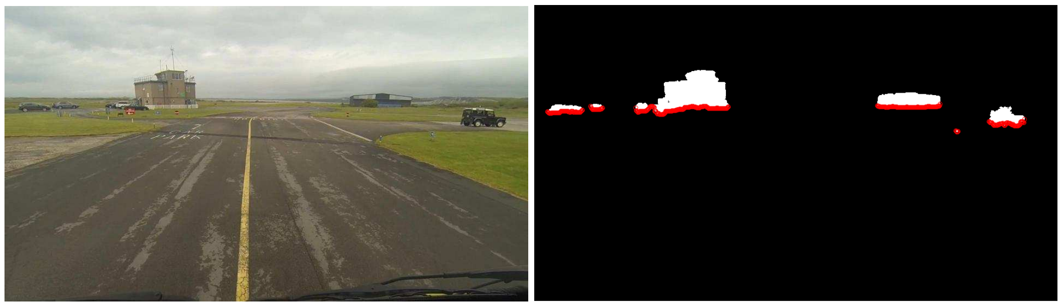

(a) Image with collision risks (b) Closest Point on Clusters

Fig. 9.6 Example of Nearest point approximation for distance estimation

point approximation of distance, bounding boxes are only appropriate where an object will present a consistent face to the camera (such as the back of a preceding road vehicle).

Attempting to fit a bounding box representations to larger and less uniform objects can dramatically alter the position estimate, especially at range. Just as it is important to define the boundaries of an object for collision avoidance, it is also important to avoid introducing false detections which would impede progress. For example, if a long building runs along the side of the taxiway, the bounding box will likely infer onto the asphalt, due to the effects of perspective.

Instead, as the risks are already segmented into clusters, the closest point of risk to the camera is most easily determined by finding the lowest ‘risk’ pixels within each column of the risk layer. Although the lowest pixel is not always the closest point, by selecting the entire lower edge of each cluster, all contacts points with the ground must be sampled, with the closest point found from amongst them.

For objects which are partially suspended (such as aircraft and road vehicles mounted on wheels) only a small amount of the bottom edge actually represents points at which the object meets the ground, with most of the vehicle detected further away. As such, despite not using a bounding box, the closet point on each cluster will need to be associated with the cluster as a whole.

9.3.5 Homogeneous Perspective Transformation

In addition to being able to apply an IPM to individual points within the image plane, IPM is also commonly used to transform the entire image. By mapping the terrain features back onto a ground plane (to where they originally were), image processing can be performed as if the image were taken from above. Although the features have already been extracted during the classification process, techniques such as map matching and surface marking inspection can still benefit from whole image perspective transform.

For this work, the MVTec HALCON image processing library [79] has been used to map from the image plane into the world plane, using a homogeneous perspective transformation. This process is achieved by establishing a quadrilateral within the image plane and then determining the associated quadrilateral in the world plane, using the IPM. Figure 9.7 shows the results of applying the IPM when the camera is still.

Returning to the collision risk image shown in Figure 9.3, the IPM was applied in order to determine the distance of the collision risks from the camera. Despite all collision risks occupying a relatively similar vertical position within the original image, the two buildings are both far outside the 50 metre accuracy range. As such, only the road vehicle (and the small signpost) were close enough for the results to be considered accurate. As GPS data was only recorded for the camera vehicle, map data has been used to obtain an approximate distance to the 'collision risk vehicle', of 36 metres. Using the IPM, the closest point on the other vehicle is 33 metres. Although this difference could simply be down to cluster border error, the shorter distance estimate is partly due to the effects of temporal smoothing, which extend the size of the cluster to include it's position in previous frames. As the most recent frame is also included in the risk cluster, this should not impact the time taken to detect a collision risk.

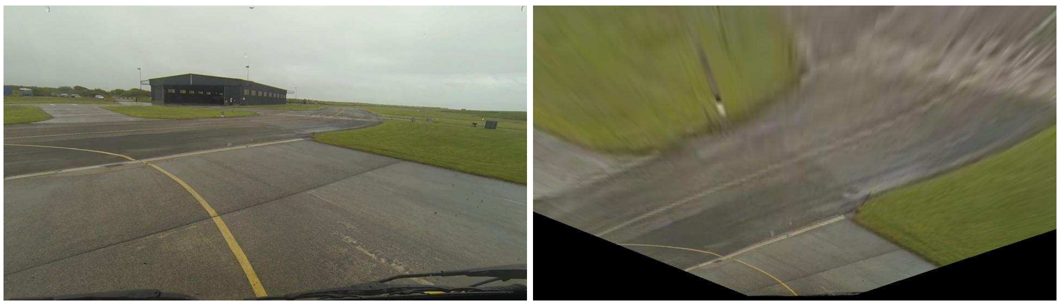
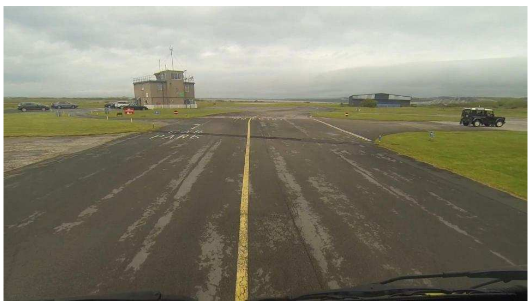
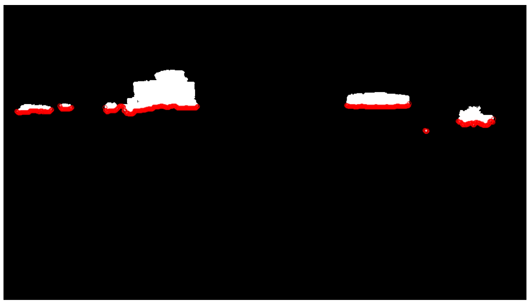

(a) Image taken on taxiway ramp (b) Inverse Perspective Mapping

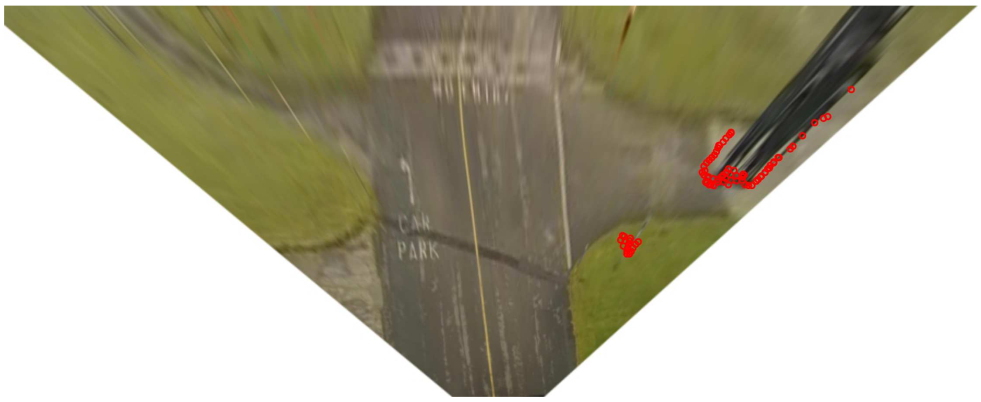
Fig. 9.7 Application of Inverse Perspective Mapping using Homogeneous Perspective Transformation



(a) Original Image



(b) Nearest Points to Camera



(c) Inverse Perspective Mapping of Nearest Points, combined with original image transformed using a Homogeneous Perspective Transformation.

Fig. 9.8 Summary of stages in determining the relative position of collision risks

9.4 Results

A brief study has been undertaken to determine the accuracy of the IPM approach using real-world data. Five sections of the data set were chosen in which the camera moved steadily towards a known static object at the aerodrome (for which both the control tower and the hanger were used). For each frame the IPM distance was calculated, using the closest point on the detected cluster. As known objects were selected, a comparative distance was found using map data and the GPS position of the camera vehicle. (Although the GPS derived data cannot be considered accurate, it should be sufficient for a general comparison).

Despite only using five short periods of footage, as the IPM distance was calculated for each frame the object was in view, several thousand data points were generated. As the five sections of footage were taken from different positions and at different speeds (to ensure that a broad spectrum of the working range was captured), the error between IPM and GPS data was calculated and the results grouped corresponding to the nearest (GPS) metre from the object. Two metrics have been selected for comparison: the maximum absolute error and the average absolute error. Figure 9.9 shows the combined results.

From the graph it can be observed that the results follow the expected pattern, with greater separation between the camera and the object resulting in noticeably decreased accuracy. Despite the most extreme error being an overestimation of distance, under-

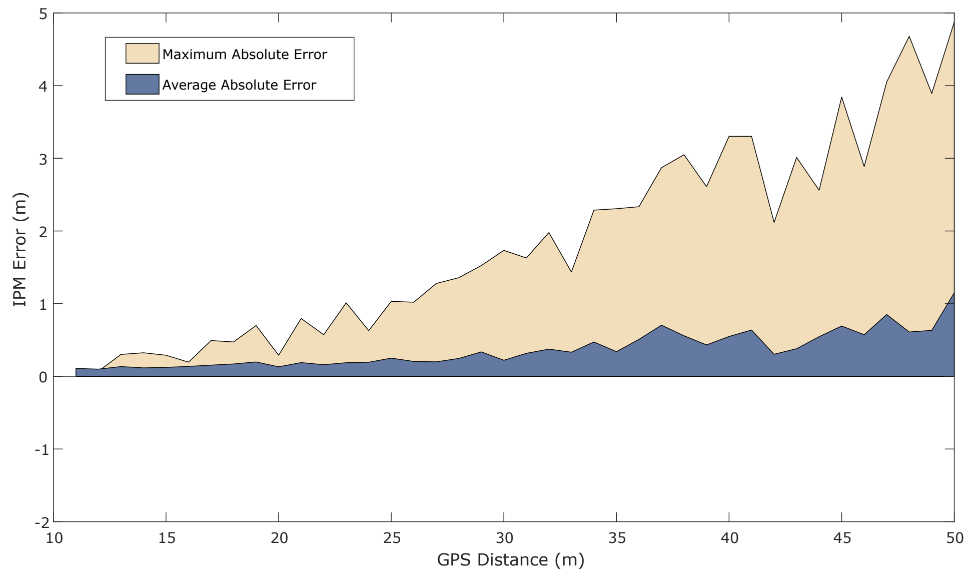


Fig. 9.9 Maximum and Average IPM Error found when processing dataset.

estimated distance error occurred more frequently. As this could very well be caused by an unintended offset between the GPS position of the vehicle and the map-derived location of the building, absolute error is displayed.

The following conclusions can be drawn from this result:

- As the maximum absolute error has been demonstrated to approach five metres at fifty metres range, the decision to limit sensing to this point is justified.
- The steadily increasing maximum error but fairly stable average error suggests a source of inaccuracy which is highly influential but also irregular and brief. This directly supports the idea that variance in the pitch of of the camera is the biggest cause of error, even with the inclusion of visually derived horizon correction. An application of filtering could be capable of resolving the position estimation error, either within the image plane or after the transform is complete.
- The fairly constant average error suggests that the overall accuracy of both the IPM and the temporally-extracted cluster border is acceptable. Further testing using a static camera would better determine the maximum effective range without the inclusion of additional sources of error.

9.5 Conclusions

This chapter has covered a basic process for extracting position data from the classified image. At the conclusion of the classification process, each cluster within the image has either been assigned a known class, or identified as an unknown cluster. With the assumption that an unknown clusters is likely to represent a collision risk, a position estimate of each cluster relative to the camera is sought.

As static elements such as buildings are commonly classed as unknowns despite having their own class, a class amalgamation is used to combine them into a single layer. This is done with the intention of providing an output which consists of just navigable space and detected risks, similar to the range and bearing output produced by a LIDAR system.

To combat the effects of single frame classification error, temporal smoothing has been applied by combining the position of clusters in sequential frames, whilst removing clusters which don’t consistently appear. After reviewing the distance through which an object is likely to move per frame, a four frame image buffer was selected. In combining the perimeter of a cluster from multiple frames, a better representation of the actual position of the cluster is achieved.

With the final position of clusters considered to be accurate, depth extraction methods were reviewed for monocular cameras. After a brief investigation, it is concluded that there are very few methods of accurately determining the distance to unknown objects at range using a monocular camera. Therefore, as IPM has been demonstrated to produce good results in the associated field of self-driving cars, the same approach was adopted here. The equations and underlying principles of IPM are stated and its limitations are reviewed with specific consideration given to the maximum effective range. Due to the depth estimation being based directly on pixels, the resolution of the image and proximity of the cluster to the horizon line limit then accuracy of extraction. As such, a maximum viable detection of 50 metres was selected.

To help limit the influence of error on the detected range of objects, the principle of assigning the distance of the nearest point to the entire object was stated. In addition, the relative merits of image wide transforms using a homogeneous perspective transformation were discussed, with use cases including the suitability for localisation through map matching. Finally, real-world data is presented having been extracted from the data set and using GPS co-ordinates to provide a comparison. From the test results, the main concern is the quickly varying accuracy due to vibrations. Therefore, methods of stabilizing the position estimate, either within the image plane or world space will be required.

In conclusion, a method of determining the position of unknown collision risks relative to the aircraft has been demonstrated, typically accurate to within five metres at fifty metres range. As the position of collision relative to the aircraft is now known, localising the aircraft within the aerodrome is the next step. This will involve the combination of directly extracted data with prior knowledge about the aerodrome layout, which have been deliberately kept separate up until this point.

Chapter 10

Conclusions

10.1 Summary

This thesis has approached the issue of civil surface operations for UAS, specifically with with the intention of producing an ATS. It has been predicted that UAS will require this capability in order to share ground facilities with manned aircraft in the coming decade. This thesis has considered the extreme case, in which no specific accommodation is made for UAS activities, with all capability located on board the aircraft. In addition, limitations were placed upon this work both in terms of regulations and hardware; with only a forward facing monocular camera available for sensing, and with all interpretation achieved using deterministic techniques.

The ability of a UAS to move through an aerodrome is highly dependant on the ability to perceive risks and self-localise. Although localisation is already aided by external systems such as GPS, there is no existing method for UAS to detect other vehicles whilst on the ground. For collision risk detection to be meaningful, it must not be limited to only detecting specific objects. Therefore, this thesis has endeavoured to produce a generic method of detecting risk. As visual localisation is also a useful output, generic object detection through terrain classification was chosen as the focus of this thesis.

The terrain classification problem is identified as belonging to the field of semantic segmentation, in which every pixel within the input image should be classified. This has been broken down into three distinct stages; segmentation, data extraction and classification. A segmentation approach has been developed based upon the existing SLIC superpixel segmentation technique in addition to re-clustering using graph based reachability. For each cluster created in this process, data is then extracted and provided to a Bayesian network for probabilistic classification. It has been assumed that a prior survey of the aerodrome environment will be required in order to obtain the training data for this process.

Data extraction for texture involves a texton based approach, in which a texture response is generated for each pixel, allowing each cluster to be classified through Support Vector Machine (SVM) voting within a BDT. As evidence for the Bayesian network must be provided in a probabilistic form, a novel method of assigning representative probability is introduced, based on the position of the texture point (texton) within the texture feature space and its distance to class boundaries. Further data extraction includes the use of NRL, a novel fusion of distance and brightness data for ground pixels, using horizon detection to both re-normalise the luma component whilst also countering the effects of atmospheric scattering.

The classification stage consists of a Bayesian network, believed to be novel in the area of detecting generic collision risks through terrain classification. A sub-network framework is introduced, in which each sub-network provides an independent estimate of class, allowing a manually defined CPD to determine the final result by weighting the results as appropriate. As the winning classification is associated with a confidence in the final result, a simple threshold is used to distinguish uncertainties within the final result. This has been shown to be an effective and reliable method of detecting unknown objects within images. For extraction into the world-space, a brief review of monocular depth extraction techniques has concluded that IPM is the most suitable, capable of localising extracted objects within five metres at fifty metres range.

10.2 Contributions

As surface operations for civil UAS is not an area of previous academic research, the application area itself can be considered novel. However, this thesis has made more specific contributions to knowledge:

1. A deterministic machine vision system for semantic segmentation of outdoor scenes, using a Bayesian Network.

2. A novel method of graphical reachability, intended for use in combining superpixels into larger regions without sudden changes in colour.

3. The specification of Normalised Relative Luminance (NRL) and its relationship with distance for surface marking extraction.

4. The representative probability calculation for texton classification data, specifically for converting a BDT based SVM classification into a probabilistic output.

10.3 Recommendations for future work

10.3.1 Further testing - Multi-platform and Other Locations

Throughout this work, the development and testing have been undertaken using the data captured by BAE Systems equipment. Although highly appropriate for the task, this limited dataset only validates the machine vision approach for this hardware configuration. As the system is intended to provide a generic solution which can be used on many aircraft without the need for hardware modification, further testing with multiple camera types should be undertaken. In addition, as the mounting height of the camera directly effects the range that can be observed within the image, a study into the necessary height requirement for the camera should also be undertaken.

Furthermore, as Walney Island Airport was the single aerodrome from which images were captured, further testing at highly contrasting civil aerodromes is required to truly validate the approach. With UAS being globally developed, this would ideally include aerodromes from many different countries. Not only should this include the effects of differing climates, weather conditions and terrain, but it would also provide information on the surface markings and aerodrome conventions used internationally.

10.3.2 Expansion to a Full ATS

As is likely apparent, there is still much work required in order to produce a full ATS. Specifically for this thesis, additional chapters pertaining to visual localisation were not completed due to the limitations of time. The original concept consisted of a particle filter based tracking system, to facilitate map matching in addition to tracking potential collision risks over time. With both collision risk detection and improved localisation achieved, the remaining major element would be a decision making process which would be required to allow the UAS to act on the data without direct human input. Prior to the future recommendation suggestion below, the rest of the ATS system should be completed, to verify that automated taxiing is possible under the original constraints assigned to this work.

10.3.3 Deep Learning

Despite much of this thesis being specifically tailored to only use method which avoid deep learning, the future viability of these methods should be recognised. When considering the recent advent of ANN and the many benefits they bring it is clear that the future of autonomous vehicles will likely depend on machine-learning. With related technology fields (such as self driving cars) already making use of machine learning for control, it is likely that future aircraft will do the same.

As the potential risks of an single aerodrome collision are far greater than the equivalent dangers in a SDC, only systems with the highest level of certainty will be permitted to operate. As such, any review of machine-learning for use in surface operations should ideally focus on the confidence of the output.

10.3.4 Task Specific Hardware

As this work was aimed to function with specific UAS hardware, the available sensors were defined from the start. Although a review of other sensor types concluded that visual camera systems were amongst the most appropriate for the task, there has been little investigation into alternative configurations.

Although this could include other sensor types, an investigation into the use of additional cameras would likely be more appropriate as a follow up to this thesis. For example, the inclusion of cameras positioned to provide continuous views around the aircraft would allow the existing system to detect risks in all directions.

Additional experiments in the use of stereoscopic depth extraction would also be recommended. As the operating range of stereoscopic depth extraction depends on the separation between cameras, replacing the singular monocular camera with dual cameras would only allow for short range depth estimation, suitable for immediate risk detection. However, as aircraft have wide wingspans, an additional study into wing tip mounted cameras may provide long range stereo vision for aircraft, suitable for both use on the ground and when airborne. As wing tips are highly dynamic, the tracking of the relative motion of the cameras to each other would be a challenging problem.

References

[1] National Transportation Safety Board . Aviation accident database and synopses. http://www.ntsb.gov/_layouts/ntsb.aviation/index.aspx, 2016.

[2] UAV Systems International. Drone laws by country, 2017. https://uavsystemsinternational.com/drone-laws-by-country/.

[3] West Wales Airport. The nation's centre for unmanned systems operations, 2017. http://www.flyuav.co.uk/.

[4] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurélien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC Superpixels. Technical report, EPFL, 2010.

[5] Edward H Adelson. Lightness perception and lightness illusions. *The New Cognitive Neurosciences*, pages 339–351, 2000.

[6] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(12):2037–2041, 2006.

[7] Omar S Al-Kadi. Combined statistical and model based texture features for improved image classification. In *Advances in Medical, Signal and Information Processing, 2008. MEDSIP 2008. 4th IET International Conference on*, pages 1–4. IET, 2008.

[8] Jose M. Alvarez, Theo Gevers, Yann LeCun, and Antonio M. Lopez. *Computer Vision – ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VII*, chapter Road Scene Segmentation from a Single Image, pages 376–389. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-33786-4.

[9] JoseM. Alvarez, Yann LeCun, Theo Gevers, and AntonioM. Lopez. Semantic road segmentation via multi-scale ensembles of learned features. In Andrea Fusiello, Vittorio Murino, and Rita Cucchiara, editors, *Computer Vision – ECCV 2012. Workshops and Demonstrations*, volume 7584 of *Lecture Notes in Computer Science*, pages 586–595. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-33867-0. doi: 10.1007/978-3-642-33868-7_58. URL http://dx.doi.org/10.1007/978-3-642-33868-7_58.

[10] A. Angelova, L. Matthies, D. Helmick, and P. Perona. Fast terrain classification using variable-length representation for autonomous navigation. In *Computer Vision*

and Pattern Recognition, 2007. CVPR '07. IEEE Conference on, pages 1–8, June 2007. doi: 10.1109/CVPR.2007.383024.

[11] anyTRACS Airport Technologies. Jettracs - we move airline business! Press Release, 2011.

[12] Astraea. Autonomous systems technology related airborne evaluation and assessment, year = 2015, url = http://astraea.aero/, howpublished = 2015-02-05.

[13] UK Civil Aviation Authority. Guidance on registration and airworthiness approval (where necessary) for unmanned aircraft. 2017-02-01, 2017. URL https://www.caa.co.uk/Commercial-industry/Aircraft/Unmanned-aircraft/Large-unmanned-aircraft/.

[14] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[15] M. Bertozzi and A. Broggi. Gold: a parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing*, 7(1): 62–81, Jan 1998. ISSN 1057-7149. doi: 10.1109/83.650851.

[16] Jeff Bilmes. On virtual evidence and soft evidence in bayesian networks. 2004.

[17] C.A. Bouman and M. Shapiro. A multiscale random field model for bayesian image segmentation. *Image Processing, IEEE Transactions on*, 3(2):162–177, Mar 1994. ISSN 1057-7149. doi: 10.1109/83.277898.

[18] O. Bourquardez and F. Chaumette. Evaluation of an autonomous taxi solution for airport operations during low visibility conditions. In *Ninth USA/Europe Air Traffic Management Research and Development Seminar*, 2011.

[19] Louisa Brooke-Holland. Overview of military drones used by the uk armed forces. In *House of Commons Library*, October 2015.

[20] John Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-8(6):679 –698, nov. 1986. ISSN 0162-8828. doi: 10.1109/TPAMI.1986.4767851.

[21] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. *Computer Graphics Forum*, 22(3):223–232, 2003. ISSN 1467-8659. doi: 10.1111/1467-8659.00669. URL http://dx.doi.org/10.1111/1467-8659.00669.

[22] Dong chen He and Li Wang. Texture unit, texture spectrum, and texture analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 28(4):509–512, Jul 1990. ISSN 0196-2892. doi: 10.1109/TGRS.1990.572934.

[23] Bruce T Clough. Metrics, schmetrics! how the heck do you determine a uav's autonomy anyway. Technical report, DTIC Document, 2002.

[24] M. Coombes, W. Eaton, and W. H. Chen. Colour based semantic image segmentation and classification for unmanned ground operations. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 858–867, June 2016. doi: 10.1109/ICUAS.2016.7502570.

[25] M. Coombes, W. Eaton, and W. H. Chen. Unmanned ground operations using semantic image segmentation through a bayesian network. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 868–877, June 2016. doi: 10.1109/ICUAS.2016.7502572.

[26] Matthew Coombes, William Eaton, and Wen-Hua Chen. Machine vision for uas ground operations. *Journal of Intelligent & Robotic Systems*, pages 1–20, 2017. ISSN 1573-0409. doi: 10.1007/s10846-017-0542-5. URL http://dx.doi.org/10.1007/s10846-017-0542-5.

[27] Robert G Cowell, Philip Dawid, Steffen L Lauritzen, and David J Spiegelhalter. *Probabilistic networks and expert systems: Exact computational methods for Bayesian networks*. Springer, 2007.

[28] Gabriela Csurka and Florent Perronnin. An efficient approach to semantic segmentation. *International Journal of Computer Vision*, 95(2):198–212, 2011. ISSN 0920-5691. doi: 10.1007/s11263-010-0344-8. URL http://dx.doi.org/10.1007/s11263-010-0344-8.

[29] Kosmas Dimitropoulos, Nikos Grammalidis, Dimitrios Simitopoulos, Niovi Pavlidou, and M Strintzis. Aircraft detection and tracking using intelligent cameras. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 2, pages II–594. IEEE, 2005.

[30] Cheng-Jin Du and Da-Wen Sun. Comparison of three methods for classification of pizza topping using different colour space transformations. *Journal of Food Engineering*, 68(3):277 – 287, 2005. ISSN 0260-8774. doi: http://dx.doi.org/10.1016/j.jfoodeng.2004.05.044.

[31] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. *Conference on Computer Vision and Pattern Recognition*, 2014. URL http://www.cv-foundation.org/openaccess/content_cvpr_2014/papers/Erhan_Scalable_Object_Detection_2014_CVPR_paper.pdf.

[32] T. Geeritsen E.W. Frew Durrie, J. and S. Pledgie. Vision-aided inertial navigation on an uncertain map using a particle filter. In *In Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4189–4194, 2009.

[33] Will Eaton. and Wen-Hua Chen. Image segmentation for automated taxiing of unmanned aircraft. In *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*, pages 1–8, June 2015. doi: 10.1109/ICUAS.2015.7152268.

[34] Ahmed Elgammal, David Harwood, and Larry Davis. *Non-parametric Model for Background Subtraction*, pages 751–767. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. ISBN 978-3-540-45053-5. doi: 10.1007/3-540-45053-X_48. URL http://dx.doi.org/10.1007/3-540-45053-X_48.

[35] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[36] European RPAS Steering Group. Roadmap for the integration of civil remotely-piloted aircraft systems into the european aviation system. Technical report, European RPAS Steering Group, 2013.

[37] FAA. Integration of civil unmanned aircraft systems (uas) in the national airspace system (nas) roadmap. Technical report, Federal Aviation Administration, 2013.

[38] Federal Aviation Administration. Unmanned aircraft systems (uas) for airport operators. *Special Programs*, 2016. https://www.faa.gov/airports/special_programs/uas_airports/.

[39] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004. ISSN 1573-1405. doi: 10.1023/B:VISI.0000022288.19776.77. URL http://dx.doi.org/10.1023/B:VISI.0000022288.19776.77.

[40] Phil Finnegan. Press release: Uav production will total $93 billion. *Teal Group Press Release*, 2015.

[41] F Dan Foresee and Martin T Hagan. Gauss-newton approximation to bayesian learning. In *Neural Networks, 1997., International Conference on*, volume 3, pages 1930–1935. IEEE, 1997.

[42] Eric W Frew, Tristan Gerritsen, Stephen Pledgie, Chris Brinton, Shivang Patel, and Bonnie Schwartz. Vision-based navigation for airfield surface operation. In *AIAA Guidance, Navigation, and Control Conference*, pages 4189–94, 2008.

[43] J. Fuller. *SAFETY WAS NO ACCIDENT: History of the UK Civil Aviation Flying Unit CAFU 1944 -1996*. Trafford Publishing, 2012. ISBN 9781466968936. URL https://books.google.co.uk/books?id=1HzUCRQwSH4C.

[44] Jason Gauci. Obstacle Detection in Aerodrome Areas Through the use of Computer Vision.

[45] Google Maps [Online]. Walney island airport. https://www.google.co.uk/maps/@54.129027,-3.2614247,2558a,35y,353.53h/ Accessed on: 2017-02-01, 2017.

[46] Stephen Gould, Jim Rodgers, David Cohen, Gal Elidan, and Daphne Koller. Multi-class segmentation with relative location prior. *Int. J. Comput. Vision*, 80(3):300–316, December 2008. ISSN 0920-5691. doi: 10.1007/s11263-008-0140-x. URL http://dx.doi.org/10.1007/s11263-008-0140-x.

[47] R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, May 1979. ISSN 0018-9219. doi: 10.1109/PROC.1979.11328.

[48] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, Nov 1973. ISSN 0018-9472. doi: 10.1109/TSMC.1973.4309314.

[49] Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33 (12):2341–2353, 2011.

[50] D.A.P Hersman. *National Transportation Safety Board - Safety Recommendations*.

[51] Honeywell and Safran. Electric green taxiing system - electric green taxiing system increasing aircraft efficiency with a reduced environmental impact. Press Release, 2011.

[52] Thomas Popham Keith J Burnham Ionut Gheorghe, Weidong Li. Superpixel based semantic segmentation for assistance in varying terrain driving conditions. *Progress in Systems Engineering*, 2015.

[53] S Jewell. *ASTRAEA - Opening the airspace for UAS*. BAE Systems, June 2012.

[54] Gang Yi Jiang, Tae Young Choi, Suk Kyo Hong, Jae Wook Bae, and Byung Suk Song. Lane and obstacle detection based on fast inverse perspective mapping algorithm. In *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, volume 4, pages 2969–2974 vol.4, 2000. doi: 10.1109/ICSMC.2000.884452.

[55] Joint Planning and Development Office. Nextgen avionics roadmap version 2.0. 2011. URL http://www.dtic.mil/dtic/tr/fulltext/u2/a561244.pdf.

[56] Bela Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802):91–97, 1981.

[57] Harshad Harshad Dilip Khadilkar. *Networked control of aircraft operations at airports and in terminal areas*. PhD thesis, Massachusetts Institute of Technology, 2013.

[58] Alireza Kheyrollahi and Toby P. Breckon. Automatic real-time road marking recognition using a feature driven approach. *Machine Vision and Applications*, 23 (1):123–133, Jan 2012. ISSN 1432-1769. doi: 10.1007/s00138-010-0289-5. URL http://dx.doi.org/10.1007/s00138-010-0289-5.

[59] Kiyosumi Kidono, Jun Miura, and Yoshiaki Shirai. Autonomous Visual Navigation of a Mobile Robot Using a Human-Guided Experience. 40(2001):1–11, 2002.

[60] Hak Gu Kim, Seung Ji Seo, and Byung Cheol Song. Multi-frame de-raining algorithm using a motion-compensated non-local mean filter for rainy video sequences. *Journal of Visual Communication and Image Representation*, 26:317 – 328, 2015. ISSN 1047-3203. doi: https://doi.org/10.1016/j.jvcir.2014.10.006. URL http://www.sciencedirect.com/science/article/pii/S1047320314001655.

[61] Praveen Kollaikal, Sridevi Ravuri, and Eddie Ruvinsky. Connected cars. *Sutardja Center for Entrepreneurship and Technology*.

[62] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques.* MIT press, 2009.

[63] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia. Available from: <http://www.csse.uwa.edu.au/~pk/research/matlabfns/>.

[64] EL Leblanc. Automated aircraft towing vehicle system. US Patent 6305484B1, 2001.

[65] Thomas Leung and Jitendra Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001. ISSN 0920-5691. doi: 10.1023/A:1011126920638.

[66] D. Liu and J. Yu. Otsu method and k-means. In *2009 Ninth International Conference on Hybrid Intelligent Systems*, volume 1, pages 344–349, Aug 2009. doi: 10.1109/HIS.2009.74.

[67] Fei Liu, Dongxiang Xu, Chun Yuan, and W. Kerwin. Image segmentation based on bayesian network-markov random field model and its application to in vivo plaque composition. In *Biomedical Imaging: Nano to Macro, 2006. 3rd IEEE International Symposium on*, pages 141–144, April 2006. doi: 10.1109/ISBI.2006.1624872.

[68] Greg Loegering and Steve Harris. Landing dispersion results global hawk auto-land system. In *AIAA s 1st Technical Conference and Workshop on Unmanned Aerial Vehicles*, 2002.

[69] T. Ludwig, B. Korn, and R. Geister. Towards higher levels of automation in taxi guidance: Using gbas terminal area path (tap) messages for transmitting taxi routes. In *Digital Avionics Systems Conference (DASC), 2011 IEEE/AIAA 30th*, pages 4B5–1–4B5–11, Oct 2011.

[70] G. Madzarov and D. Gjorgjevikj. Multi-class classification using support vector machines in decision tree architecture. In *EUROCON 2009, EUROCON '09. IEEE*, pages 288–295, May 2009. doi: 10.1109/EURCON.2009.5167645.

[71] S Marĉelja. Mathematical description of the responses of simple cortical cells*. *JOSA*, 70(11):1297–1300, 1980.

[72] Cerman Martin. Background subtraction using running gaussian average: a color channel comparison. In *Seminar aus Bildverarbeitung und Mustererkennung*, 2011.

[73] J. C. McCall and M. M. Trivedi. An integrated, robust approach to lane marking detection and lane tracking. In *IEEE Intelligent Vehicles Symposium, 2004*, pages 533–537, June 2004. doi: 10.1109/IVS.2004.1336440.

[74] M. A. Mohammad, I. Kaloskampis, and Y. Hicks. Evolving gmms for road-type classification. In *2015 IEEE International Conference on Industrial Technology (ICIT)*, pages 1670–1673, March 2015. doi: 10.1109/ICIT.2015.7125337.

[75] Mahmud Abdulla Mohammad, Ioannis Kaloskampis, and Yulia Hicks. New method for evaluation of video segmentation quality. In *Proceedings of the 10th International Conference on Computer Vision Theory and Applications (VISIGRAPP 2015)*, pages 523–530, 2015. ISBN 978-989-758-089-5. doi: 10.5220/0005306205230530.

[76] G.J. Morgan-Owen. Differential gps positioning. *Electronics Communication Engineering Journal*, 7:11–21(10), February 1995. ISSN 0954-0695. URL http://digital-library.theiet.org/content/journals/10.1049/ecej_19950104.

[77] Robert Morris, Mai Lee Chang, Ronald Archer, Ernest V Cross, Shelby Thompson, Jerry Franke, Robert Garrett, Waqar Malik, Kerry McGuire, and Garrett Hemann. Self-driving aircraft towing vehicles: A preliminary report. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[78] Kelly Murphy. Proxy incorporates autonomous aircraft taxiing system. November 2012.

[79] MVTec Software GmbH. *HALCON Solution Guide 1*. MVTec Software GmbH, München, Germany, München, Germany, sixth edition, 2016.

[80] Ashley Napier, Peter Corke, and Paul Newman. Cross-calibration of push-broom 2d lidars and cameras in natural scenes. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, May 2013.

[81] NTSB. Review of U.S. civil aviation accidents. Technical Report NTSB/ARA-12/01, National Transportation Safety Board, January 2010. URL http://www.ntsb.gov/doclib/reports/2012/ARA1201.pdf.

[82] NTSB. *Review of U.S. Civil Aviation Accidents*. March 2011.

[83] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[84] J Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

[85] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10 (3):61–74, 1999.

[86] AG Ramakrishnan, S Kumar Raja, and HV Raghu Ram. Neural network-based segmentation of textures using gabor features. In *Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on*, pages 365–374. IEEE, 2002.

[87] C Yuheng Ren and Ian Reid. gslic: a real-time implementation of slic superpixel segmentation. *University of Oxford, Department of Engineering, Technical Report*, 2011.

[88] Ricardo. Iai awards ricardo contract to support next phase of taxibot development. Press Release, 2011.

[89] Steve. Roberts. Development of uav sensors and information management. *RUSI Defence Systems*, June 2010.

[90] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. ISSN 1573-1405. doi: 10.1007/s11263-015-0816-y. URL http://dx.doi.org/10.1007/s11263-015-0816-y.

[91] Boeing Airplane Safety. Statistical summary of commercial jet aircraft accidents: Worldwide operations, 1959-2014. *Boeing Commercial Airplane, Seattle, WA*, 2014.

[92] Ian Savage. Comparing the fatality risks in united states transportation across modes and over time. *Research in Transportation Economics*, 43(1):9–22, 2013.

[93] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. Learning depth from single monocular images. In *Advances in Neural Information Processing Systems*, pages 1161–1168, 2005.

[94] William Robson Schwartz, Fernando Roberti de Siqueira, and Helio Pedrini. Evaluation of feature descriptors for texture classification. *Journal of Electronic Imaging*, 21(2):023016–1–023016–17, 2012. doi: 10.1117/1.JEI.21.2.023016. URL http://dx.doi.org/10.1117/1.JEI.21.2.023016.

[95] N. Sebe, I. Cohen, T.S. Huang, and T. Gevers. Skin detection: a bayesian network approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 903–906 Vol.2, Aug 2004. doi: 10.1109/ICPR.2004.1334405.

[96] FV Seysses. Method and system for automatically managin a convoy of aircraft during taxiing. Airbus Operations SAS, US Patent 8229604B2, 2012.

[97] Khaled M. Shaaban and Nagwa M. Omar. Region-based deformable net for automatic color image segmentation. *Image and Vision Computing*, 27(10):1504 – 1514, 2009. ISSN 0262-8856. doi: http://dx.doi.org/10.1016/j.imavis.2009.02.003. URL http://www.sciencedirect.com/science/article/pii/S0262885609000183. Special Section: Computer Vision Methods for Ambient Intelligence.

[98] Khaled M. Shaaban and Nagwa M. Omar. 3d information extraction using region-based deformable net for monocular robot navigation. *Journal of Visual Communication and Image Representation*, 23(2):397 – 408, 2012. ISSN 1047-3203. doi: http://dx.doi.org/10.1016/j.jvcir.2011.12.001. URL http://www.sciencedirect.com/science/article/pii/S1047320311001635.

[99] J. Shen, X. Hao, Z. Liang, Y. Liu, W. Wang, and L. Shao. Real-time superpixel segmentation by dbscan clustering algorithm. *IEEE Transactions on Image Processing*, 25(12):5933–5942, Dec 2016. ISSN 1057-7149. doi: 10.1109/TIP.2016.2616302.

[100] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008. doi: 10.1109/CVPR.2008.4587503.

[101] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *Int. Journal of Computer Vision (IJCV)*, January 2009.

[102] Shiyu Song and Manmohan Chandraker. Robust scale estimation in real-time monocular sfm for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1566–1573, 2014.

[103] The Joint JAA/EUROCONTROL Initiative on UAVs. *UAV Task-Force Final Report. A concept for European Regulations for Civil Unmanned Aerical Vehicles.* 2004. URL https://www.easa.europa.eu/system/files/dfu/NPA_16_2005_Appendix.pdf.

[104] ThinkDefence. Watchkeeper tactical unmanned aerial system (tuas). 2017-02-01, 2017. URL http://www.thinkdefence.co.uk/watchkeeper-tactical-unmanned-aerial-system-tuas/description/.

[105] T. Thom. *Air Pilots Manual.* Airlife Publishing Limited, 1999. ISBN 9781840371741. URL http://books.google.co.uk/books?id=91VWAAAACAAJ.

[106] Yan Tian, Judith Gelernter, Xun Wang, Weigang Chen, Junxiang Gao, Yujie Zhang, and Xiaolan Li. Lane marking detection via deep convolutional neural network. *Neurocomputing*, 280:46 – 55, 2018. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2017.09.098. URL http://www.sciencedirect.com/science/article/pii/S0925231217317630. Applications of Neural Modeling in the new era for data and IT.

[107] Luz Abril Torres-Mendez, J. Carlos Ruiz-Suarez, Luis Enrique Sucar, and G Gomez. Translation, rotation, and scale-invariant object recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30(1):125–130, 2000.

[108] Shane Tuohy, D O'cualain, E Jones, and M Glavin. Distance determination for an automobile environment using inverse perspective mapping in opencv. 2010.

[109] UK NATS [Online]. Walney island airport. http://www.nats-uk.ead-it.com/public/index.phpAccessed on: 2017-01-30, 2017.

[110] Unmanned Aircraft System Roadmap. Unmanned aircraft system roadmap: 2005-2030. Technical report, DOD, 2005.

[111] Stefan Vacek, Constantin Schimmel, and Rüdiger Dillmann. Road-marking analysis for autonomous vehicle guidance. In *EMCR*, 2007.

[112] M. Varma and A. Zisserman. Classifying images of materials: Achieving viewpoint and illumination independence. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, volume 3, pages 255–271. Springer-Verlag, May 2002. URL http://www.robots.ox.ac.uk/~vgg.

[113] M. Varma and A. Zisserman. Texture classification: Are filter banks necessary? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 691–698, June 2003. URL http://www.robots.ox.ac.uk/~vgg.

[114] Andrea Vedaldi and Stefano Soatto. *Quick Shift and Kernel Methods for Mode Seeking*, pages 705–718. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 978-3-540-88693-8. doi: 10.1007/978-3-540-88693-8_52. URL http://dx.doi.org/10.1007/978-3-540-88693-8_52.

[115] Yue Wang, Eam Khwang Teoh, and Dinggang Shen. Lane detection and tracking using b-snake. *Image and Vision Computing*, 22(4):269 – 280, 2004. ISSN 0262-8856. doi: http://dx.doi.org/10.1016/j.imavis.2003.10.003. URL http://www.sciencedirect.com/science/article/pii/S0262885603002105.

[116] Tim Warchocki. *Autonomy research for civil aviation : toward a new era of flight*. The National Academies Press, Washington, D.C, 2014. ISBN 978-0-309-30614-0.

[117] Justin Wastnage. Off the radar. *Flight International*, 2004. https://www.flightglobal.com/news/articles/off-the-radar-177441/.

[118] Waukegan National Airport. Walney island airport. http://waukeganairport.com/, 2015.

[119] A Waxman and J LeMoigne. A visual navigation system for autonomous land vehicles. *..., IEEE Journal of*, (2):124–141, 1987. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1087089.

[120] Andreas Wedel, Uwe Franke, Jens Klappstein, Thomas Brox, and Daniel Cremers. *Realtime Depth Estimation and Obstacle Detection from Monocular Video*, pages 475–484. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-44414-5. doi: 10.1007/11861898_48. URL http://dx.doi.org/10.1007/11861898_48.

[121] Xiaolin Wu. Efficient statistical computations for optimal color quantization. *Graphics Gems II*, pages 126–133, 1992.

[122] Brian Wynne. Overview of the unmanned aircraft systems industry. *Association for Unmanned Vehicle Systems International*, 2016.

[123] Song Xue, Xiaojun Jing, Songlin Sun, and Hai Huang. Binary-decision-tree-based multiclass support vector machines. In *Communications and Information Technologies (ISCIT), 2014 14th International Symposium on*, pages 85–89. IEEE, 2014.

[124] Hong-Zhao Yuan, Xiu-Qiong Zhang, and Zi-Liang Feng. Horizon detection in foggy aerial image. In *Image Analysis and Signal Processing (IASP), 2010 International Conference on*, pages 191–194, April 2010. doi: 10.1109/IASP.2010.5476135.

[125] Hong-Zhao Yuan, Xiu-Qiong Zhang, and Zi-Liang Feng. Horizon detection in foggy aerial image. In *Image Analysis and Signal Processing (IASP), 2010 International Conference on*, pages 191–194, April 2010. doi: 10.1109/IASP.2010.5476135.

[126] L. Zhang and Q. Ji. A bayesian network model for automatic and interactive image segmentation. *IEEE Transactions on Image Processing*, 20(9):2582–2593, Sept 2011. ISSN 1057-7149. doi: 10.1109/TIP.2011.2121080.

[127] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr. Conditional random fields as recurrent neural networks. In *International Conference on Computer Vision (ICCV)*, 2015.

[128] Song-Chun Zhu, Cheng-en Guo, Yizhou Wang, and Zijian Xu. What are textons? *International Journal of Computer Vision*, 62(1-2):121–143, 2005. ISSN 0920-5691.

[129] Hanqi Zhuang, R. Sudhakar, and Jen yu Shieh. Depth estimation from a sequence of monocular images with known camera motion. *Robotics and Autonomous Systems*, 13(2):87 – 95, 1994. ISSN 0921-8890. URL http://www.sciencedirect.com/science/article/pii/0921889094900515.