Perceptual Grouping and Knowledge-Based Vision Systems

by

Anthony Tai

Thesis submitted to the University of Surrey
for the degree of
Doctor of Philosophy

Department of Electronic and Electrical Engineering
University of Surrey
Guildford, Surrey
United Kingdom
June 1997
Acknowledgements

I would like to thank all members of the Vision, Speech and Signal Processing group for their support. Special thanks goes to Professor Josef Kittler and Mr Terry Windeatt for their continual support and guidance. Their invaluable suggestions and ideas have been vital during my research. I would also like to thank George Matas and Radek Marik for lending me their software.

I would also like to take this opportunity to acknowledge the financial support offered by the Defence Research Agency (DRA).
Abstract

One of the goals in computer vision is to interpret scene objects and establish relationships between them. One of the problems associated with this task is that the image to be interpreted and the objects to be recognised correspond to different levels of information. The image is, on the one hand, represented as a collection of pixels in which three-dimensional information is transformed into two-dimensional one under perspective projection dictated by the camera position as well as photometric parameters such as focal length etc. On the other hand, the object is represented as a collection of three-dimensional structures and relations between them. These rather different representations highlighted the need to construct an intermediate-level representation which can facilitate the accomplishment of the goal of establishing correspondence between image features and scene objects. The complexity of the interpretation task is further compounded by image imperfections caused by lighting, total reflectance, surface markings, accidental viewpoints and so on.

The problems highlighted earlier motivated the development of a novel feature grouping framework which takes into account feature stability and the underlying noise. This work advanced the state of the art in perceptual group extraction as the existing techniques tend to be ad hoc. Built upon the framework that we have established we developed the computational representation of higher level features such as junctions, collinear line and parallel line groupings.

The low level feature representation and extraction phases of the work were the necessary prerequisites for the extraction of intermediate representations using AI techniques. These representations serve as visual cues in our rule-based system (RBS) to classify runways/taxiways in most of the DRA supplied imagery captured from unknown viewpoints. Complexity problems reported in previous work on RBS for low and intermediate level vision tasks are apparently overcome by identifying a set of prioritised feature cues, uncertainties are handled by hypothesis generation and hypothesis verification, and the method can be regarded as a constrained search through the space of candidate hypotheses.
# Contents

## 1 Introduction
1.1 Intermediate Level Representation ................................................................. 3
1.2 Motivation ........................................................................................................... 5
1.3 Approach ............................................................................................................ 6
1.4 Summary Of Achievements .............................................................................. 8
1.5 Thesis Structure ............................................................................................... 8
1.6 References ....................................................................................................... 10

## 2 Perceptual Grouping
2.1 Introduction ..................................................................................................... 11
2.2 Review of Perceptual Grouping ..................................................................... 13
2.3 Line Representations ....................................................................................... 14
2.4 Monte Carlo Experiment ............................................................................... 16
2.5 Parallel and Collinear Groups, and Junctions ................................................ 21
  2.5.1 Parallel Groups ........................................................................................... 21
  2.5.2 Collinearity ................................................................................................ 22
  2.5.3 Junctions .................................................................................................... 24
2.6 Evidential Support .......................................................................................... 24
2.7 Conclusions ..................................................................................................... 26
2.8 References ....................................................................................................... 27

## 3 Vanishing Point Detection
3.1 Introduction ..................................................................................................... 29
3.2 Review of VP Detection Techniques .............................................................. 30
3.3 Probabilistic Vanishing Point Detection (statVP) .......................................... 32
3.4 Experimental Results ...................................................................................... 37
3.5 Optimised Vanishing Point Algorithm (accum VP) ...................................... 39
3.6 Experimental Results ...................................................................................... 43
3.7 Discussion and Conclusion ............................................................................ 45
3.8 References ....................................................................................................... 48

## 4 Knowledge Based Vision Systems
4.1 Introduction ..................................................................................................... 51
4.2 Expert system building tools ......................................................................... 52
  4.2.1 Expert System Shells ................................................................................ 52
  4.2.2 High-Level Programming Languages ..................................................... 53
  4.2.2.1 Production Systems ............................................................................. 53
5.3 Previous Work ..................................................................................................................................... 77
5.4 Proposed Framework and Methodology ............................................................................................. 82
5.4.1 Hypothesise and Test Paradigm .......................................................................................................... 82
5.4.2 Feature Space Representation ............................................................................................................... 83
5.4.3 System Overview ................................................................................................................................ 85
5.4.4 Rule-Based System ............................................................................................................................ 88
5.4.4.1 Knowledge Rules .......................................................................................................................... 90
5.4.4.2 Control Rules .................................................................................................................................. 93
5.4.5 Implementation ................................................................................................................................ 96
5.5 Results ..................................................................................................................................................... 98
5.6 Conclusions ............................................................................................................................................. 103
5.7 References ............................................................................................................................................ 106

6 Conclusions ............................................................................................................................................. 109
6.1 Summary ............................................................................................................................................... 109
6.2 Contributions ....................................................................................................................................... 111
6.3 Limitations and Future Work .................................................................................................................. 113

Integrated Bibliography
Chapter 1

Introduction

1.1 Intermediate Level Representation

One of the goals in computer vision is to interpret scene objects and establish relationships between them. One of the problems associated with this task is that the image to be interpreted and the objects to be recognised correspond to different levels of information. The image is, on the one hand, represented as a collection of pixels in which three-dimensional information is transformed into two-dimensional one under perspective projection dictated by the camera position as well as photometric parameters such as focal length etc. On the other hand, the object is represented as a collection of three-dimensional structures and relations between them. These rather different representations highlighted the need to construct an intermediate-level representation which can facilitate the accomplishment of the goal of establishing correspondence between image features and scene objects. The complexity of the interpretation task is further compounded by image imperfections caused by lighting, total reflectance, surface markings, accidental viewpoints and so on.
A popular model for vision systems in which a multitude of processes are used in collaboration to extract depth information and other intrinsic physical properties from an image emerged. This representation is known as the \(2\frac{3}{2}D\) sketch \([6][7][8]\). As this representation serves to disambiguate seemingly plausible interpretations of the original image, this representation is used in many vision systems. However, in situations where processes like shape from shading, motion, stereo and so on are unavailable there is a need for an alternative representation.

An alternative route to image recognition through the use of depth recovery is perceptual organisation. Perceptual organisation (note that the term perceptual organisation and perceptual grouping are used interchangeably) is a process which groups together image features into groupings which are deemed perceptually significant or belonging to a single object. Perceptual grouping has its root in the 1920s work in Gestalt psychology \([9]\) which developed a number of demonstrations of grouping phenomena and categorised them into groups. These categorisations are: (1) Proximity - elements that are close together are grouped, (2) Similarity - elements that are similar in colour, orientation and size are grouped together, (3) Closure - tendency for curves to form a closed region, (4) Symmetry - elements that are symmetrical about some axis are grouped together, (5) Continuation - elements that lie along a common line or curve are grouped together, and finally (6) Familiarity - elements that are often seen together are grouped.

Perceptual grouping possesses a few advantages. Firstly, grouping can speed up the recognition process by reducing the combinatorics of the search. Without this grouping stage, a model would need to be matched against all possible combinations of features. Secondly, as groupings are unlikely to have arisen by chance, \(X\) this reduces the chance of producing intermediate level \(X\) structures that can lead to confusion in the higher level interpretation/model invocation module. Finally, grouping is more tolerant to image imperfections and occlusions. Let us imagine a situation where, say, a book is put on top of another and therefore occluding some of the edges of the book at the bottom. In this case, one can see that perceptual organisation can recover from the loss of information due to occlusion of the relevant structure using the collinearity, parallelism and proximity criteria. Witkin and Tenenbaum \([10]\) examined the role of grouping in computer vision systems and suggested that current areas of research such as structure from motion and stereo can both be re-formulated as grouping problems in which related features in the image are grouped into sets. They also claim that much of the interpretation process consists of labelling of perceptual groups.

In order to fully exploit the advantages offered by grouping one needs to be able to define the term perceptual significance. Witkin and Tenenbaum suggest that the extent to which some relations are unlikely to have arisen accidentally is the main component of perceptual significance. For instance, two parallel lines are considered significant if the probability of the
1.2 Motivation

Two lines being parallel is low. Theoretically, there can be an infinite number of relations, however, only those relations that are viewpoint independent are significant as there is no evidence to suggest that non-viewpoint invariant structures are anything other than the result of an accidental interaction between some three-dimensional angles and viewpoint. This viewpoint invariant constraint restricts the type of relations corresponding to meaningful structures. These classes of relations (including relations that remain invariant over a wide range of viewpoints) include collinearity, parallelism and junctions.

Although many computer vision systems [4][6] have incorporated the perceptual grouping process, none of them has given the topic a formal treatment. Instead, ad hoc measures are used to identify and prioritise groupings. This thesis is concerned with the use of perceptual grouping as visual cues to identify man-made objects in aerial images. Perceptual grouping is used due to its insensitivity to viewpoints. This thesis consists of two main parts, the first part formalises the approach to perceptual grouping based on probability. The groupings we consider are parallelism, collinearity and junction. To establish a formalism for grouping we first identify the requirements for the representation of the basic constituent of this set of features, namely a straight line. We then adopt a representation which meets these requirements and experimentally establish the conditions under which the representation is valid. Although curve features are important, in this thesis we will restrict ourselves to groupings of straight lines only. This is due mainly to time constraint and the suitability of straight lines to representation of higher level features such as junctions, collinear line and parallel line groupings, and vanishing points are provided. The second part of the thesis presents a rule-based vision system which uses perceptual groupings and the hypothesise and test paradigm to identify runways and taxiways in aerial images.

1.2 Motivation

The objective of the project was to investigate the role of Artificial Intelligence approaches to the problem of extracting intermediate level structures of straight line segments in aerial imagery. The images supplied for the project (referred to as DRA images) all have high levels of noise and contain man-made objects with varying degrees of discernibility. It was agreed that the main goal would be to develop an automated recognition system capable of extracting polygons and thereby classifying different runway/taxiway structures evident in the images.

As we intend to use feature groups as visual cues to prune the search space it is desirable to use techniques that are theoretically sound. However, a review of existing techniques used to extract feature groups showed that extraction techniques are heuristic based and failed to take into account issues like feature stability and image imperfections.
Considerable efforts have been made to develop theoretically sound techniques for the extraction of feature groups namely, parallel and collinear pairs, junctions and vanishing points. It is particularly important in images supplied by DRA as we need extraction techniques that degrade gracefully in noisy environments.

Previous work on aerial image understanding systems has shown that Artificial Intelligence techniques have been successfully incorporated to represent knowledge and realise flexible control structures. In high/intermediate level vision there is a need to apply model constraints effectively when searching for an interpretation, and systems based on Blackboards, Truth Maintenance, Relaxation Labelling and Constraint Satisfaction have been utilised. The distinctive aspect of the work described here is that two central issues of AI, representation and search, are explored in the context of intermediate level vision where missing data needs to be hypothesised and verified and where only qualitative model information is available.

A rule-based system (RBS) is chosen to implement a hypothesise-and-test paradigm at each level in the feature hierarchy. The assumption is that the task may be regarded as a series of representational transformations interleaved with verification tasks. The advantages of RBS’s are well documented and have been used extensively, but the main criticism in low and intermediate level vision tasks is that they become overly complex (an explosion of rules has been reported) when the need is to deal with uncertain or noisy information, or when incorporating global context. The complexity issue is dealt with here by extracting robust and reliable feature cues, which keep the number of false hypotheses to a manageable number.

1.3 Approach

The work carried out on the project can be broken down into five main sections: Feature definition and representation, hierarchical feature extraction, AI tool comparison, rule-based implementation and experimental evaluation.

The low level feature representation and extraction phases of the project were the necessary prerequisites for the AI extraction of intermediate representations. A literature survey of feature grouping criteria revealed that existing schemes were mostly heuristic and failed to distinguish between various kinds of uncertainties. Consequently representations were inadequate in that they failed to address feature stability and nature of underlying noise. In particular it was necessary to review line representation with respect to its parameters and associated error models. By analysing error distributions, it was concluded that a Gaussian assumption was permissible for lines of sufficient length, and this model was used as a basis for defining other higher level features in the hierarchy.
1.3. Approach

The Vanishing Point (VP, the point of intersection in the image of those straight lines which are mutually parallel in 3D) proved to be a reliable feature cue for detecting runways/taxiways. However, existing VP detection schemes relied on accumulation of line pair junctions and ignored uncertainties of constituent line segments; they work well in well-structured scenes that have strong perspective, for example scenes containing office windows and corridors, but not in most of the DRA images. Consequently a novel probabilistic VP detection technique was developed which takes into account underlying error models. Probabilities of line passing through a common point are computed, and subsequent combination of probabilities allows a confidence measure to be assigned, reflecting the likelihood of being a VP.

The goal of RBS is to take in edgelists as input and provide polygons as output, and for most of the DRA images one pass through the RBS is sufficient to recognise the runway/taxiway structure. Input pre-processing of the edgelists is kept to a minimum so as to preserve generality of the overall system. For example, the thresholds of the edge detector are kept fixed and chosen on the basis of a reasonable compromise between missing information and the number of non-significant lines.

Runways and taxiways are modelled as rectangles under perspective projection, which enables use of what we term bounding pair as a cue for hypothesis generation (in reality the RBS uses polygons rather than rectangles in order to take into account the effects of corners and image imperfections). A bounding pair corresponds to the longer sides of a rectangle, and is only considered after line segments are linked by passing a collinearity criterion. There are, however, circumstances under which the collinearity constraint either fails to link segments or provides improper re-construction. These conditions are detected by the RBS and a combined region growing/boundary detection approach is used to resolve ambiguities. A rule-based control scheme allows boundary detection to be integrated with a form of region growing such that linear segments define and constrain the region growing area. Parameters for seed position, region homogeneity and boundary gap threshold are dynamically determined within the RBS.

Being relatively expensive to compute, the VP is used as part of the verification phase of the hypothesise-and-test framework after un-promising lines have been filtered out. The verification strategies proved to be robust so that heuristic generation of hypotheses could be quite liberally handled. Hypothesis generation is accommodated by heuristics which identify candidate features such as approximate parallelism as well as asserting missing data by end-point proximity or, for example, polygon closure. Heuristics are not too conservative and are aimed at minimising the number of false positives.
For the implementation of the RBS, three categories of tools were identified as candidates, C-based interpreters, Hybrid environments and AI Languages. One from each category, CLIPS, Knowledge Craft and Quintus Prolog was evaluated against an agreed set of criteria including maintainability, extensibility, knowledge representation, C-interface and run-time performance. Although CLIPS was chosen as most suitable, it is emphasised that the runway/taxiway problem did not exercise the tools across the whole range of intermediate level tasks. In particular, uncertainty handling and other knowledge representation issues associated with complexity of reasoning due to model matching might well have resulted in a different implementation choice.

1.4 Summary Of Achievements

The most significant achievement of the project is the successful implementation in CLIPS of a rule-based system that is able to classify runways/taxiways in most of the DRA supplied imagery. Complexity problems reported in previous work on RBS for low and intermediate level vision tasks are apparently overcome by identifying feature cues and developing extraction techniques which take account of the underlying uncertainties. Besides the advantages normally associated with RBS implementations, we single out two that are particularly relevant to the intermediate level vision task:

- convenient common framework for the integration and flexible control of different approaches to solve a problem.
- modular rule structure allowing the effect of parameter tuning to become more transparent, compared with embedding of heuristics in procedural algorithms where unpredicted interactions are more likely to occur.

Other achievements include:

- definition of line representation and error models as basis for extracting hierarchical intermediate level structures.
- Vanishing Point (VP) detection algorithm for use with images where conventional accumulation type VP detection schemes fail and an optimised conventional VP detection scheme to reduce error caused by undersampling.

1.5 Thesis Structure

The rest of the thesis will elaborate on the points mentioned in this chapter. Chapter 2 introduces a line representation appropriate for the purpose of developing an error model which we can use formally to define basic feature groupings. These include junctions, collinear line and parallel line groupings, and vanishing points. The Monte Carlo experiment is conducted to establish the error distribution of one of the parameters assuming that other
parameters are normally distributed. This experiment is one of the main contributions of this chapter. This information is necessary in order to design an “optimal” kernel function for vanishing point detection.

Chapter 3 proposes two new methods for vanishing point detection, the first takes a different perspective to detecting vanishing points as compared with the histogramming (accumulator based) methods. Instead of accumulating intersection points, we compute the probability of a group of lines passing through the same point. This approach provides a probability measure for discriminating between competing hypotheses irrespective of the size of the vanishing group. In addition, its performance also degrades gracefully in noisy environments. The second novel approach is an extension of the accumulation idea which is applicable when a sufficient number of lines intersect at the same vanishing point. The main contribution of the method is that it substantially improves the accuracy of the vanishing point estimate.

Chapter 4 reviews the various classes of expert systems building tools (ESBT), classic topics such as knowledge representations and control mechanisms are also discussed. Several commercially available ESBTs as well as the rationale behind the choice of tool for the implementation of the vision system described in chapter 5 are discussed. The basic concept of knowledge-based vision is reviewed and a review of existing knowledge-based vision systems is provided.

In chapter 5 we present a vision system which is capable of identifying aerial images captured from an unknown viewpoint. The system is based on the hypothesise-and-test framework. This system differs from other hypothesise-and-test methods developed for aerial imagery in the way verification tests have to be devised without using structural constraints derived from a model of the scene.

Finally, chapter 6 provides a summary of the thesis, an assessment of the current achievements, limitations of the current system and proposal for future work.
1.6 References


Chapter 2

Perceptual Grouping

2.1 Introduction

A considerable research effort in computer vision has been aimed at the reconstruction of depth information from 2-dimensional visual input. The main assumption underlying this approach is that the recognition of 3-dimensional objects can easily be carried out by matching against reconstructed 3-dimensional data. Depth information can be obtained both indirectly by matching stereo images and directly by using a laser range finder to acquire 3-dimensional data. Although depth measurements have an important role in visual recognition, they are often expensive to extract or unavailable to the user.

Another approach which has recently been adopted by many researchers involves the extraction of intermediate level information - feature groups. Feature groups encode structural inter-relationships between component elements. These groupings identify structural relationships that are most common among objects of our visual domain and remain invariant in 2D projections over a wide range of viewpoints (note that this property is of particular importance in situations where camera position is not available); these relationships can lead to specific 3D interpretation
and therefore provide useful specific cues to either support or reject a hypothesis. Higher level structures such as parallelograms, which can be built by defining relationships between low level features, can facilitate the process of model invocation. For instance, various groupings correspond to different degrees of accidentalness. This information can be used in designing the model invocation stage.

Perceptual organisation first emerged in the 1920s when Gestalt psychologists studied the ability of human to group perceptually significant structures. A main theme of their research was a set of laws describing what types of perceptual structures are evoked by certain types of pattern. These include the laws of similarity, proximity, good continuation and closure. Unfortunately, a computational theory for perceptual organisation is lacking.

The goal of current research is to explore methods for extracting structures which remain invariant or quasi-invariant over a wide range of viewpoints. They are regarded as ‘perceptually significant’ in the sense that they are unlikely to have arisen by accidents. It is vital that the extracted groupings are meaningful otherwise this would only create confusion during the interpretation process. This also underlies the difficulties in devising a set of theoretically sound definitions for extracting intermediate level features i.e. it is essential to ensure the algorithm for extracting groupings degrade gracefully in the event of noise and occlusion.

The necessity of feature group extraction is underlined by the complexity analysis performed by Tsotsos [22] which showed that object recognition can be brought into the realm of feasibility by means of representing the image content in terms of a hierarchy of image features which range from simple edgels to complex feature groupings. Thus the extraction of features and feature groupings is an important prerequisite of solving any image understanding problem.

In this chapter we address the problem of feature extraction. In the context of our application which is concerned with the detection of man made structures such as runways we concentrate on features such as lines, junctions and parallel lines. The aim of the chapter is to provide an extensive review of the literature on this topic which is presented in section 2.2. We then identify the requirements for the representation of basic constituent of this set of features, namely the straight line, in section 2.3. We then adopt a representation which meets these requirements and experimentally establish the conditions under which the representation is valid. The Monte Carlo validation procedure in section 2.4 is one of the main contributions of this chapter. In section 2.5 we develop the computational representation of higher level features such as junctions and collinear line and parallel line groupings. Finally, in section 2.6 we summarise the main results of this chapter.
2.2 Review of Perceptual Grouping

The difficulty in visual recognition is to establish a match between the object model and the given data. In many cases, the possible set of interpretations is combinatorially explosive and cannot be explored in a reasonable time span. However, by matching groups instead of individual segments, the interpretation tree is pruned to a huge extent. In order to make proper use of perceptual groupings we need to solve the following problems:

- the identification of the appropriate groups,
- to formalise the detection of these groups.

Most papers [1][2][3][6][7][8][9][10][17] approach the first problem by identifying groupings which are deemed to be significant in the sense that it is unlikely to have arisen by accidents. That is, structures in the image that are invariant over a wide range of viewpoints. The groupings used include proximity, parallelism and collinearity etc. In addition to using groupings to build higher level structures they are also used to augment the low level image processing process. Boldt and Weiss [18] introduce geometric grouping of straight lines using a hierarchical linking and merging algorithm. In this context, they refer to a ‘low’ level grouping, which extracts straight line segments. Quadrilaterals [20] are also identified as groupings to facilitate the task of model matching. Whilst most research exploits line groupings, groupings of curve features have also been used. For instance, Dolan and Weiss [19] extend the idea of hierarchical linking and merging to curves. Rosin et al [15] consider the grouping of ellipses generated by circular features under projection. They proposed groupings like parallel planar, solids of revolution etc. The groupings identified are as follows,

- **Parallel planar**: Ellipses that have the same major axis angle and tilt angle.
- **Solids of revolution**: Ellipses which satisfy the parallel planar conditions and whose centres lie on a straight line perpendicular to their major axis.
- **Concentricity**: Ellipses which satisfy the parallel planar conditions and sharing the same centre.
- **Gestalt Grouping Laws**: The grouping laws of proximity, closure, similarity and continuation.

It is obvious that ellipses convey much more information than straight lines and therefore the detection of groupings of ellipses provides strong bottom-up cues for model invocation.

The application of perceptual grouping is not restricted to monocular vision. The use of grouping in stereo vision facilitates the task of matching perspective images. Quan and Mohr [13] exploit geometric constraints and perceptual grouping to reduce the search space during the matching stage. The geometric constraints used are principally perspective information like vanishing point (VP), horizon line and projective coordinates. The perceptual groups are directional group,
collinear group and rays. Thirion and Quan [16] also employ feature groups in the domain of stereo vision to predict the position of a given view with respect to the current model and the correction of noisy features by geometric constraints of feature groups. Mohan and Nevatia [11][12] also exploit the use of groupings in stereo vision. Although the advantage of perceptual groupings has been acknowledged by the vision community the problem of formalising the approach to the identification of these groupings is still unsolved. The papers mentioned above employ only *ad hoc* techniques for the extraction of groupings. None of the papers reviewed take into account the effect of uncertainties or errors on lines or curves. It is vital that the extracted groupings are meaningful otherwise this would only create confusion during the interpretation process. This underlies the difficulties in devising a set of theoretically sound definitions for extracting intermediate level features i.e. it is essential to ensure the algorithm for extracting groupings degrade gracefully in the events of noise and occlusion.

The work in this chapter makes use of error uncertainties in lines and we also develop the computational representation of higher level features such as junctions and collinear line and parallel line groupings.

### 2.3 Line Representations

From the literature survey on feature definitions, it is clear that the proposed schemes are predominantly heuristic. In order to provide a systematic treatment for all feature groupings of interest it is necessary to find a parametric representation as well as a computational representation for line segments. In contrast to approaches found in the literature our computational representation consists of information such as errors associated with the line parameters, evidential support, shapeness of the line etc. The detailed information provides the basis for devising feature definitions. In addition to making the explicit use of such extended information, we also would like a line representation which is easy to transform to and from the standard Hough Transform (HT) $\rho - \theta$ space representation. The Hough transform is commonly used for the detection of image features of a given shape or form. Each shape or form can be described by some parameters which specified precisely the shape of interest. These parameters constitute the transform domain or the parameter space of the Hough transform. Depending on the information available to the Hough transform, each neighbourhood of the image or object surface being transformed will map to a point or a set of points in the Hough parameter space. The Hough transform discretises the Hough parameter space into bins, and counts for each bin how many neighbourhoods on the image or object surface have a transformed point lying in the volume assigned to the bin. Peaks in the parameter space correspond to the likely occurrences of certain shape or form.
The parametric representations that we adopted for a perfect line are

\[ v_1 = [\rho, l, \theta, L]^T \]

\[ v_2 = [x_m, y_m, \theta, L]^T \]

where \( \rho \) is the distance between the foot of the normal and the origin, \( l \) is the distance from the foot of the normal to the line midpoint; \( \theta \) and \( L \) are the line orientation and length respectively. \( x_m \) and \( y_m \) are the coordinates of the line midpoint (see fig. 2.1).

Deriche and Faugeras [4] also address the issue of finding an appropriate line representation. Their goal is the tracking of line segments in a sequence of time-varying images acquired by a camera mounted on a robot. The tracking approach utilizes a Kalman filter to estimate the position in the next image of each token (line segment) in the current image. This defines a search area in which the corresponding measured token may be found. In order to implement an efficient tracking algorithm, an appropriate line representation is required. Since tracking endpoints are unreliable due to the fact that line segments can be broken from one image to another. Essentially, they are looking for a line representation which has independent attributes; so that different Kalman filters can be applied on each parameter.

In order to determine which representation is more appropriate for the tracking algorithm, the covariance matrices associated with vectors \( v_1 \) and \( v_2 \) were found. From these results, it is obvious that the \( \rho, l, \theta, L \) representation leads to the covariance matrix that strongly depends upon the position of the associated line segment in the image through the effect of \( \rho \) and \( l \). That is, segments with the same orientation and length will have significantly different uncertainty on the
2.4. Monte Carlo Experiment

A Monte Carlo Experiment is essentially a synthetic sampling technique. It simply consists of computer algorithms for selecting the samples necessary to compute the value of the random variable in question and a method of organising and displaying the results of a large number of repetitions of the procedure (see table 1 for results). The rationale behind the Monte Carlo experiment here is to establish the error distribution of $p$ assuming that $x$, $y$ and $\theta$ are normally distributed. This is necessary in order to design an “optimal” kernel function which is a function of both $p$ and $\theta$. The values of $p$ were simulated using three equations (the exact formula, the first and second order approximations) as follows,

$$ p + \delta p = (x + \delta x) \cos (\theta + \delta \theta) + (y + \delta y) \sin (\theta + \delta \theta) \quad (2.1) $$

$$ \delta p = -(x \sin \theta - y \cos \theta) \delta \theta + \delta x \cos \theta + \delta y \sin \theta \quad (2.2) $$

$$ \delta p = -\rho \frac{\delta \theta^2}{2} - (x \sin \theta - y \cos \theta) \delta \theta + \delta x \cos \theta + \delta y \sin \theta \quad (2.3) $$

The results of the experiments which model lines of various orientations, lengths and locations in an image, show that the distribution of $p$ ceases to be normally distributed for lines shorter than 4 pixels long. According to equation (2.2) above, the term $-\rho \delta \theta^2 / 2$ is the major factor which affects the distribution of $p$. As a consequence, it is advisable to define the origin of an image at the centre of an image. Note that the reason for performing the test for line segments as short as 4 pixels in length is that the RA $^{1}$ data suggest that it is necessary to include them, in order not to

1. Royal Aircraft Establishment.
discard any meaningful features. It also establishes that the first order approximation is adequate for the purpose of modelling the noise distribution of \( \rho \). Note that the experiments were conducted under the following assumptions,

\[
\sigma_x^2 = \sigma_y^2 = 1 \text{ pixel}^2
\]  
(2.4)

\[
\sigma_\theta^2 = \frac{\sigma^2}{L^2} \quad \text{where } \sigma = 5^\circ
\]  
(2.5)

As mentioned earlier since we possess the necessary statistical parameters for line representation \( v \), we can build the error models which can be utilised for the development of a formal approach to feature grouping namely, parallel pairs, junctions and vanishing point. Equipped with the line representation we can, in principle, develop a formal approach to any complex groupings built upon lines.

A simple analysis involving Taylor series expansion leads to the approximate relationship between the errors in line orientation, line midpoint and the distance from the origin to the foot of the normal shown in equation (2.2). Note that in deriving this equation, we assume that any terms involving the cross product of positional and orientational errors can be neglected. For lines four pixel long or more the quadratic terms become negligible. Equation (2.6) then gives a linear relationship between the errors in orientation \( \theta \), line segment midpoint position and the errors in \( \rho \). Thus if \( \delta \theta \), \( \delta x \) and \( \delta y \) are normally distributed, so will the errors \( \rho \). The covariance matrix shown in equation (2.27) is derived as follows:

To derive the covariance matrix we first derive the equation for \( l \) - the distance from the foot of the normal and the line midpoint \( P_m \) (see fig. 2f).

\[
\rho = x \cos \theta + y \sin \theta
\]  
(2.6)

where \( x \) and \( y \) are the coordinates of the line midpoint.

\[
h^2 = x^2 + y^2
\]  
(2.7)

where \( h \) is the distance between the line midpoint and the origin.

\[
l = \sqrt{x^2 + y^2 - \rho^2}
\]  
(2.8)

From equation (2.6) and (2.8) we have,

\[
l = \sqrt{x^2 + y^2 - x^2 \cos^2 \theta - y^2 \sin^2 \theta - 2xy \sin \theta \cos \theta}
\]  
(2.9)
From equation (2.9) we have,

\[ l = x\sin \theta - y\cos \theta \]  \hspace{1cm} (2.10)

\[ E\{\delta \rho^2\} = E\{ (x\sin \theta - y\cos \theta)^2 \delta \theta^2 + \delta x^2 \cos^2 \theta + \delta y^2 \sin^2 \theta + H\} \]  \hspace{1cm} (2.11)

where \( H \) signifies the higher order terms. As the higher order terms are negligible we have,

\[ E\{\delta \rho^2\} \approx E\{\delta \theta^2\} (x\sin \theta - y\cos \theta)^2 + E\{\delta x^2\} \cos^2 \theta + E\{\delta y^2\} \sin^2 \theta \]  \hspace{1cm} (2.12)

Given equation (2.10) and (2.12) we have,

\[ \sigma^2_\theta \equiv l^2 \sigma^2_\theta + \sigma^2 \]  \hspace{1cm} (2.13)

From equation (2.3) we have,

\[ E\{\delta \rho \delta \theta\} = E\{ -l \delta \theta^2 + \delta x \delta \theta \cos \theta + \delta y \delta \theta \sin \theta\} \]  \hspace{1cm} (2.14)

Neglect the higher order terms and apply the expectation operator we have,

\[ \sigma_{\rho \theta} \equiv -l \sigma^2_\theta \]  \hspace{1cm} (2.15)

To derive \( \sigma_{\rho l} \) we first need to derive \( \delta l \),

\[ l + \delta l = (x + \delta x) \sin (\theta + \delta \theta) + (y + \delta y) \cos (\theta + \delta \theta) \]  \hspace{1cm} (2.16)

After expanding the equation above we have,

\[ \delta l = \delta \theta (x\cos \theta + y\sin \theta) + \delta x \sin \theta + \delta y \cos \theta + H \]  \hspace{1cm} (2.17)

where \( H \) represents the higher terms as before.

\[ E\{\delta \rho \delta l\} \equiv E\{ (-l \delta \theta + \delta x \cos \theta + \delta y \sin \theta) (\rho \delta \theta + \delta x \sin \theta + \delta y \cos \theta)\} \]  \hspace{1cm} (2.18)

After expanding equation (2.18) we have,
2.4. Monte Carlo Experiment

\[ E \{ \delta p \delta l \} \equiv E \{ -l \delta \theta^2 + \delta x \sin \theta \cos \theta + \delta y \sin \theta \cos \theta \} \]  
(2.19)

Assuming that \( \sigma_x = \sigma_y = \sigma \) and applying the expectation operator we have,

\[ \sigma_{pl} \equiv \sigma^2 \sin 2\theta - l \rho \sigma_\theta^2 \]  
(2.20)

\[ E \{ \delta \phi \delta \theta \} = E \{ \rho \delta \theta^2 + \delta x \delta \theta \sin \theta + \delta y \delta \theta \cos \theta + H \} \]  
(2.21)

Neglect all the higher order terms we have,

\[ \sigma_H \equiv \rho \sigma_\phi^2 \]  
(2.22)

\[ E \{ \delta l^2 \} \equiv E \{ \rho^2 \delta \theta^2 + \delta x^2 \sin^2 \theta + \delta y^2 \cos^2 \theta \} \]  
(2.23)

Assuming that \( \sigma_x = \sigma_y = \sigma \) and applying the expectation operator we have,

\[ \sigma_I \equiv \rho^2 \sigma_\theta^2 + \sigma^2 \]  
(2.24)

A Monte Carlo experiment was performed to check the validity of the approximate model and its dependence on line length. From table (1) it is apparent that provided the line length \( L > 4 \) the linear model yields a distribution of errors \( \delta p \) with negligible skew and curtosis which can be taken to imply that it closely approximates a Gaussian. Table (1) shows four statistical parameters related to the first four sample moments from the underlying distribution. These are the mean, the standard deviation, the skew and the curtosis. The first two parameters are well known and will not be described here. The skew [21] is related to the third order moment and is defined as follows:-

\[ Skew = \frac{\sum_{i=1}^{n} (S_i - \mu)^3}{\sigma^{3/2}} \]  
(2.25)

The curtosis [21] is related to the fourth order moment and is defined as follows:-
2.4. Monte Carlo Experiment

\[ \sum_{i=1}^{n} \frac{(S_i - \mu)^4}{\sigma^2} - 3 \]

\[ \text{Curtosis} = \sum_{i=1}^{n} (S_i - \mu)^4 \]

\[ s_i \text{ in both equations (2.25) and (2.26) represent the } \text{i}th \text{ sample.} \]

Skew gives some measure of the symmetry of a density and a symmetrical density will have a skew of zero. Curtosis is related to the fourth order moment and is often used as a test for normality. For a normal distribution we expect curtosis to be small. Note that the 3 in equation (2.26) is introduced such that a normal distribution will give a curtosis of zero. As the values shown in table (1) for both the skew and curtosis are small we can justifiably assume that the distribution of \( \delta \rho \) is Gaussian. Thus if \( \delta \theta, \delta x \) and \( \delta y \) are Gaussian, the joint distribution of \( \delta \rho, \delta \theta \) and \( \delta \ell \) will be Gaussian with covariance matrix

\[
\begin{bmatrix}
\sigma_0^2 + \sigma^2 & -\sigma_0^2 (\sin 2\theta) \sigma^2 - \ell \rho \sigma_0^2 \\
-\sigma_0^2 (\sin 2\theta) \sigma^2 - \ell \rho \sigma_0^2 & \rho \sigma_0^2 \\
(\sin 2\theta) \sigma^2 - \ell \rho \sigma_0^2 & \rho \sigma_0^2 & \sigma_0^2 + \sigma^2 \\
\end{bmatrix}
\]  

(2.27)

where \( \sigma_x^2 = \sigma_y^2 = \sigma^2 \).

From equation (2.27) we can deduce that the distribution of interest \( \rho (\delta \rho \delta \theta) \) is normal

<table>
<thead>
<tr>
<th></th>
<th>mean ( \bar{\rho} )</th>
<th>stdev ( \sigma_\rho )</th>
<th>skew ( \bar{s}_\rho )</th>
<th>curtosis ( \bar{c}_\rho )</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>( L=1.0 )</td>
<td>-0.41122</td>
<td>8.76676</td>
<td>-0.24121</td>
</tr>
<tr>
<td></td>
<td>( L=4.0 )</td>
<td>-0.03572</td>
<td>2.39889</td>
<td>-0.04495</td>
</tr>
<tr>
<td></td>
<td>( L=10.0 )</td>
<td>0.01190</td>
<td>1.32851</td>
<td>-0.00216</td>
</tr>
<tr>
<td>First</td>
<td>( L=1.0 )</td>
<td>0.06199</td>
<td>8.77310</td>
<td>-0.01172</td>
</tr>
<tr>
<td></td>
<td>( L=4.0 )</td>
<td>0.01441</td>
<td>2.40076</td>
<td>0.00618</td>
</tr>
<tr>
<td>approx.</td>
<td>( L=10.0 )</td>
<td>0.00489</td>
<td>1.33047</td>
<td>0.02300</td>
</tr>
<tr>
<td>Second</td>
<td>( L=1.0 )</td>
<td>-0.41220</td>
<td>8.79913</td>
<td>-0.24290</td>
</tr>
<tr>
<td></td>
<td>( L=4.0 )</td>
<td>-0.03577</td>
<td>2.39935</td>
<td>-0.04489</td>
</tr>
<tr>
<td>approx.</td>
<td>( L=10.0 )</td>
<td>-0.00119</td>
<td>1.32850</td>
<td>-0.00228</td>
</tr>
</tbody>
</table>

\[ TABLE 1. \text{Statistical results of error in } \rho (\delta \rho) \text{ obtained from Monte Carlo experiment (line parameters } x_m = -36.6, y_m = 136.6, \theta = 150^\circ). \]

with zero mean and covariance matrix
where \( N \) is the number of points providing evidential support for the line. Note that \( N \) may differ from \( L \), the line length as some pixels inside the line segment may be undetected. The covariance matrix in equation (2.27) applies if the Hough Transform (HT) line detection scheme has an optimisation facility to estimate the most likely values of \( \rho \) and \( \theta \). Note that the covariance matrix is scaled by the factor \( N \) because covariances go down as the number of observations (samples) increases through the properties of mean of random variables.

### 2.5 Parallel and Collinear Groups, and Junctions

Let us consider a set of image lines \( \Lambda = \{ \alpha_i | i = 1...N \} \) where for each line \( \alpha_i \) in the set we have available the following measurements \( (\rho_i, \theta_i, l_i, N_i, H_i, P_i) \) where \( P_i \) is the probability of occurrence which for any image line extracted by the low level processes will be set to unity. \( N_i \) is the number of pixels associated with the line and \( H_i \) is a measure of line quality (shape) evaluated by the Hough Transform (HT) test statistics. \( H_i \) reflects the distribution of edge pixels supporting the line and will assume a maximum value for all edge points lying perfectly on the line modelled with parameters \( \rho_i \) and \( \theta_i \) and it will go to zero if all edges deviate from the perfect line by more than a given threshold.

From the point of view of image understanding we are interested in detecting, in the set of image lines, subsets or groups of lines which have certain non-accidentalness property and may therefore be indicative of man-made structures. The groups of importance, in the first instance, are parallel groups, collinear groups and junctions. In the following subsections, we shall concentrate on these three groups and develop appropriate distribution models.

#### 2.5.1 Parallel Groups

We shall consider the simplest parallel group, i.e. a pair, as any parallel group containing more than two lines can be found and represented by a straightforward application of the results for a pair of lines.

Two lines \( \alpha_i \) and \( \alpha_j \) will be considered parallel if their orientation is identical. Now the nominal orientation associated with line \( \alpha_i \) is \( \theta_i \). However, the actual orientation may be slightly different and it is defined by the distribution function (density) \( p_i(\theta) \) which can be assumed to be Gaussian with mean \( \theta_i \) and variance \( \sigma_{\theta_i}^2 \), i.e.
2.5. Parallel and Collinear Groups, and Junctions

2.5.1 Parallel Groups

The joint probability density of lines \( \alpha_i, \alpha_j \) having orientation \( \theta \) and \( \lambda \) respectively is given by

\[
p_{ij}(\theta, \lambda) = p_i(\theta) \cdot p_j(\lambda)
\] (2.29)

as the two events are independent. Thus the probability density of the two lines being parallel with orientation \( \theta \) is given by

\[
\tilde{p}_{ij}(\theta) = p_i(\theta) \cdot p_j(\theta)
\] (2.30)

Note that \( \tilde{p}_{ij}(\theta) \) is not a density function. It is a cut of the density function \( p_{ij}(\theta, \lambda) \) along the line \( \theta = \lambda \). For \( \tilde{p}_{ij}(\theta) \) to become density, it has to be normalised by the area under \( \tilde{p}_{ij}(\theta) \), under the assumption that \( \tilde{p}_{ij}(\theta) \) is nonzero at least for some \( \theta \). The area equals the probability \( p_{ij} \) of the two lines being parallel, i.e.

\[
p_{ij} = \int p_i(\theta) p_j(\theta) d\theta
\] (2.31)

The probability density of the group of lines \( \alpha_i \) and \( \alpha_j \) being parallel at orientation \( \theta \) is given by

\[
p_{ij}(\theta) = \frac{\tilde{p}_{ij}(\theta)}{p_{ij}}
\] (2.32)

The most probable orientation \( \theta \) corresponds to the argument of \( \max \). \( p_{ij}(\theta) \) in equation (2.12).

2.5.2 Collinearity

For two lines \( \alpha_i \) and \( \alpha_j \) to be collinear they have to be parallel and have an identical \( \rho \)-parameter. A line with nominal parameters \( \rho_i, \theta_i \) will have the actual parameters distributed according to \( p_i(\rho, \theta) \) which under the Gaussian assumption will be

\[
p_i(\rho, \theta) = \frac{1}{2\pi\sqrt{|\Sigma_i|}} \exp \left\{ -\frac{1}{2} [\rho - \rho_i, \theta - \theta_i]^T \Sigma_i^{-1} [\rho - \rho_i, \theta - \theta_i] \right\}
\] (2.33)

where \( \Sigma_i \) from equation (2.27) is given as
2.5. Parallel and Collinear Groups, and Junctions

The joint probability density of lines $\alpha_i, \alpha_j$ at respective distances from the origin of $p$ and $\mu$, and orientations $\theta$ and $\lambda$ is given by

$$p_{ij}(p, \theta, \mu, \lambda) = p_i(p, \theta) \cdot p_j(\mu, \lambda) \quad (2.34)$$

as the two events are independent. Thus for the two lines being collinear at orientation $\theta$ and normal distance $p$ we have

$$\tilde{p}_{ij}(p, \theta) = p_i(p, \theta) \cdot p_j(p, \theta) \quad (2.35)$$

and the probability of the two lines being collinear is given by

$$P_{ij} = \int_0^1 \int_0^{2\pi} p_i(p, \theta) \cdot p_j(p, \theta) \, dp \, d\theta \quad (2.36)$$

The normalised probability density function is then

$$p_{ij}(p, \theta) = \frac{\tilde{p}_{ij}(p, \theta)}{P_{ij}} \quad (2.37)$$

When we merge two line segments into a single segment we needed to represent its parameters. This can be achieved by adopting the argument of the global peak of $p_{ij}(p, \theta)$ as the nominal parameters of the resulting line $\alpha_{ij}$. Note that the distribution of $p_{ij}(p, \theta)$ is not necessarily Gaussian and that the probability of occurrence of the line is no longer unity but instead it is set to $P_{ij}$ in (2.16). The length $L_{ij}$, position $l_{ij}$ and the number of supporting edgels $N_{ij} = N_i + N_j$ will have to be recomputed together with the measure of the line quality $H_{ij}$ at the parameter value $p_{ij}, \theta_{ij}$. The role of this line quality is to provide a shape measure such that all lines can be compared to each other. Furthermore, the quality of higher level structures built by constituent lines can be computed as the average evidential support of the constituent lines. The evidential support of each constituent line is a function of line quality. For details of line quality $H_i$ please refer to [23].
2.6. Evidential Support

2.5.3 Junctions

Given a pair of lines $\alpha_i, \alpha_j$, their intersection will form a potential junction. It is relatively easy to show that the intersection point $(\hat{x}, \hat{y})$ is given by

$$\hat{x} = \frac{\rho_j \sin \theta_j - \rho_i \sin \theta_i}{\sin \theta_j \cos \theta_j - \sin \theta_i \cos \theta_i}$$

(2.38)

$$\hat{y} = \frac{\rho_j \cos \theta_j - \rho_i \cos \theta_i}{\sin \theta_j \cos \theta_j - \sin \theta_i \cos \theta_i}$$

(2.39)

where $(\rho_i, \theta_i)$ and $(\rho_j, \theta_j)$ are the nominal line parameters. Note that as each line is characterised by a distribution $p(\rho, \theta)$ rather than the nominal values only, we would determine the distribution function of the intersection point using the above formulae with $(\rho_i, \theta_i)$ replaced by $(\mu, \lambda)$. However, our application of junctions composed of two lines, the nominal parameter representation will invariably suffice. The discussion of the more complicated case of a triple junction is deferred to chapter 3 where we shall see that the line and junction distributions will play an important role.

2.6 Evidential Support

Evidential support $E(\Omega)$ is, as the name suggests, a measure of the amount of support for an underlying hypothesis. As all our feature groups are defined in terms of line segments, we shall first concentrate on developing evidential support measure for a single line $\alpha_i$.

There are two components which contribute to evidential support. In the first instance it is the line quality $H_i$, but this number does not convey, how much support for a line hypothesis is lent by edgels in relation to line length $L_i$. The support for a line contributed by $N_i$ edge pixels where $N_i = L_i$ is much stronger than if $N_i$ were only a fraction of $L_i$. Thus evidential support $E_i$ for line $\alpha_i$ is given as

$$E_i = \frac{N_i}{L_i} \cdot H_i$$

(2.40)
2.6. Evidential Support

For a feature group defined in terms of \( n \) lines \( \alpha_1, \ldots, \alpha_n \), the evidential support \( E \) is defined as the average evidential support of the constituent lines. The evidential support of a feature group is defined as a straightforward average as there is no reason to believe that one constituent line is more important than another in the formation of the higher level structure.

\[
E = \frac{1}{n} \sum_{k=1}^{n} E_k
\]  
(2.41)

Note that by constituent lines we understand the lines consistent with the group parameters. Thus when calculating the evidential support for a parallel pair composed of lines \( \alpha_i \) and \( \alpha_j \) with group orientation \( \theta \) we first have to evaluate the quality \( H_i \) of each line at this orientation and use this modified quality in computing \( E_i \). Similarly, for junctions we have to update the length of each line forming a junction so that it terminates at the junction. This virtual length \( L_i \) will invariably reduce the evidential support for the line \( \alpha_i \) and the corresponding feature group, with the exception of the T-junction where the length of one of the lines of the T-junction will be unaltered. Let's examine the process of determining the evidential support of the two T-junctions as shown in fig.2.2. Fig.2.2 shows two T-junctions which are identical in terms of the variables which determine the evidential support except that the lines in fig.2.2(a) are further apart than those in fig.2.2b. That is, \( H_i = H_j, N_i = N_j \) and \( L_i = L_j \). From equation (2.40) we can deduce that \( E_i = E_j \). However, here is where the similarities end. The evidential support of the junction group in fig.2.2(a) will be reduced by a larger extent as the virtual length \( L_i \) necessary to form the junction is longer than that of fig.2.2(b). This shows that the evidential support measure we introduced here conforms to human perception of what a 'good' junction is. Now consider the case of a parallel group. In order to compute the evidential support we first have to evaluate the quality \( H_i \) of each line in the group using the group orientation \( \theta \). The modified quality measure would have taken into account the support of edgels in the new orientation \( \theta \) and hence reflects its value in the evidential support metric. For details of line quality \( H_i \) please refer to [23].
2.7 Conclusions

Feature groups have generally been regarded as important immediate level cues for vision systems. However, techniques utilised for extraction of these feature groups tend to be ad hoc. In this chapter, we developed formal techniques for the extraction of these groups. We identified the requirements for the representation of basic constituent of these groups - line in section 2.3. We adopted a line representation which meets these requirements as well as experimentally established the conditions under which the representation is held by conducting the Monte Carlo experiment in section 2.4. Built upon the framework that we have established we developed the computational representation of higher level features such as junctions, collinear line and parallel line groupings. However, due to time constraint it has been unable to incorporate this part of the work into the system described in chapter 5. This, however, serves as the focus of future work and sets the milestone for research in formalising the approach to detection of feature groups.
2.8 References


2.8. References


Vanishing Point Detection

3.1 Introduction

Geometrical cues and constraints provide valuable information as to how certain image features should be interpreted. For instance, in many man made scenes such as airfields with runways there exist a number of straight lines which are mutually parallel in 3D. Under perspective projection these lines will meet at a common point known as the vanishing point (VP). Once this point is identified, one can infer 3D structures from 2D features and this constrains the search for other structures. Also, under known camera geometry the orientation of the lines that are grouped together can be determined from the corresponding VP. Furthermore, two or more vanishing points arising from lines which lie on a certain 3D plane give a vanishing line. This property provides an additional constraint which is particularly relevant when analysing, for instance, aerial imagery where one can often assume that structures of interest lie in a common plane -- the ground plane.
3.2. Review of VP Detection Techniques

The relationship among camera parameters, structures in 3D scenes and VPs has been established by Haralick [4]. The applications of VP analysis range from extracting 3D structures to the calibration of camera parameters [2].

Because of the practical importance of vanishing point detection detections a considerable number of techniques have been developed to solve it. In this chapter these techniques are reviewed in Section 3.2. Most of these techniques are based on histogramming line intersection points. Inevitably, they work successfully only if a sufficient number of lines contribute coherently to the same vanishing point, so that the histogram count for that point is significantly above the background level of random line intersections.

This chapter proposes two new methods for vanishing point detection. The first takes a different perspective to detecting vanishing points as compared with the histogramming (accumulator based methods). Instead of accumulating intersection points, we compute the probability of a group of lines passing through the same point. This approach provides a probability measure for discriminating between competing hypotheses irrespective of the size of the vanishing group. In addition, its performance also degrades gracefully in noisy environments. The chapter is organised as follows. After the literature review in Section 3.2 this chapter introduces the novel vanishing point detection method in section 3.3. In section Section 3.4 we present the experimental results obtained with this method. The second novel approach is an extension of the accumulation idea which is applicable when a sufficient number of lines intersect at the same vanishing point. The main contribution of the method is that it substantially improves the accuracy of the vanishing point estimate. The method is described in Section 3.5 and experimental results validated in section 3.6. Finally, section 3.7 offers some conclusions and discussion.

3.2 Review of VP Detection Techniques

VP is a popular visual cue in computer vision as it helps to identify lines that are parallel in 3D. Other usage of VP includes camera calibration, recovery of rotational component of motion [13][15] and road tracking [5]. An obvious approach to locating VPs is to exploit directly the property that all lines with the same orientation in 3D converge to a VP under perspective transformation. Thus the task of VP detection can be treated as locating peaks in a two dimensional array where the intersections of all line pairs in an image plane accumulate. However, the line pairs can intersect anywhere from points within an image to infinity and this poses an implementation problem.
3.2. Review of VP Detection Techniques

In order to avoid analysing an open space Barnard proposed the projection of image lines onto a Gaussian sphere \([1],[6],[13]\) which neatly represents any 3D orientation. The plane which contains the lens centre and the line segment in the image intersects with the Gaussian sphere centred at the origin to form a great circle. That is a line segment on the image plane is mapped to a great circle. Hence, VPs can be detected as elements on the surface of the Gaussian sphere which have relatively high votings. Obviously, the Gaussian surface has to be partitioned in order to accumulate votes. One popular parameterisation is in terms of the azimuth and elevation angles of the unit vector in the sphere. Note that a uniform partitioning in the Hough space maps to a non-uniform area in the image plane. For example, the elementary areas at the poles are small compared with those at the equator. This implies that some lines may intersect within a larger area and still be grouped together to hypothesise a VP whereas others may not. A crude way of ensuring accuracy is to partition the Hough plane into finer bins. This improvement in accuracy is paid for by an increase in memory requirement and computational load. Moreover, it does not alleviate the problem of inhomogeneity of the Hough Transform.

The Hough based approach mentioned earlier suffers from a speed and accuracy tradeoff. That is, in order to improve the accuracy of vanishing points identified one needs to sample the parameter space more finely. This incurs high computational costs both in terms of memory requirement and amount of operations. In order to circumvent this problem Quan and Mohr \([13]\) propose an algorithm similar to a Fast Hough Transform method. However, detailed experimental studies of hierarchical approaches to vote accumulation in the HT suggest that the steps that need to be taken to ensure the detection of features at all levels may render the technique computationally inferior to standard HT implementation \([12]\). The techniques reviewed so far ignores the issue of noise and uncertainties. Collins and Weiss\([3]\) attempt to take into account the effect of uncertainties by treating the task of VP detection as a statistical estimation problem.

In summary, all existing methods for extraction of VPs perform some form of accumulation of line pair junctions. The Gaussian sphere parameterisation is the most popular, and while being a valued approach, there are several shortcomings associated with it.

As bins in the Hough plane map to non-uniform area on the image plane, some intersection points are grouped together under a more stringent condition than others depending on the locations of the VPs. Problems also arise when votes fall into neighbouring bins which might cause a significant peak to diminish in strength. However, most important of all is that accuracies of detected lines are ignored. As far as VPs are concerned the positional and orientational errors cause incorrect intersection points to be formed which reduce the strength of the ‘true’ peaks and give rise to spurious intersection points which might in turn group with other points to produce ‘false’ vanishing points. Additionally, this would also disperse intersection points (as the bin size
3.3 Probabilistic Vanishing Point Detection (statVP)

has an impact on Hough Transform) which inherently belong to the same VP. The above points show the sensitivity of this approach to noise. Furthermore, due to the nature of the algorithm any convergent group which consists of a relatively small number of 3D parallel lines would be left undetected.

Most research on the topic analyse images of scenes such as offices and corridors which are highly structured and have strong perspective. Consequently, there are fewer potential VPs and the strength of true VPs are significantly higher than the background and are therefore distinguishable from random intersections. In situations where there are a small number of 3D parallel lines, the signal to noise ratio would be too small to reliably extract 3D parallel lines based upon the strength of the Hough peak.

More recently, O’Malony [7] took into consideration the uncertainty in the detection of line segments and designed an isotropic accumulator space where the probability of erroneous VP detection is uniformly distributed throughout all the cells. In any case, the approach is still based on the accumulator array idea and thus is inappropriate for images with sparse parallel lines.

However, there are many domains of applications where the scene contains only a few lines which are parallel in 3D and on such imagery the existing techniques gave results as illustrated in fig.3.1. Fig.3.1(b) shows that there is no dominant peak for the set of lines shown in fig.3.1(a).

In the following sections we develop two novel techniques which overcome some of the problems identified for the various vanishing point finders reviewed above. One is the statVP detection technique which addresses the issue of noise and the other is an accumVP technique which helps to improve the accuracy of VPs identified without incurring high computational costs.

3.3 Probabilistic Vanishing Point Detection (statVP)

Let us consider an image with line segments represented by $\rho - \theta$ parameterisation. Due to the geometrical constraints dictated by the image formation process, all perspectively projected line segments having the same orientation in three space converge to a single point -- the vanishing point -- in the image under a noise free condition. However, both the imaging and low level edge and straight line extraction processes are inherently noisy resulting in uncertainties in the $\rho$ and $\theta$ parameters of the detected lines. Errors in $\rho$ and $\theta$ will result in a considerable scatter of the intersection points of the pairs of line segments which make it difficult to identify true vanishing points. As pointed out earlier, this problem is particularly pertinent when the scene structure contains only a small number of parallel lines.
In this chapter the search for vanishing points makes an explicit use of distribution models of the parameters of the detected lines. With such a probabilistic description for each line we can pose the question of how likely a given point is the common intersection point of a group of lines. In this manner, for any selected group of lines, we can determine the probability $P(x, y)$ for their mutual intersection point $(x, y)$. A vanishing point is then identified as the point which exceeds some pre-specified threshold.

Let us start by considering a single line with parameters $(\rho_i, \theta_i)$ and let the distribution of errors $\delta \rho, \delta \theta$ in $\rho_i$ and $\theta_i$ be $P_i(\delta \rho, \delta \theta)$ respectively. Now the probability of the line passing through a point $(x, y)$ in the image will be given by compounding all the combinations of errors $\delta \rho$ and $\delta \theta$ such that the true line with parameters
satisfies the constraint equation

\[ \rho = x\cos\theta + y\sin\theta \] (3.3)

the compound probability \( p_i(x, y) \) is thus given by

\[
P_i(x, y) = \frac{1}{z_i} \int_{\Pi} P_i(\delta\rho, \delta\theta) \, ds
\] (3.4)

where the integration is performed in the parameter space along the sinusoidal line defined by equation 3.3 and \( z_i \) is the normalising constant to ensure that \( p_i(x, y) \) is a probability density function. In terms of parameter errors the compound probability can be expressed as

\[
P_i(x, y) = \frac{1}{z_i} \int_{\Pi} P_i(\delta\rho, \delta\theta) \sqrt{1 + \left( \frac{d\rho}{d\theta} \right)^2} \, d\theta
\] (3.5)

The compounding process is illustrated in fig. 3.2.

Now let \( X = \{ e_i | i = 1, 2, \ldots k \} \) be a group of lines selected from the best of lines output by an image description process, with the measured parameters for each line denoted by vector \( \mathbf{w}_i = [p_i, \theta_i]^T \) and the associated error distribution by \( p_i(\delta\rho, \delta\theta) \). By analogy the probability
3.3. Probabilistic Vanishing Point Detection (statVP)

![Figure 3.2](image)

Figure 3.2: Compounding of the probability of line passing through point \((x, y)\) is achieved by integrating \(P(\delta \rho, \delta \theta)\) along a sinusoidal path.

that the lines jointly pass through a point \((x, y)\) in the image plane (which extends beyond the physical imaging area of the sensor) is given by

\[
P_i(x, y) = \prod_{i=1}^{k} \frac{1}{\sqrt{2\pi}} \int p_{i}(\delta \rho, \delta \theta) \sqrt{1 + \left(\frac{d\rho}{d\theta}\right)^2} \ d\theta
\]

From the knowledge of \(P_i(\delta \rho, \delta \theta)\) the probability of \(P(x, y)\) can easily be evaluated. Its mode \((x_v, y_v)\) then defines a vanishing point provided \(P(x_v, y_v)\) is above the threshold. Note that in equation (6) we assume the error of lines in the group are independent. This is justified in the results.

In order to develop a practical procedure based on the above idea we first need to select a suitable groups of lines. Regarding these lines, the method is intended for finding vanishing points of small sets of 3D parallel lines, hence the cardinality of the group should be quite small. Moreover, the computational complexity of the problem could potentially grow combinatorially with the number of lines in the group. In the present approach the initial analysis is performed for line triplets. Any larger group of lines is formed after this first analysis stage by considering the proximity of detected vanishing points and the overlap of the two participating line sets.
To prune the set of all possible triplets each candidate group of lines must satisfy a number of criteria. These include:

(i). angular constraints (similarity of $\theta$ values)
(ii). distance constraint (the perpendicular distance of line pair intersection point from the third line)
(iii). junction quality constraint (the lines should intersect at a point which is remote from all participating line endpoints as illustrated in fig.3.3.)
(iv). imaging geometry constraints (if known)

These constraints are designed to prune away any irrelevant lines in our problem domain (aerial imagery). It is apparent that our approach requires an appropriate line representation which can associate uncertainties with its parameters. Such a representation was developed in Chapter 2 where we showed that the distribution of errors $\rho$ ($\delta \rho, \delta \theta$) on line parameters $\rho$ and $\theta$ detected with the Hough transform is normal with zero mean and covariance matrix

$$
\Sigma = \frac{1}{N} \begin{bmatrix}
\sigma^2 + \sigma^2 & -l\sigma^2 \\
-l\sigma^2 & \sigma^2
\end{bmatrix}
$$

(3.7)
where \( N \) is the number of points providing evidential support for the line, \( \sigma^2 \) is the variance relating to the positional accuracy of edge pixels providing the line support and \( \sigma^2_\phi \) is the variance of the orientation errors of the edge pixels. We thus have all the necessary ingredients to apply the method and the next section presents some results obtained with it.

### 3.4 Experimental Results

The method was applied to aerial imagery of resolution 256x256. The rectangular line parameter probability distribution was used in the experiment. The implementation involved the following steps:

(i). Compute the intersection points of all possible pairs and discard those that fall outside the virtual image (this is an imaginary image of size 512X512 centred at the origin).

(ii). Combine a line with a pair to create a triplet to see if it satisfies the pursuing constraints of section 3.2.

(iii). Set up a window around the line pair intersection point and compute the probability \( P(x, y) \) for all \((x, y)\) in the window. Find the mode of \( P(x, y) - (x, y) \)

(iv). Repeat steps (ii) and (iii) for all other triplets.

![Figure 3.4 Vanishing point detection result](image)
The result shown in fig. 3.4 demonstrates that the vanishing points which grouped together the lines of interest (i.e. the runways and taxiways) are ranked highly and their corresponding VPs are in close proximity to each other (these VPs can be grouped to extract larger vanishing groups). The vanishing points (four of them) that are grouped together by proximity are highlighted by the white box and these VPs correspond to lines 1 to 6. Fig. 3.5 shows the probability envelope of a typical VP in image space.

When the conventional VPs finder was applied on the image shown in fig. 3.4, the runways and taxiways were again detected but there is no way of discriminating between the various hypotheses. If the size of the peak were used as a discriminating criterion, false vanishing groups would be extracted.

The parameter space in which the proposed algorithm operates is an open one, since the intersection point of a line pair can lie somewhere between the image and infinity. This, however, should not present much difficulty as lines whose 3D orientations are the same only cease to converge to a point under a very restricted viewpoint. This also demonstrates the relationship between convergent and parallel groups. 3D parallel features when projected onto the image plane can only transform into convergent or parallel groups. Prior to the detection of vanishing points, we filter out lines that have similar orientations by thresholding to avoid the open space problem.
The algorithm proposed in this section provides an estimate of the VP location as well as producing a performance measure. Thus VP hypotheses can be compared and discarded on the basis of their quality measure values rather than the number of lines which converge to a certain point. This offers a common ground for comparison between convergent groups (i.e., small convergent groups can also be detected). However, there is a trade-off between small convergent groups and the probability of accidental coincidence. As the size of the convergent groups reduces, the chance of accidental coincidence increases, which renders the location of true vanishing points more difficult. Therefore some kind of geometrical cue is needed to recover true vanishing points. Vanishing lines are powerful cues to be exploited for the purpose. They are defined as the locus of VPs formed by 3D parallel groups which lie on the same plane. This property means that the hypothesised points must lie on the vanishing line and thereby provide a constraint for discarding false VPs. Nevertheless, it is acknowledged that more tests need to be carried out to demonstrate the effectiveness of the proposed approach.

3.5 Optimised Vanishing Point Algorithm (accumVP)

In this section we propose an adaptation of the standard accumulation method reviewed in Section 3.2 which takes proper account of the effects of noise and errors in line segment parameters. This is achieved by adjusting the way votes are cast during the accumulation. We further introduce a post-processing optimisation method that overcomes the shortcomings described in the review above without incurring high costs due to high sampling frequency of the parameter space. In the next section we describe the method and give estimates of the errors in vanishing point location due to under-sampling. In the third section we show results of our optimised vanishing point detector and show that we can overcome these sampling inaccuracies at cheap computational cost. We demonstrate the accuracy achieved using synthetic imagery and apply the method to two outdoor scenes. In contrast to the method proposed in Section 3.3 the modified accumulation approach is appropriate when a sufficient number of lines in the image intersect at the same vanishing point.

Our vanishing point algorithm uses as input a set of line segments provided by a Hough transform algorithm. The Hough transform was chosen as it is a very robust method for detecting straight line segments, and a fast algorithm has been developed [16]. The input to this routine comes from an edge detector using subpixel accuracy [14] and an optimising filter [10]. In this way we minimise the errors associated with the edge detection phase of the processing. The errors associated with the estimated line parameters from the Hough algorithm are dominated by the sampling frequency in the Hough parameter space. As a result, algorithms have been
developed to remove this uncertainty in the line parameters. One of these methods uses an optimisation scheme [8] and another uses focus of attention inside the Hough algorithm to improve the accuracy on identified groups of lines in the image [9].

In this chapter we describe an algorithm for vanishing point detection which preserves the accuracy maintained at the lower levels of processing. This algorithm employs an accumulator over the two angles defined by:

\[ \alpha = \arctan \frac{y_p}{x_p} \]  
\[ \beta = \arctan \frac{r_0}{r_p} \]

where the vanishing point is located at \((x_p, y_p)\) and \(r_p\) is the radius at which this vanishing point is located in the image plane, taking the origin at the point where the optical axis cuts the image plane. The quantity \(r_0\) is a parameter which is chosen arbitrarily. For vanishing points which lie near the focal axis of the camera, the angle \(\alpha\) is very uncertain and the method fails. We therefore use the radius \(r_0\) to exclude any candidate vanishing points which lie within this circle. This is achieved by only accumulating over angles \(\beta\) in the range \(-\pi/4\) to \(\pi/4\) \((r_p \geq r_0)\). The range of the angle \(\alpha\) is \(-\pi\) to \(\pi\).

An accumulation is performed over this parameter space using all possible pairs of lines from the Hough algorithm. The computational cost is reduced by eliminating pairs of lines which intersect near the centre of the image and lines which are closely parallel. The accumulator is then passed through a routine which suppresses all the non-maximal peaks and then is thresholded. The uncertainty in the two angles \(\alpha\) and \(\beta\) and hence in the location of the vanishing point is dominated by the sampling frequency of the parameter space. In our algorithm we use 150 bins in each of the two parameters. This corresponds to an uncertainty of 0.04 radians in \(\alpha\) and 0.01 radians in \(\beta\). From equation (3.8) and (3.9) we find the location of the vanishing point to be:

\[ x_p = r_0 \cot \beta \cos \alpha \]  
\[ y_p = r_0 \cot \beta \sin \alpha \]
The uncertainty in the angles produces an uncertainty in the location of the vanishing point of \( \Delta x_p \) in \( x_p \) and \( \Delta y_p \) in \( y_p \), where:

\[
\Delta x_p = y_p \Delta \alpha + x_p \left( \frac{r_0^2 + r_p^2}{r_0 r_p} \right) \Delta \beta
\]

(3.12)

\[
\Delta y_p = x_p \Delta \alpha + y_p \left( \frac{r_0^2 + r_p^2}{r_0 r_p} \right) \Delta \beta
\]

(3.13)

It is clear from these expressions that the errors in the vanishing points are small if \( x_p \) and \( y_p \) are comparable and \( r_p \) is comparable to \( r_0 \). The problem is that we cannot fine tune \( r_0 \) without knowing roughly how far out the true vanishing point is beforehand since we discard all candidate vanishing points within radius \( r_0 \). As a result the second term in equations (3.12) and (3.13) are amplified by roughly a factor of \( r_p/r_0 \). Suppose the vanishing point is located at approximately \( x_p \approx y_p \approx 100 \) pixels and \( r_p = 5r_0 \), then the errors in the location of the vanishing point, just due to sampling errors of the parameter space, would be approximately 5%. If, however, the vanishing point were at \( x_p \approx y_p \approx 1000 \) and \( r_0 \) unchanged, then the error would be 27%. Hence the sensitivity of the test depends strongly on the choice of \( r_0 \).

Since we do not know \textit{a priori} how to choose \( r_0 \), we have adopted a scheme which allows us to remove the sampling error after accumulation has taken place. In this way we can reduce the uncertainties \( \Delta \alpha \) and \( \Delta \beta \) in the above formulae to their smallest possible values associated with the accumulated errors from the edge detector and Hough line finders, which have already been optimised. For each pair of lines found in the image, we determine an intersection point and find the corresponding values of \( \alpha \) and \( \beta \). Instead of incrementing the accumulator bin in which this point resides in the usual way, we allow for the uncertainties in the line parameter estimates and spread the vote of the line pair over a number of accumulator bins, dependent upon how long the line segments are, and how far from the ends of these segments the point of intersection was found to be. The vote that the line pair contributes to the accumulator was calculated from a smooth voting kernel which peaks at unity when the intersection point coincides exactly with the centre of the accumulator bin, and falls smoothly to zero. To determine the shape of this voting kernel we view the accumulation process in terms of hypothesis testing [11]. The values of the two angles at the centre of the bin is the hypothesis and the intersection points of line pairs provide the support for the different hypotheses being tested. The shape of the voting kernel is determined by the requirement that the hypothesis has a high probability of being accepted (above threshold) when it
coincides with a true vanishing point in the image, and this probability must fall rapidly to zero as
the hypothesis differs from the true vanishing point location. A detailed analysis of this is given
by Palmer et al [8], with the result that the form of the voting kernel was chosen to be:

\[ K(\delta \alpha, \delta \beta) = F \left( \frac{\delta \alpha}{K_\alpha} \right) F \left( \frac{\delta \beta}{K_\beta} \right) \]  \hspace{1cm} (3.14) 

where

\[ F(x) = 1 - 2x^2 + x^4 \]  \hspace{1cm} (3.15) 

provided \(|x < 1|\) or 0 otherwise, and \(K_\alpha\) and \(K_\beta\) are the predetermined widths over the
parameter space that each line pair is allowed to vote. The values of \(\delta \alpha\) and \(\delta \beta\) are computed
as the difference between the angles equivalent to the intersection point and the angles at the
centre of the current bin in the accumulator.

The hill climbing method we employed is based on making a surface fit to the accumulator
function up to quadratic terms in each of the two parameters. By using the first and second
derivatives of the kernel, which can easily be computed from equations (3.14) and (3.15), the
coefficients of this fit can be determined, and then by differentiating the fit function, the location
of the peak in the fitting surface found. We then move to this peak location and re-evaluate the
accumulator and its derivatives until the difference between our current location in parameter
space, and the estimated location of the local peak coincide within some tolerance.

<table>
<thead>
<tr>
<th>True X</th>
<th>True Y</th>
<th>VP X</th>
<th>VP Y</th>
<th>Opt X</th>
<th>Opt Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>300.50</td>
<td>406.44</td>
<td>309.43</td>
<td>413.89</td>
<td>299.99</td>
<td>405.93</td>
</tr>
<tr>
<td>-43.50</td>
<td>133.42</td>
<td>-43.95</td>
<td>135.20</td>
<td>-44.00</td>
<td>132.84</td>
</tr>
<tr>
<td>-171.50</td>
<td>31.83</td>
<td>-174.21</td>
<td>36.76</td>
<td>-172.01</td>
<td>31.33</td>
</tr>
</tbody>
</table>

*TABLE 2.* Coordinates of the three vanishing points determined from ground truth. In the
middle two columns are the results from the VP detector before optimisation. There is no
noise so this error just represents sampling errors. The last two columns are the locations after
optimisation.
3.6 Experimental Results

In order to test our algorithm for vanishing point detection we created an artificial image containing three groups of seven lines. Each of these lines corresponded to a set of 3D parallel lines which would be coplanar in the 3D world. The orientation of this plane is known, and the locations of the three vanishing points on the image plane are also known. In this way we could test the effects of sampling errors and the accuracy of our optimisation scheme. The locations of the three vanishing points are given in table 1, along with the ground truth values. Also presented are the values obtained from the vanishing point detector before optimisation. We see that the results were significantly improved by the optimisation scheme. The amount of cpu time required to perform the optimisation on all these vanishing points was less than 0.1 secs on a Sparc2.

We show the locations of the three vanishing points found to show how close to collinear they are (see fig.3.8). We determined from these three points the equation of this vanishing line by a least squares fit and hence were able to estimate the orientation of the 3D plane on which these lines would have sat. The orientation of this plane was determined in terms of the three components of the unit vector orthogonal to the plane. From the least squares fit line we obtained (0.5002, 0.6303, 0.5937) for this vector. Ground truth for this orientation was (0.5, 0.63, 0.594). Using the non-optimised values we obtain for the plane orientation (0.4972, 0.6273, 0.5993).

<table>
<thead>
<tr>
<th>True X</th>
<th>True Y</th>
<th>VP X</th>
<th>VP Y</th>
<th>Opt X</th>
<th>Opt Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>300.50</td>
<td>406.44</td>
<td>297.15</td>
<td>394.50</td>
<td>299.80</td>
<td>403.10</td>
</tr>
<tr>
<td>-43.50</td>
<td>133.42</td>
<td>-43.95</td>
<td>135.21</td>
<td>-46.30</td>
<td>133.92</td>
</tr>
</tbody>
</table>

*TABLE 3.* Coordinates of the three vanishing points determined from the Hough line segments. These results include the effects of noise. In the middle two columns are the results from the VP detector before optimisation and the last two columns are the locations after optimisation.

We also ran the image through the edge detector and line finder to see how large the errors were due to these two levels of processing. The values for the locations of the three vanishing points are given in table 2. For comparison the ground truth has been duplicated in the first two columns and the locations before and after optimisation in the next four columns. The first vanishing point has been located very accurately and the largest error in the co-ordinates of the second point is roughly 6.4%. The last point, however, has a larger error of 20% in the y co-ordinate. This large error was traced to the edge detection phase of the processing. We tested that the optimisation procedure was correct by making an exhaustive search and reproduced the same values as in
Apart from this exceptional error, the errors in the other co-ordinates are roughly 6%. This table demonstrates that the effects of optimisation have significantly improved the location estimates of the three vanishing points.

Having demonstrated the improvement obtained on a synthetic image for which ground truth was known, we next applied the algorithm to a real outdoor scene shown in figure 3.6. Using lines obtained for the balconies on the front of the building in the centre of the image, we were able to determine a vanishing point at (286.3, 105.2). Using the optimisation scheme the location of this vanishing point had moved to (286.1, 108.0). We see that in this particular case the x location of the vanishing point coincided very closely with one of the bins although the y co-ordinate shifted by 3%. Averaging over all the vanishing points showed that the optimisation scheme has a 0.1 sec cpu overhead per peak in the accumulator.
Finally, to demonstrate the effects of optimisation on the location of the vanishing point, we applied the scheme to the image in figure 3.7. This shows an infra-red image of a runway taken from an approaching aircraft. There is an important vanishing point obtained from the sides of the runway, which was found to be at (141.2, 2.4) before optimisation and (139.5, 2.6) afterwards.

3.7 Discussion and Conclusion

We have presented two novel vanishing point detection algorithms. The detection of vanishing points in an image using pre-detected straight line segments developed in Section 3.5 allows for uncertainties in the line parameters used and is based on a rigorous hypothesis testing analysis. With this method we have been able to produce a post-processing optimisation level within the detector to remove the errors in vanishing point location due to undersampling the two dimensional parameter space. This optimisation scheme is fast and efficient and allows us to

Figure 3.7 Runway seen from approaching aircraft. Vanishing point is determined from the sides of the runway.
reduce the number of bins in the accumulator array and so speed up the accumulation phase of the detection process. We experimented with reducing the number of accumulator bins and found we could still reproduce the exact locations of the vanishing points by reducing the number of bins to 100 in each parameter. This increased the speed of the algorithm by more than a factor of two. The adaptation of the standard vanishing point detection process is easy to encode.

![Figure 3.8 Least Square fit to 3 vanishing points.](image)

The resulting algorithm can locate vanishing points accurately and efficiently. The main problem remains in determining which line segments in the image should be associated together with a common vanishing point. If the optimisation scheme uses a set of lines of which some are not 3D parallel, then the improvement in accuracy will not, in general, determine the correct location for the vanishing point but a compromise between the different line intersections. To overcome this difficulty requires some higher level knowledge of the image, as any group of lines which are 3D parallel will tend to produce peaks in the accumulator with other spurious line segments in the image. The hypothesis testing voting kernel we have employed in our algorithm is designed to reduce this problem, but clearly this is now the main source of error.
An alternative approach to vanishing point detection (statVP) was proposed in Section 3.3. This method has two main advantages - firstly it works when the image contains few 3D parallel lines. In such an image the accumulator method would produce very small peaks which would be difficult to detect on thresholding. The second main advantage is that it provides a direct measure of the confidence in the detected vanishing points as it takes full account of the uncertainties in the line parameters associated with it. The hypothesis testing accumulation we have described in this chapter is a natural way to extend the accumulation method to take account of line segment errors, and since the final vote of each line pair can vary smoothly between zero and one, the height of the accumulator gives a measure of the support for the hypothesised vanishing point. This accumulator based method (accumVP) therefore is a natural extension of the above method to images where enough parallel lines exist for the accumulation method to be adequate.
3.8 References


3.8. References
4.1 Introduction

Artificial Intelligence (AI) is a fast growing branch of computer science which enables computers to perform tasks that require human intelligence. Many existing AI research themes focus on abstract mathematical problems or simplified real world problems. This is intended to gain insights for further developments and refinements. Amongst AI research, expert systems (Note that we use the terms expert systems and knowledge based systems interchangeably, strictly speaking, expert systems is a branch of knowledge based systems with more specialised applications,) have gained popularity over the years for solving complex real world problems. This stems from the fact that expert systems have been applied successfully in various areas. In contrast to conventional programming techniques in which knowledge is encoded implicitly and mixes the control mechanism and the program code, expert systems embody an explicit representation of knowledge of the application domain and this is kept separate from the code that performs reasoning. These distinct programming modules are commonly known as the knowledge base and the inference engine respectively. In addition to these, the control flow of data within an expert system is non-deterministic and depends on the state of the data and the
4.2. Expert system building tools

Due to the success of expert systems, various kinds of expert system building tools emerge. They include expert system shells, high level programming languages and integrated programming environments. The following sections describe the various ESBTs.

4.2.1 Expert System Shells

Expert system shells are constructed by abstraction from concrete expert systems. That is, they are simply an expert system with the domain specific knowledge and its data removed. A shell, therefore, consists of an inference engine, an empty knowledge base and some debugging facilities. A shell eases the implementation of an expert system significantly, since the builder of the system requires only to encode the knowledge specific to a particular domain using the knowledge representation scheme supported by the shell. However, there is a price to pay for this advantage [22][6][13]; a given knowledge representation would only be best suited to express certain kinds of domain knowledge. That is, the knowledge might not be encoded efficiently, although different knowledge representation formalisms are, in principle, epistemologically equivalent. Another disadvantage is that the static control regime does not necessarily represent the best strategy for the domain concerned. For instance, the appropriate control strategy could be either goal-directed or data-driven depending on the task.
4.2.2 High-Level Programming Languages

High-level languages are, in general, more flexible than an expert system shell, since they provide a higher degree of freedom in terms of control. There are four main kinds of high-level languages such as production systems, logic programming systems, integrated programming environments and blackboard systems.

4.2.2.1 Production Systems

Production systems normally provide only one form of control mechanism. For instance, C Language Integrated Production System (CLIPS) supports only forward chaining. Backward chaining can be emulated by using goal patterns in forward chaining rules and fuse and split various goals (subgoals). This approach results in more rules than would be required by a backward chaining tool. Modularity of production rules are maintained through the use of independent modules to represent knowledge. Also production rules are independent of each other (i.e. rules may not call another rule directly). Some authors take the view that this property implies the ease of adding and deleting rules in the system and therefore enables system maintenance with minimal effort.

4.2.2.2 Logic Programming Languages

Logic programming is based on a theorem prover. A Prolog program, for instance, can be regarded as a collection of formulae in propositional logic with a theorem to be proved. Prolog's syntax is that of the first order propositional logic written in a clause form (a conjunctive form with no quantifier) and further restricted to Horn clauses i.e. clauses with at most one positive literal. Additionally, Prolog also supports non-logical operations such as bagof and setof to collect solutions into a list, and cut to prevent backtracking. A distinct feature of a logic-based representation is the monotonic nature of the knowledge base. This implies that there is no contradiction in the knowledge base. While this is an advantage in many contexts, it turns out to be inappropriate in some applications. Apart from representation in propositional logic, Prolog allows data structures such as frames and semantic networks to be implemented with reasonable ease. The disadvantage associated with first order logic is the lack of any explicit method to index into relevant knowledge since each rule is similar to another.

4.2.2.3 Functional Programming Languages

Functional programming languages such as Lisp enhance modularity since they consist of a collection of function definitions. In this programming style no assignments are manipulated. The hierarchy of functions are organised carefully to construct problem solving procedures. Functional programming is based on the mathematical theory of recursive function, therefore its
correctness can be proved mathematically in a more straightforward manner. This kind of language is essentially procedural so the programmer is required to encode the data flow explicitly.

4.2.2.4 **Integrated Programming Environments**

Integrated programming environments [18][8][6], also called hybrid programming environments, attempt to provide expert system builders with a rich set of programming paradigms. Unlike other ESBTs which provide a single knowledge representation framework, hybrid tools usually provide two or more representation schemes such as frames, rules, logic etc. Procedural knowledge can also be represented using, for example, access oriented programming paradigm -- incorporating demons or active-values within the slots of frames. In terms of reasoning strategies, a hybrid tool supports forward and backward chaining, and sometimes non-monotonic reasoning which allows the user to backtrack to certain dependent data (either to retract certain data or update hypothesis) when the state of certain data become untrue. This is useful when dealing with uncertain knowledge. It is obvious that different representation schemes and inferences have their weaknesses; therefore combining all these schemes should allow the knowledge engineer to optimise and tailor the expert systems depending on the task and the application domains.

![Figure 4.1 Blackboard system.](image)
4.2.2.5 Blackboard Systems

A blackboard system [2] is designed to provide better control, that is, to allow control or meta knowledge to be encoded more explicitly. A blackboard architecture divides the knowledge into modules, each containing related entities and provides a separate inference engine for each module. The interconnection between knowledge modules is done through a global data structure called a blackboard. These open the options for using different knowledge representations for different knowledge modules, and applying various reasoning methods in different stages (see figure 4.1).

4.3 Knowledge Representation

Knowledge representation [7] is an important issue in AI as this has an impact on access, update and maintenance of data. The common knowledge representations are logic, frames, rules, semantic networks and object orientation. The sections below briefly discuss the knowledge representation schemes.

4.3.1 Production Rules

Rules [20][3][6][13] can be seen as a kind of procedural representation which represent knowledge as a set of instructions for problem solving. This contrasts with declarative representations supported by logic and semantic networks. Production rules can be viewed as a special kind of procedure with constrained functionalities. The syntax of production rules is limited by the if... then ... constructs and rules cannot communicate with other rules directly, this can only be done via side-effect on the database. These properties are thought to enhance modularity and maintain the readability of the code. Note that in this kind of system the functionalities of rules are buried. The order of the production rules as well as the ordering of conditions within a rule has a direct impact on the search space traversed. Although rule-based systems are successful in various applications, in order to improve the efficiency of the system it is necessary to use meta-knowledge.

4.3.2 Logic

Representation in logic uses expressions in formal logic [20][3][13] to model the problem domain. There are many kinds of logics such as multi-valued logics, modal logics, temporal logics, higher-order logics etc. Logic programming languages provide rich pattern matching facilities which are generally more powerful than that of the production systems. A logic-based system shares a similar problem to production systems in that the search for solution relies to a certain extent on the structure of the knowledge base. Another disadvantage is that theorem
proving with full unification is computationally expensive and its inability to hold related information together. Some of the problems may be remedied by partitioning the knowledge base, employing meta-knowledge and implementing data structures like frames.

### 4.3.3 Frames

Frames may be viewed as a static data structure, used to represent well understood stereotyped situations. The components of this data structure are called slots. Slots have names and contain information of various kinds (slots can contain frames as well). Unfilled slots may be filled through inheritance. The most common type of inference in a frame system is inheritance. It means that when a frame represents a class of objects, then this class frame can inherit values from its superclass frame. This is particularly suitable for domains with a clear hierarchical classification of objects. Inheritance also avoids inconsistencies arising from duplication of information. The structure of a frame makes it suitable for binding knowledge about a class objects. In addition to these, some frame systems allow procedural attachments (demons) to slots. Thus a frame essentially provides a way to combine both procedural and declarative knowledge about some entity into a single data structure.

### 4.3.4 Semantic Networks

Semantic networks are graph based representations in which nodes represent objects or events in the problem domain and arcs represent relations between them. Attempts have been made to formalise this notation by defining the semantic net concepts by first order predicate logic operations [6][13]. This representation is very similar to frame except that frame allows each node to be a complex data structure consisting of slots.

### 4.3.5 Object Orientation

Object oriented representations are normally combined with other representations described above to provide a more complete description of the problem domain in the context of knowledge representation. Most ESBTs do not provide complete object oriented capabilities and the common ones are inheritance, messages and methods. There are five major concepts in object orientation: objects, classes, messages, methods, and inheritance.

Objects are entities that contain both data and procedures which act on the data. The term object is commonly used to refer to the class of objects and instances of objects. A class defines the data structure of an object (i.e. the data description and procedures). A frame is an implementation for a class. Classes are made up of slots and procedures (methods) which are arranged in hierarchies.
Messages invoke attached procedures to act upon object data. Objects respond to messages they receive by determining the appropriate procedure/s to execute. A message is rather like a specification of ‘what’ to do.

A facility which is fundamental to object oriented systems is inheritance of characteristics. Slots and methods can be inherited from parent classes, if a class is a daughter of two or more classes then it can inherit from both parents (this is known as multiple inheritance).

The advantages of object orientation are,

- data structures allow information to be encoded compactly due to the hierarchy from generality to specificity.
- data values can be inherited rather than duplicated across related objects and thus avoiding any possible contradiction of information.
- procedures can be associated with different objects which improve readability.

4.4 Control Mechanisms

In most real world problems, the search space tends to be combinatorially explosive. This implies the need for some sophisticated search strategies. There exists a reasonably large set of inferencing strategies such as forward chaining, backing chaining, unification etc. The choice of an appropriate strategy, depends on the underlying task and the available data [8].

Backtracking and cut constitute the fundamental control mechanisms for Prolog. Cut is used in order to prevent backtracking up to a certain point, therefore avoiding any waste of time. The application of cut also allows Prolog to emulate other search strategies.

The control mechanism in a production system is the recognise-act-cycle. A recognise-act-cycle consists of two steps,

- match the condition parts of the rules against facts in database.
- if there is more than one rule that matched the condition, conflict resolution strategies are applied to select one to fire.

The recognise-act-cycle repeats until no rule can be fired.

The most common conflict resolution strategies are refraction, recency and specificity. Refraction ensures that once a rule has been fired, it may not be fired again until the elements in the knowledge base that match the condition have been modified. This prevents the system from getting caught in a loop. Recency selects rules whose conditions match the patterns most recently added to the knowledge base. Specificity assumes that a more specific rule is preferable to a general rule. A rule is more specific than another if it has more conditions; this implies that it is more unlikely for a ‘specific’ rule to be matched.
It is found that the way production systems work is equivalent to searching through a state space. The successive states of the knowledge base form the nodes of the state space graph. The production rules define the legal transitions between states and the conflict resolution strategies determine which branch to take.

Apart from using pure classical search strategies, probabilistic reasoning can also be used to formalise the control mechanisms. The well-known approaches are Bayesian statistics and Dempster-Shafer model of evidential reasoning [19]. Bayesian statistics permit the computation of various conditional probabilities, unfortunately, it requires certain independence assumptions which could be difficult to satisfy. Furthermore, it requires a reasonable amount of statistical parameters which is hard to obtain in some applications. The latter point demonstrates the advantage of the Dempster-Shafer theory which does not require a complete probabilistic model. The Dempster-Shafer formalism allows an explicit representation of ignorance and the use of belief function to estimate how close the evidence is to forcing the truth of the hypothesis, rather than the Bayesian approach which directly estimates the veracity of the hypothesis.

Using these reasoning formalisms, we should be able to order the search and manipulate various processes opportunistically.

4.5 AI Programming Paradigms and Languages

This section provides an overview of various commercially available AI tools; we also attempt to highlight the features associated with each tool. Note that we have rejected the use of expert system shells since it is inflexible both in knowledge representation and control. Note that there is no global definition of what a shell is. Traditionally, a shell is an abstraction of an expert system. Nowadays, in order to facilitate the development of expert systems, many ESBTs emerge and since they are designed for rapid prototyping some people may regard them as shells as well. Nevertheless, in this chapter we stick to the former definition.

CLIPS [18] was developed by NASA in order to overcome the problems they encountered using Lisp-based tools. The main problems was the poor portability and integration with other languages. As the name suggests, CLIPS is written in C and was designed using ART's rule system as a model and therefore inherited the Lisp-like rule syntax originally developed for ART. CLIPS supports a powerful pattern matching facilities for specifying rule conditions. The pattern matching facilities operate on both single and multi-field sequences composed of strings and numbers. Furthermore, conditions can also be formulated in such a way that a rule is instantiated only if a pattern cannot be matched by any fact in the knowledge base. This means that reasoning can be based on the absence of information as well as its presence. In addition, CLIPS also provides procedural programming constructs (if ... then ... else, while) on the right hand sides of
the rules. Control strategy of CLIPS is based on forward chaining that implements the classical
recognise-act-cycle. The nature of the recognise-act-cycle means that pattern matching between
the left hand sides of rules (conditions) and the knowledge base are carried out frequently. This is
inefficient since the left hand sides often share conditions. Another source of efficiency arises
from matching patterns to the whole knowledge base since only a small portion of it is modified
during each cycle. The former and latter sources of inefficiency are called within-cycle and
between-cycle interaction. CLIPS reduces a significant amount of matching operations through
its support of the Rete matching algorithm. CLIPS does not provide backward-chaining,
inheritance or procedural attachments. Its main advantage, apart from portability, is the execution
speed.

ART-IM [18] is a C based tool developed using CLIPS as a base and it also supports frame based
representation called schemata and object oriented programming paradigm. Schemata can appear
on the lefthand sides of the rules and are constructed by using either a defschema statement
which creates schemata when execution is initiated or by an assert action on the right-hand side
of a rule that builds schemata during execution. ART-IM supports single inheritance scheme in
which both values and functions are inherited via is-a and instance-of relations between
schemata. Procedural knowledge is supported by attaching functions to schemata. As one would
expect, there is a great deal of similarity between CLIPS and ART-IM; the primary difference
being the logic dependency mechanism provided by ART-IM. This means that if a fact is
specified as being logically dependent upon another data object and when this data is retracted
from the knowledge base, the logically dependent fact is automatically retracted. This effectively
provides a means for non-monotonic reasoning which would be useful for uncertain data. Both
the debugging aids and the level of integration with external systems are comparable to that of
CLIPS.

KEE [8][20][3] provides a well designed representation language based on units which are
similar to frames. Reasoning in KEE is based on rules which are represented as units and can be
used in either forward or backward chaining. Non-monotonic reasoning is also possible using a
Truth Maintenance System (TMS). A reasoning process is usually initialised by a call to the
knowledge base Assert and Query Language supplied by the package. KEE supports object
oriented programming paradigm with features such as inheritance, bindings of object data with
procedures associated with that object and message passing between objects. Procedural
knowledge can also be represented by means of active-values, values to which are attached
functions that are triggered when the value is accessed. In addition to these, KEE provides a very
user-friendly interface for debugging and allows access to all features of KEE through pop-up
menus.
Knowledge Craft \cite{8}\cite{20}\cite{3} has a very complete representation of frames called schemata, which can represent objects, actions, situations, facts and events. Inheritance path can be defined by means of user defined relations. Knowledge Craft also supports meta-schemata, meta-slots and meta-values (which are themselves user-defined schemata) which can be used to encode meta-knowledge. For instance, meta-slots attached to a particular slot make it possible to specify restrictions on the possible values of the slot, inheritance specification and demons. Knowledge Craft like KEE supports both object oriented programming and access oriented programming. Reasoning in Knowledge Craft can be achieved via CRL-OPS rules or Prolog predicates. CRL-OPS combines the representational power of CRL’s schemata and the reasoning power of OPS5’s rules. CRL-OPS uses Means End Analysis (MEA) which is based upon refraction, recency and specificity to determine which rule to fire in a cycle. CRL-PROLOG provides a mechanism for deductive queries and goal-directed reasoning. It is also possible to perform hypothetical reasoning using the context mechanism provided by the package. Hypothetical reasoning allows many virtual copies of the knowledge base to be kept, reasoning about hypothetical data and, testing and modelling of different situations.

In comparison to KEE and Knowledge Craft, ART \cite{8}\cite{20}\cite{3} has a less complete schema representation (neither meta-knowledge nor demons can be attached to slots). Reasoning can be done via forward or backward chaining (backward chaining are emulated using goals, patterns and viewpoints). ART also supports hypothetical reasoning using viewpoints. It provides a comparable interface to Knowledge Craft.

4.6 Framework of Knowledge Based Vision Systems

Computer vision encompasses areas such as signal processing, pattern recognition and artificial intelligence. Whilst knowledge based interpretation of visual scenes (also known as image understanding) refers to similar disciplines it emphasises knowledge representation and reasoning methods.

Image understanding systems (the term image understanding systems and knowledge based vision systems are used interchangeably) are concerned with the construction of symbolic descriptions of the scene depicted in images. Image understanding systems (IUSs) seek to analyse and interpret scenes in terms of world models supplied to IUSs as knowledge about the world. In other words, IUSs establish correspondence between models supplied and image
structures. Objects in the scene (see fig. 4.2) are matched to image features such as points, lines and other groupings.

IUSs usually consider scenes composed of mutually related objects of different types. This is in contrast to recognition systems which aim to extract target objects from scenes. As the objective of recognition systems is to identify target objects, the knowledge used tends to be intrinsic properties of target objects and knowledge about the object world is rarely used. Both image understanding systems and recognition systems discriminate between information about the object world and that about the image. They differ mainly in the amount of world knowledge used. The task of IUSs is to establish the mapping between the object model and the image features in the scene.

As mentioned earlier, the task of an IUS is to bridge the information described by the world model normally in three-dimensional form and that of the image in two-dimensional form by computation and reasoning. Performance of earlier IUSs was found to be inadequate and this was attributed to the use of a single step in mapping the two levels of information. In modern IUSs, the information is organised into various levels. Processes associated at each level are used to
transform information at each level to another. The figure shown in fig. 4.3 depicts the architecture of a generic image understanding system. The following paragraphs describe the generic system depicted in fig. 4.3.

Image is captured by various types of sensors such as CCD cameras, infra-red sensors, range sensors etc. Each type of sensor detects a certain type of information such as brightness, colour, depth and so on. Here, we consider images captured by sensors like CCD cameras.

In general, an input image in its raw form is processed to produce image features such as points, lines, regions etc. These features are grouped into higher level image features based on different kinds of geometric relationships between them. These relationships include proximity, for instance, which links together disjuncted lines to form continuous line segments. We term the resulting information after feature extraction and grouping intermediate level structures. Major sources of rules or geometric relationships used for grouping are based on the so-called Gestalt laws. For further information of Gestalt laws and intermediate level structures please refer to chapter 2. Haralick [5] noted that different sensors are used to capture images and they all possess different characteristics and he approximated the spatial distribution of pixel values in a local area by a polynomial and defined various image features based on the shape of the function.
Marr [14][15][16] proposed several grouping methods based on the edge based description known as the primal sketch. Lowe [9][10][11][12] emphasised the significance of perceptual organisation for feature grouping.

An image is simply a projection of a 3D scene onto a film. In order to establish a correspondence between the scene objects and the image features the IUSs need to back-project the information from the image to the scene domain. The information needed for the back-projection is the physical properties and the conditions under which the image is taken.

The matching process comprises two subprocesses, the process of model invocation and the process of establishing correspondence. Information such as feature groupings, scene features and the description of the scene can be used to select or invoke an appropriate model. Through the transformation process, correspondence between the object model and image features can be established. One can also match image features and object model directly.

The matching process described so far is a bottom-up process. That is, feature extraction and groupings are used to construct a description of the image, this information is then fed to the model invocation process to instantiate an appropriate model for matching. In a top-down process, the instantiated model ‘knows’ what image features are expected and directs the feature extraction and the grouping process in an attempt to locate missing parts.

IUSs use various types of knowledge, scene domain knowledge, image domain knowledge and knowledge for mapping between image and scene. Scene domain knowledge corresponds to inter-relationships between world objects and intrinsic properties of objects. These include spatial relationships between objects in the problem domain and their constituent parts, physical sizes of the parts etc. Image domain knowledge refers to information that is used to facilitate extraction of image features. For instance, knowledge used for grouping of image features to build up a structural description and colours of objects of interest. Knowledge used to establish correspondence between image features and world objects is, for instance, focal length of the camera, resolution, aspect ratio etc.

Apart from these types of knowledge, control knowledge used for the guidance of reasoning processes also plays an important part of IUSs. It determines the type of objects or features to detect first, region of interest and so on. Due to the large number of world objects in a world scene and the number of common constituent image features that make up individual objects, the combinatorics present in IUSs is enormous. To have a realistic chance of handling this search space, control knowledge incorporates heuristics to prune the problem space.

In this thesis we do not pursue the use of BBN [28][29][30] as the proposed system in Ch5 does not perform information integration.
4.6.1 Problems in Image Understanding

In order to achieve the objective of image interpretation, IUSs, in general, have to tackle two basic tasks. One is to reliably extract image features and construct an intermediate level representations of the image. Two is to establish a mapping between image features (intermediate level representations) and object models.

The construction of intermediate level representations involves the utilisation of image processing techniques such as edge detection, Hough transform, texture analysis etc, to extract primitive image features like lines and curves. These lines and curves are then grouped based on rules namely, continuity, proximity etc. The grouping rules proposed by Marr [14][15][16], Haralick [5] and Lowe [9][10][11][12] have been briefly mentioned in the previous section.

In the interpretation stage, processing techniques both numeric and symbolic are exploited to realise versatile IUSs. These techniques include pattern recognition, graph matching, symbolic reasoning etc. Although significant research effort is made in these areas, there remain issues in these areas which need to be addressed and are widely regarded as non-trivial.

Numerous image processing techniques exist to segment images. That is, to extract image features like lines, curves, regions etc. Whilst they work well in controlled environments, none of them can reliably extract these features in the presence of surface markings, shadows in natural scenes and so forth. As a result, some features facilitate the grouping and interpretation processes and others lead to confusion when information is promoted from lower to higher level. Another source of error stems from the grouping process. The grouping methods developed so far are based mostly on heuristics measures and are generally inappropriate for images where the signal to noise ratio is low.

The problem of imperfect segmentation is inevitable. It is almost impossible to produce ‘perfect’ segmentation in the first instance as this would require the establishment of global context. However, the framework of hypothesis testing adopted in chapter 5 is found to be effective in combating erroneous segmentation. As to erroneous information introduced by improper grouping of image features we recommend the use of formal computational methods which produce results that degrade gracefully in the presence of noise. For details of formal computational approach to feature grouping please refer to chapter 2.
4.6.2 Reasoning Based on Geometric Information and Functionality

The use of geometric information and functionality for reasoning is a direct result of knowledge derived from the problem domain. This type of knowledge helps to guide control processes as well as accepting and rejecting hypotheses. For instance, in order to search for glasses the control processor would dispatch commands to look for cylinders and verify the hypothesis by functionality. As the functionality of a glass is to act as a container for fluid, the control processor might reject any hypothesis which corresponds to any cylinder which is shallow. Another example in which functions are represented by the geometric properties and spatial relations is the search for runways. In this case, the control processor would look for long, thin structures as this geometric property is necessary for aeroplanes to take off.

4.6.3 Reasoning with Incomplete Information

The problem of image understanding is inherently under-constrained with obviously limited input of information (apart from systems that operate in controlled environments) and large number of interpretations of image scene. This is further compounded by erroneous information derived from the segmentation process. This implies the need for IUSs to reason and handle incomplete information. The approaches used to handle incomplete information are summarised as follows:-

- Association of a probability with each piece of information is used to measure the likelihood of a correct match. It follows that a correct interpretation of the scene corresponds to the interpretation with the highest probability.
- Incorporation of rich knowledge sources to guide matching processes and to generate hypotheses is used to complement the lack of information.
- Matching which is based on the optimisation of evaluation function which measures the degree of match between object models and image features.

The system presented in chapter 5 tackles the problem of incomplete information by the use of feature grouping process as well as the hypothesise and test framework.

4.6.4 Spatial relationship

Spatial relationships such as adjacency, intersection, left-of, right-of, above etc. have often been used to describe structural relationships between image features and object models. These relationships are often found to be insufficient for the purpose of characterising spatial relations between entities. Spatial relationships were popular and were often used to transform numerical
information into symbolic ones. ACRONYM, which is described in more detail in later section, used geometric relations between generalised cylinders to represent the transformation matrix which transforms one coordinate system into another.

### 4.6.5 Control Reasoning

Control reasoning process is an integral part of a versatile IUS. Control knowledge facilitates flexible control strategies and this is an important factor in ensuring that the search space is manageable given that the task of image understanding is intrinsically under-constrained.

Broadly speaking, IUSs can approach the interpretation task from two directions. One is the bottom-up approach and the other is the data-driven approach. In the bottom-up approach, data is organised into image features and intermediate representation via the process of feature extraction and grouping. Subsequently, the matching process can then establish correspondence between the structural description of the image and the world model. In top-down approach, the model of the target is determined and image processing procedures and grouping process are then invoked in an attempt to verify the existence of the constituent parts of the target and hence the target itself. In order to realise an effective top-down strategy one needs to ensure that the appropriate model is invoked to guide the top-down process as well as the ability to focus on the part of the image where the targets exist. These imply the need to incorporate the focus of attention mechanisms.

It is generally accepted that the top-down reasoning and the bottom-up reasoning should complement each other. The bottom-up approach on its own is not flexible enough to adapt to the scene structure and is therefore difficult to extract meaningful structure without cooperation from the top-down process. Top-down analysis, on the other hand requires control knowledge to drive the focus of attention process. A sensible control reasoning in IUSs should organise raw image data into intermediate level structures such as rectangles etc and then invoke the appropriate model based on the result of the bottom-up process.

The problem of information integration can be solved through the use of Bayesian Belief Network [28][29][30] (BBN). BBN is a directed acyclic graph whose nodes represent propositional variables and whose arcs represent causal relationships. The propagation of beliefs through a BBN allows evidence to be interpreted through updating of beliefs from both top-down and bottom-up processes.

### 4.7 Review of Knowledge Based Vision Systems

This section provides a survey of knowledge-based vision systems. The problems encountered by these vision systems and the rationale behind their approaches are reviewed.
SCHEMA [1] and SPAM [17] are both built upon the schema based representation. The primary design philosophy for SCHEMA is that both knowledge and computation should be partitioned. Each class of object and object parts has a corresponding schema which stores all object and control knowledge specific to that class. These schema instances run independent concurrent processes, communicating through a global data structure called a blackboard and the 'solution' is built incrementally in the blackboard. This allows the control mechanism to be changed opportunistically during each stage of the reasoning process. On the other hand, McKeon et al experienced unforeseen interaction between rules and difficulty in achieving a desirable control flow. The main problem can be traced back to the lack of explicit description of the control knowledge. The different experiences of these systems highlight the importance of control reasoning.

SCERPO [10] recognises three-dimensional objects from unknown viewpoints using three mechanisms to bridge the gap between three-dimensional and two-dimensional objects. First, a process of perceptual organisation is used to build groupings and structures which are viewpoint invariant from a wide range of viewpoints. This criterion is particularly important as this allows objects to be recognised from unknown viewpoints. Second, a probabilistic ranking method is employed to prune the search tree during model matching. Finally, a spatial correspondence process matches projections of three-dimensional models with that of the image features by solving for the unknown viewpoint.

Prior to the recognition process all possible groupings and the model features that could give rise to them are maintained in a list. These groupings are used for model invocation. Each grouping in the image is matched against the models which are likely to give rise to the image structure. As there are potentially a large number of models which match any one structure, the models are prioritised so that the most promising model are verified first. More complex structures are regarded as more promising as they have fewer matches with the models and are unlikely to arise by chance. Each match provides a set of known correspondences between the three-dimensional model and the image structure. Once the initial matches has been used to solve for the viewpoint, this prediction can then be used to predict the locations of other model features in the image and hence extending the match. The viewpoint and the model parameters are updated after each match.

ACRONYM [23] is a general purpose vision system that employs both top-down as well as bottom-up reasoning. Frames are used to represent structured knowledge about an object. Frame has been widely used in many IUSs including MAPSEE [24] and VISIONS [25]. Geometric descriptions of objects and its constituent parts are modelled as generalised cylinders. A generalised cylinder is defined by (1) a spine, (2) a cross section and (3) a sweeping rule to
transform the cross section as it is moved along the spline. World model in ACRONYM is organised into two hierarchies, the composition and the generalisation/specialisation hierarchies based on the PART-OF and the A-KIND-OF relations respectively. Generic model of aeroplanes as well as the spatial relationships between their sub-parts are captured in the frames. Specialisation is achieved by associating a set of constraints with the generic model. Like the SCERPO [10] system, ACRONYM also exploit the viewpoint invariant properties of image features to prune the search space. Observed features that match the predicted features help to constrain the interpretation of the three-dimensional model. Viewpoint and the model parameters are then solved. Internal model parameters are used to verify or discriminate image features.

CLIPS (C Language Integrated Production System) [26][27] has been adopted as the choice of tool for the implementation of the prototype vision system to be described in Chapter 5. CLIPS has been selected for its flexible implementation and simple control structure. As CLIPS is implemented in the C language it facilitates simple and efficient interfacing with low level image processing routines. It is also a highly portable tool given the popularity of the C programming language.

4.8 Discussion

Most knowledge-based vision systems exploit knowledge about spatial constraints, geometrical constraints and relationships between image features. This kind of knowledge can easily be modelled by frames and semantic networks, and therefore, not surprisingly, these representations are adopted by most researchers. There are several factors which can be used as guidelines for deciding an appropriate AI tools for vision; they are the knowledge representation formalism, reasoning and control mechanisms, ease of modification and extension, execution speed and the debugging aids supported by the tool.

In terms of knowledge representation, it should be noted that although different representation schemes are epistemologically equivalent, some are more suitable for modelling a certain domain and task than others. An inappropriate choice of representation would lead to an inefficient runtime performance, increase the time required for the implementation of the system and can also be difficult to make relationships explicit.

Reasoning and control are two important aspects in knowledge-based systems. They are virtually inseparable in the sense that control may be viewed as the pruning of the state space by application of reasoning. In higher level vision there are a huge amount of data or tokens to be processed, it is vital to be able to reason, for instance, the sequence of data manipulations, the use of contextual information etc. The so-called reasoning needs to be emulated using sophisticated and efficient access, update and retrieval of data and facts in knowledge base. It is also an
advantage to be able to make explicit the control knowledge in the system (i.e. the use of meta-
knowledge). Furthermore, due to the nature of data in vision, it would be beneficial to perform
hypothetical reasoning.

The ease of modification and extension of the system are, to a certain extent, related to the
modularity of the code. This favours the kind of representation which holds together information
relevant to a certain entity in a module.

The goal is to use the prototype system in Ch5 as a means to develop an efficient algorithm for
intermediate-level vision. Nevertheless, if one wants to optimise for speed the code should be
translated to some conventional programming languages.

The debugging aids provided by the tool can help to identify any potential problems due to
programming errors, knowledge acquisition etc. The middle to high range AI tools available
nowadays are normally well-equipped with user interface and debugging aids. Their
functionalities are comparable to each other. However, these AI tools are normally difficult to
migrate from one architecture to another.

In [4], Goodman et al suggest that the programming languages that come closest to providing
facilities for vision applications fall under the category of object oriented database languages.
They found the following properties are necessary for programming vision tasks,

- **persistent data objects should appear to the programmer no differently than do transient
  objects.**
- **the programmer should be able to build and specify behaviour for highly connected
  networks of objects. Additionally, there is also a need for type constructors such as sets,
  sequences, relations and arrays to impose restrictions on these arbitrary graphs of
  objects.**
- **an object oriented construct for data modelling with semantics such as set and relation.**

The first point can be seen in a situation such as model matching. During the off-line stage the
stored models and its associated data structures will be constructed. At the on-line stage the
various image features are extracted for the construction of data structures to describe the object
and subsequently the correspondence between the data structures obtained from the on and off
line stages can be found. Clearly, the data structures constructed in the off-line stage need to exist
between processes, whereas data structures computed from the on-line stage need only to persist
during the matching process. The second and third points basically emphasise the need of rich
modelling constructs for vision applications.
In the light of previous sections, we decided that expert system shells, primitive production systems and blackboard systems are not initially suitable candidates for our application in intermediate and high level vision. Expert system shells are designed for rapid implementation of the system, it is assumed that the knowledge representation and the control mechanism suit the application at hand.

Production systems are suitable for tasks that map naturally to state space search. It is, in general, difficult to control the line of reasoning and extend or modify the system. It is due to interactions between rules and lack of indexing mechanism into the appropriate rules for focus of attention. When we say ‘primitive’, we mean a pure production system that does not support features such as inheritance and coding of procedural constructs.

The architecture of the blackboard systems allows the expert system designers to change the control mechanisms (ie. which module of knowledge to apply is determined dynamically) at every stage of the reasoning process. In the simplest case, a blackboard system consists of several logically independent knowledge sources, each communicates with others through the blackboard. Note that the locus of control can be in the blackboard, knowledge sources, in a separate module or in a combination of three. This clearly provides a wider choice of control. This form of architecture is particularly suitable for problems that inherently consist of a few application independent hierarchies. For instance, in the SCHEMA system the hierarchies span from low level edge detection to high level model matching. However, explicit reasoning about control and task scheduling incur expensive computational cost. Our application domain (i.e. intermediate level vision) does not possess several independent knowledge sources and therefore probably not worth the overhead of the computational cost.

In the light of the problem in hand and the pros and cons of various AI tools available we find CLIPS to be most appropriate. Apart from the wide range of facilities available, routines written in C can be integrated into CLIPS with relative ease. Portability of the resulting system also plays an important role in reaching this decision.

4.9 Conclusions

In this chapter we briefly look at several areas relevant to Expert System Building Tool (ESBT). These include AI programming languages, knowledge representation and control mechanisms. An extensive review of available AI tools as well as the framework of generic image understanding systems are presented. Based on the reviews presented in this chapter we decide to use CLIPS to implement our system presented in chapter 5.
4.10 References


Chapter 5

Application of Intermediate-Level Strategies In Aerial Imagery

5.1 Introduction

Identification of linear boundary segments is known to be an important intermediate-level approach to recognising man-made objects in natural scenes. Various schemes using straight lines, or key groupings of lines, have been devised and provide reasonable segmentation when boundary information is complete or when a detailed model of objects of interest allows a model-driven approach. The segmentation problem is particularly acute however in low contrast images where data is imperfect and incomplete and little model information is available. Uncertainties can arise from a variety of sources including noise, scene clutter, extraneous data, as well as from low-level extraction routines themselves. While the art of straight line extraction has been refined for noisy images [1], there remains the task of deciding how to connect broken segments and of dealing with missing lines [2]. Handling fragmentation of this kind in the context of performing specific recognition tasks is the motivation for developing the framework proposed in this chapter.
5.2 Summary of Requirements of Framework

The primary objective is to develop a methodology capable of handling uncertainties in segmentation of images composed of imperfect and incomplete data, and robust enough to deal with the situation when uncertainty also exists in the model. A secondary objective is to investigate ways in which chosen representations can be incorporated into a global scheme for scene segmentation. To achieve these aims it has also been necessary to decide upon appropriate feature primitives and associated extraction techniques, and to characterise the type of imagery suited to the proposed methodology. In this work we show that line segments are at the bottom level of a multi-level hierarchical feature space, and a framework is proposed for pursuing the most promising among alternative hypotheses to the highest level in the hierarchy where closed polygons or polygon structures are hypothesised.

Since robustness is a prime consideration, we make the assumption that segmentation techniques used in combination can potentially provide better performance than when used alone; this has been referred to as the cooperative methods paradigm [3]. There have been many attempts to exploit this observation, but no general theory exists on how different techniques can best be integrated. The work described in this chapter proposes a framework to perform the integration for various pre-defined tasks associated with identifying runways/taxiways in aerial imagery. Along with associated robust feature extraction techniques, the framework represents a unique attempt to combine and integrate relevant problem-solving strategies in intermediate-level vision. The original motivation of the work was to investigate the use of Artificial Intelligence techniques in intermediate-level vision.

There have been many aerial Image Understanding Systems that have incorporated AI techniques to represent knowledge about scenes and to realise flexible control structures [4]. Representation and search are two central issues in AI, and controlling complexity of the search process is an important consideration when demonstrating the generality of a proposed scheme. In [5], generality, representation and control are identified as key elements by which Image Understanding Systems can be compared, and three questions are asked: 'What constitutes useful features and constraints?' 'How can features and constraints be reliably extracted?' and 'How can features and constraints be used for recognition?' In this chapter we have sought to address these questions in the context of intermediate-level extraction of runways/taxiways in aerial imagery, while paying particular attention to the underlying noise issues and the resulting implications to search complexity.

5.2 Summary of Requirements of Framework

The goal of the work described in this chapter relates to reasoning in intermediate level vision. It is customary to divide the processing stages for scene analysis and image interpretation into three levels (low, intermediate and high) though it is difficult to define the demarcation lines precisely.
Low level processing includes the image operations and calculations applied at pixel level, usually general purpose, whereas high level processing tends to be more domain-specific and is concerned with manipulations of symbolic entities in the scene. The conversion from image to symbol, which is under investigation here, occurs at intermediate-level but in general may require calls to other levels, possibly through low/intermediate and intermediate/high interfaces. Exactly what is involved in this conversion process is far from clear and it is necessary to address this issue before designing a method for robust detection.

We recognise the importance of using high level constraints for the purpose of recognising objects in intermediate-level vision. However, the goal is to extract as much information as is possible using purely intermediate-level features. In particular, the test images for this work are noisy and contain very little structure with barely discernible man-made objects so that there is no opportunity to use high level constraints among scene entities for recognition. Thus the requirement is that hypotheses be verified using constraints derived from properties of objects themselves rather than from structural constraints between objects in the scene. The output of the system is a set of hypothesised instances, which could subsequently be used to direct further verifications in cooperation with high and low level processing.

Pragmatic concerns for vision systems often require tuning of parameters to the application domain, and this can compromise the generality of the resulting system. It is therefore necessary to be careful about using informal heuristic criteria and to design the system so as to make clear what parameters are involved and to provide a structure that enables their effects to be clearly visible. In order to create a non-specific methodology it is necessary to minimise dependence on ad hoc heuristics [6] by clearly defining the implications and predicting consequences when heuristic criteria are unsuccessful [7]. Most practical systems will contain some heuristics, but clarity of operation is greatly improved if they can be organised within a uniform representation.

Search complexity is often associated with the high level vision problem of applying constraints effectively, and in the domain of aerial imagery systems based on Rules, Blackboards, Truth Maintenance, Relaxation Labelling, Constraint Satisfaction have been designed to search for interpretations [4]. In contrast, the focus here is on complexity due to the need for handling missing data as a result of high levels of noise in the image, and for which a conventional technique by itself is not appropriate (e.g. Hough Transform, and for an analysis of its search complexity see [8]). In order to limit the large number of hypotheses that can be generated when missing information is hypothesised, it is necessary to provide a way of both constraining and prioritising the search for verified hypotheses.
A further requirement is that the system be robust in the sense that images can be handled when only qualitative model information is available. Runways are modelled as rectangles under perspective projection, but in general there is no knowledge of camera geometry or viewing angle. This allows for a large variety of 2D structures to be detected, and the appearance can be dramatically different depending on the distance of the camera from the scene.

In terms of the detection task that the system is being asked to perform, many requests could conceivably be made, with or without an explicit efficiency constraint, for example:

- Find the best instances.
- Find the n best instances.
- Find a good instance quickly.
- Find all instances.

We make the assumption that quality of solution encompasses two different aspects:

(i). how likely the hypothesised instance is a true instance.
(ii). how accurate is the segmentation boundaries (i.e. how close are the hypothesised boundaries to the ground truth).

If the system is to respond to different requests of the kind above, then it is our assumption that strategies with complementary properties and advantages need to be integrated. Further, the system needs to operate in different modes; for example, to guarantee finding all solutions, in principle it needs to be able to perform an exhaustive search, even if in practice that might not always be feasible. To handle these requirements we have chosen to define a set of Priority Measures for each hypothesised feature in a hierarchical feature space, and to provide thresholds on these measures which can be varied to prune the search for verified hypotheses. If the thresholds are sufficiently relaxed, an exhaustive search of all possible hypotheses (no matter how unlikely) is performed. At the other extreme, if the thresholds are sufficiently tightened, only the most likely hypotheses are pursued, and the quality of the solutions (if indeed any are found) is dependent on the reliability of the Priority Measures. In this chapter we seek to address the first aspect outlined above. These measures are discussed in section 5.4.4 and are intended to prune the search space. While the correspondence of the hypothesised boundaries to the ground truth is a valid measure for the quality of the solution, this is beyond the scope of this chapter.
5.3 Previous Work

In this section we survey a number of vision systems developed for aerial imagery understanding. A summary is provided in chapter 6 which highlights the differences and contributions between our system and those reviewed.

Nazif and Levine [16][21] devised an image segmentation system using a production system approach which encompasses 3 levels of rules - strategy rules, control rules and knowledge rules. The strategy rules facilitate the selection of appropriate control strategy. This selection process, based on the data and the set of control rules, dynamically adjusts the priorities of metarules and focus of attention rules which in turn drives the knowledge rules at the bottom level. The focus of attention rules replace the current set of image features (lines, regions etc.) with another set in order to direct attention to a more worthwhile part of an image. Knowledge rules are domain independent rules used to encode knowledge for segmentation of regions, areas, lines and other image features that are based upon Gestalt properties (i.e. uniformity, continuity etc.). The rules are the classical condition-action pairs. Conditions that can be tested are logical comparisons on numerical and non-numerical variables as well as logical evaluations of numerical and non-numerical variables. At the knowledge rules level, condition-action pairs are categorised into groups such as regions, lines and areas. Each set of rules operates on the corresponding image feature and this organisation facilitates the strategic rules and the focus of attention rules at the higher levels. Performance analysis suggests that the rule-based approach produces superior segmentation when comparing with histogram splitting and split-and-merge algorithms at the expense of higher computational load. Finally, it is reported that this system architecture promotes separability between knowledge and control and permits easy modification of rules.

Niblack and Petkovic [17] evaluate the usefulness of knowledge based approach on two image processing applications - printed circuit boards (P.C.B.) inspection and ice floe satellite image segmentation. Scene analysis and interpretation can broadly be categorised into three levels - low, mid and high. The low level involves processing such as non-maximal suppression, convolution, thresholding etc. Mid level is concerned with the creation of image features from images. These include the creation of image features like edgelists and homogenous regions, and the computation of their corresponding set of descriptors. Physical attributes such as length, coordinates, area etc. as well as spatial relationships (e.g. in front of, is part of etc.) are examples of descriptors. At the high level, entities (image features) generated at lower levels are processed and manipulated to derive either a symbolic description or interpretation of the input image. The authors point out that rule-based systems do not work well for low level image manipulation. The reasons are manyfold:
5.3. Previous Work

- **Primitives and rules that are needed for reasoning are lacking.** Also there are no reliable definitions for features such as edge regions, texture measures, shape descriptor, etc.

- **Rules applied at the low and mid level are error prone and suffer from immature decision making.** Specific decisions are made on local information without taking into account the global context.

- **Rule-based systems in general suffer from an overuse of thresholds and consequently lead to systems which have numerous thresholds that need to be adjusted.** Thresholds in image analysis are normally static and misadjusted thresholds could cause incorrect linking of edges, missing of edgelists, etc.

- **Both the symbolic representation as well as the image data need to be updated to maintain consistency in the systems.** This imposes a significant load on book-keeping as far as the maintenance and development of the systems are concerned.

The authors concluded that procedural programming is always more robust than rule-based methods. Application of rules at the low level are found to focus too intensely on local information. To combat the effect of noise leads to an explosion of heuristics and thresholds. The resulting systems require large programming effort for maintenance and development as well as being susceptible to noise. Finally, it is pointed out that applications for scene interpretation and hypothesis initialisation and verification tend to benefit. Inefficiencies of rule-based systems due to the use of exhaustive search [17] when processing is done at the symbolic level are also highlighted.

The roles of Artificial Intelligence (AI) techniques in the field of aerial imagery understanding was discussed in [4]. This chapter uses a few systems to illustrate the pros and cons of AI techniques. These include knowledge representation and control strategy.

It was discovered/that/blackboard architecture is useful for integrating various object detection modules. This architecture together with a flexible control mechanism provides a framework for a system capable of analysing fairly complex aerial photographs. However, its capability is limited:

(i) its lack of separability between knowledge and image analysis procedures limits its scope for system modification.

(ii) 3D information was not utilised to establish correspondence between 3D models and scene data.

In so far as knowledge representation is concerned, frame is a convenient scheme for the storage of attributes of the object, its relations to others and procedures for the computation of object properties. SIGMA and LLVE use frames to represent knowledge for hypothesis generation and
for image processing. ACRONYM, on the other hand, uses frames to represent 3D objects for the modelling of aircrafts. The reasoning processes involved in the aforementioned systems can be categorised into the following levels

(i). structural and spatial relations reasoning.
(ii). reasoning between models and image features.
(iii). reasoning about image segmentation.

It is argued that these levels of reasoning correspond to different tasks and they should be implemented as different modules. As a consequence, one can select the appropriate knowledge representation and control structure to match each level of reasoning. Finally, Matsuyama discusses the use of knowledge-based system in image processing system. It is believed that with the right set of rules a fairly versatile remote sensing system can be developed. It is also suggested that geographic information systems and remote sensing systems should complement each other.

Harlow et. al. [30] developed a hierarchical vision system for the analysis of high resolution aerial scenes. The authors identify the need for operators that are capable of characterising texture. To achieve this the operators should possess the ability to detect perceptual similarity such as periodicity, continuity, etc. (properties described by Gestalt laws). As mentioned earlier, the system is hierarchical and each level corresponds to a different scale of detail. Attached to each node is a generic object and its corresponding frame for the characterisation of the object. Each frame consists of the following information:-

(i). analysis goal (e.g. polygon labelling).
(ii). subobjects.
(iii). search information.
(iv). belief values for the combination of evidence.
(v). operator for the object.
5.3. Previous Work

The system proceeds from the root of the hierarchy (see fig. 5.1) and determines whether the object to be detected is one of the subobjects. For example, if the target is a housing area the object operator will assign belief values to the subobjects. The strategy is to pursue the object with the highest belief and to label the region of interest (the subobjects being the labels) until no further labelling can be performed. Guided by the belief values the analysis proceeds to the second level. This leads to a coarse to fine labelling of the image under analysis.

McKeon et al. [32] developed a system called SPAM which incorporates a map database, image processing tools and a rule-based system for the control of low level image processing as well as the interpretation of results. The system uses explicit camera models to facilitate the use of viewpoint independent models for the computation of size, distance, relative and absolute position of objects. The system interprets results by characterising and collating regions into consistent interpretations based on spatial and structural knowledge. Subsequently, McKeon [26] extends the SPAM system by automating the process of interactive knowledge acquisition for both scene primitives and spatial constraints in the domain of aerial imagery. This is achieved.
through the use of an interactive knowledge acquisition interface and a translator which translates schema-based insertions into production rules as well as a performance analysis tool. These components improve the manageability and flexibility of the overall system.

The goal of the SCHEMA system is to interpret static, colour images by locating significant objects in the scene and identifying relevant object relationships. The SCHEMA system exploits coarse-grained parallelism in the interpretation process described below. The primary design philosophy is that both knowledge and computation should be partitioned. Each class of object and object parts has a corresponding schema which stores all object and control knowledge specific to that class. A schema instance -- an executable copy of the schema which runs as a separate process with its own state, is invoked for each object instance hypothesised to be in the scene. These schema instances run independent concurrent processes, communicating through a global data structure called a blackboard and the 'solution' is built incrementally in the blackboard. Each schema instance directs the application of general-purpose knowledge sources to gather support for its object hypothesis. The knowledge sources are classified into three areas namely, low-level, intermediate-level and high-level. That is each individual knowledge source holds information related to its level of abstraction. The reason for the use of the blackboard architecture is its flexible control mechanism. This allows the control mechanism to be changed opportunistically during each stage of the reasoning process. The weakness of the system is that it fails to apply 'deep' knowledge and instead it uses, to a certain extent, some situation-dependent knowledge.

Shufelt and McKeown [3] assumed that no single detection method is capable of correctly distinguishing between man-made objects and naturally occurring terrain features in aerial imagery. A co-operating methods paradigm is proposed for information fusion to improve overall system performance. The goal of the research is to develop a vision system that integrates results from four individual established building detection schemes to provide an accurate and robust interpretation of the underlying three dimensional scene. Building hypotheses take the form of two dimensional polygonal boundary descriptions and the methodology, referred to as scan-conversion, relies on accumulating votes from each hypothesis for each pixel in the image. The 'accumulator' image then has a count for each pixel representing the number of hypotheses voting for it, and segmentation is performed by connected component region extraction techniques. To quantify improvement in performance, an evaluation method using a hand-crafted ground truth segmentation was developed. Pixel counts are divided into four categories according to the four true/false and positive/negative combinations; a pixel branching factor,
defined as number of ground-truth background pixels incorrectly predicted (false positives) divided by number of ground-truth building pixels correctly predicted, represents the degree of overclassification of background as building pixels.

Results show that the fused information out-performs individual schemes in all cases in terms of building detection percentages. However pixel branching factors are higher because of increased number of false positives and experiments show that thresholding alone may not improve this without reducing detection rates. Similar results were obtained by applying the method to stereo pairs, and the authors claim that it should be equally applicable to sequences of images. The overall conclusion was that substantial performance improvement may be gained from information fusion in the domain of aerial image analysis.

The problem of information integration can be solved through the use of Bayesian Belief Network [27][28][29] (BBN). BBN is a directed acyclic graph whose nodes represent propositional variables and whose arcs represent causal relationships. The propagation of beliefs through a BBN allows evidence to be interpreted through updating of beliefs from both top-down and bottom-up processes.

**5.4 Proposed Framework and Methodology**

**5.4.1 Hypothesise and Test Paradigm**

AI researchers have developed various ways of handling uncertainty reasoning [9]. Some remain within conventional probability theory, but use computational enhancements to make the problem tractable e.g. belief networks. Others invent new calculi to overcome deficiencies in Bayesian inference, but still use numerical representations of uncertainty e.g. Dempster-Shafer. An alternative approach is to employ symbolic rather than numerical manipulations e.g. non-monotonic logic, TMS (Truth Maintenance Systems). All these techniques seek to handle uncertainty using specific formalisms applied locally at first and then propagated throughout the system. As yet it does not appear that non-Bayesian formalisms have been widely used in vision systems [10] [11]. In any case, we have not been concerned in this chapter with theoretical issues underlying uncertainty representations.

A different, and more pragmatic, set of approaches that do not rely on propagation in a formalised framework use global methods which handle uncertainties by hypothesis generation and hypothesis verification [12]. They can be regarded as constrained searches through the space of candidate hypotheses. In the context of scene segmentation for heavily broken images, this is likely to lead to a prohibitively large number of hypotheses unless constraints are judiciously
applied. The background of these methods is in generate-and-test, or hypothesise-and-test, a common technique in algorithm development. One process generates candidate solutions to a problem, and another process tests the candidates with a view to finding one, many or all solutions. It may be regarded as an instance of non-deterministic programming: the generator guesses an element in the domain of possible solutions and the tester verifies that the guess is correct. While easier to construct than programs that compute the solution directly, they are less efficient. The standard technique to optimise generate-and-test programs is to intertwine the tester with the generator so that fewer false candidate solutions are generated, though ability to achieve improvement is dependent on the cost of the processes involved.

Hypothesise-and-test methods have been widely employed in rule-based expert system development. In particular hierarchical hypothesise-and-test is discussed in [13], as a means of finding solutions when association between evidence and hypotheses is weak or noisy and where a mixture of heuristic classification and constructive problem-solving is required. The term ‘heuristic classification’ comes from [14] and refers to the matching between raw data and final solution via intermediate hypotheses utilising methods of ranking candidate solutions. A distinction is drawn between analytic and synthetic methods in AI, and constructive problem-solving is required when intermediate solution elements need to be combined to form composite hypotheses to account for the data.

### 5.4.2 Feature Space Representation

A hierarchical feature space for the runway recognition task can be naturally defined in ascending order of abstraction as follows:

- **Primitive-Line-Level** / **Polyline-Level** / **Line-set-Level** / **Polygon-Level** / **Polygon-structure-Level**

The input to the system is a set of edges, obtained from the image gradient using the Canny edge detector [34] with a fixed threshold. This threshold is chosen in advance and represents for most images a reasonable compromise between missing segments and number of non-significant lines. Hypotheses are propagated from low to high level in a series of transformation/verification cycles. The assumption here is that vision is goal-directed and may be regarded as a series of representational transformations interleaved with verification tasks. If a hypothesis successfully reaches the polygon level, it becomes an output of the system together with the Priority Measures of all its constituent parts.

Figure 5.2 shows a diagram of the feature space, indicating at each level the choice of transformation from one level to the next.
We note that Polygonal Approximation, Collinearity, Bounding Pair, Closed Polygon and Region/Line can be described as heuristic in that they have been derived without any underlying model of the source of uncertainty. In contrast, the remaining transformations are regarded as robust and provide more reliable Priority Measures. A perceived weakness of the system is that no robust test for collinearity at the Polyline-Level exists, and being low down in the hierarchy this gives rise to a large number of hypothesised instances which cannot be screened out until a later stage.
A hypothesise-and-test framework in conjunction with a recognition graph similar to [12] is proposed. The graph shown in figure 5.3 is derived directly from figure 5.2 and consists of multi-level decision nodes, an arc of the graph representing a transformation from a hypothesis at one level to a hypothesis at the next level. At each node a decision needs to be made about which hypotheses are to be rejected and which are to be selected for transformation to the next level. Further the most appropriate transformation needs to be chosen. This verification/ transformation cycle continues until hypotheses reach the final level, or are rejected at an earlier level; verification is thus equivalent to not being rejected. Once a hypothesis is rejected, an alternative hypothesis is chosen for propagation through the hierarchy. It can be seen that this process has the potential of becoming very complex and inefficient if hypotheses are not appropriately constrained. For example, at the Primitive-Line-Level, it is necessary to consider all triples (for statistical VP finder) or all pairs (for BP finder) of lines. Without some mechanism to direct the search for suitable candidates, the problem becomes intractable as number of lines increases.

In light of the above considerations, it is clear that the proposed framework is dependent on a hypothesis testing mechanism at each level in the hierarchy to distinguish between good and bad hypotheses. While a reliable two-class hypothesis classifier is the minimum requirement, a more efficient arrangement is possible if hypotheses can be ranked so that the most promising hypothesis can be selected for consideration. A robust Vanishing Point detection algorithm (statVP in chapter 3) can ensure that only the most promising line set (lines that are 3D parallel) be propagated to the next level. This should allow the hypothesis testing mechanism to focus on promising candidates only. This reduces the number of false hypotheses and allows the search space to be pruned. Unfortunately no such procedures previously existed for intermediate levels of vision. However, there is currently no experimental evidence to prove the need of a robust Vanishing Point detection algorithm. It is necessary to analyse the increase in computational complexity as a function of the Priority Measures.

### 5.4.3 System Overview

The system is composed of two major components, the image-processing tools and a Rule-Based System. There is no model database, because very few assumptions can be made concerning the images, except that they may contain runways or runway/taxiway structures. Only qualitative model information is available, for example isolated runways are rectangular and not as wide as runways, but camera geometry and location/orientation of the camera with respect to the scene are unknown.
Image-Processing tools include Hough Transform and Polygonal Approximation line finder, conventional Hough based VP finder (convVP), custom statistical VP finder (statVP), region-grower combined with boundary detector. The combination of region growing and line-finding is novel in that it is aimed specifically at verifying the runway/taxiway hypothesis rather than aiming at general segmentation. The goal of the chosen framework is to integrate these various strategies into a hypothesis generation and verification mechanism. It differs from other hypothesis-and-test methods developed for aerial imagery in the way verification tests have to be devised without using structural constraints derived from a model of the scene.

The Hough Transform and Polygonal Transformation are chosen because they have complementary characteristics when finding lines. The Hough-based approach performs well in presence of noise giving accurate boundary information, whereas Polygonal Approximation is noted for its ability to preserve connectivity. PA uses a ‘tolerance’ value to specify the maximum permitted error between an edgepoint and the straight-line approximation. Starting with one straight line connecting the endpoints of the edgelists, the algorithm recursively splits the edgelists when the tolerance is exceeded until all edges are within tolerance.
If there are fewer than three contributing lines to a Vanishing Point a special test referred to as Bounding Pair (BP) is required. A Bounding Pair corresponds to the longer sides of a rectangle under perspective projection. Tests for BP, Collinearity, polygon closure are described in the next section.

The System block diagram is shown in figure 5.4. The system consists of eight main processes. The initialiser initialises WM and produces the edge information for an image. There are five sets of Hypothesis-formation rules, one for each of the hierarchical levels. Focus-of-Attention (FOA) rules are used for directing the attention of the system to the more useful and worthwhile portions of the image. System control is under direction of the Supervisor which uses meta-rules to
5.4. Proposed Framework and Methodology

determine the next processing phase, which is recorded in Working Memory and can take values Initialise, Change-Level, Change-FOA as well as any of the Hierarchical Levels: Primitive-Line-Level, Polyline-Level, Line-set-Level, Polygon-Level, Polygon-structure-Level.

5.4.4 Rule-Based System

The classic recognise/aet cycle of a production system can be understood with reference to figure 5.5. The production rules, or condition-action pairs, are stored in long-term memory, and the current state of the system is defined by its Working (Short-term) Memory elements. The pattern in Working Memory is matched against the condition part of the rules to determine appropriate problem-solving actions, and operation can be viewed as a series of state transitions. The state is changed when Working Memory elements are modified by the RHS (action) operation of a firing rule, which is chosen by a Conflict Resolution Strategy from a set of rules that are said to be ‘triggered’ when their LHS (conditions) are satisfied. The firing of a single rule, and its accompanying state change completes one cycle of the production system; the cycle is then repeated until no more rules trigger. The operation, advantages and disadvantages of the production system are well-known, but for a comparison of rule-based with statistical, symbolic and connectionist learning approaches to classification, see [15].

There is evidence to suggest that using rules for representing segmentation criteria is useful in that it helps to make explicit the use (or misuse) of heuristic parameters [16]. There is however some pessimistic assessments when using rules to control low and intermediate-level processes in vision [17]. The negative conclusion was based on a different understanding of the control issues than that espoused here; we have identified control requirements more akin to high level vision, for which rules are generally recognised to be a useful means of prototyping alternative control strategies. A further reason for contemplating the use of rules is that they represent a simple well-tried structure, and recent interest in machine learning approaches applied to vision has led to some success when rules have been the chosen representation [12].
5.4.4.1 Knowledge Rules

Information about transforming hypotheses between levels is encoded in the knowledge rules as a set of situation-action pairs, where each situation consists of a conjunction (Logic AND) of logical comparisons or evaluations. The rules are classified by their actions, which cause an
appropriate hypothesis to be asserted into Working Memory depending on the level in the hierarchy. Alternative actions for the same situation (Logic OR) are accomplished by including extra rules, and allowing the Conflict Resolution Strategy to determine which rule to fire.
5.1 Knowledge Rule Sets by Hypotheses, Priority Measure and Filtering Heuristic
Thus there are five sets of knowledge rules corresponding to the five levels, Primitive-line, Polyline, Line Set, Polygon, Polygon Structure. This may be further broken down at each level according to the hypothesis that is asserted in Working Memory, labelled Output Hypothesis in Table 5.1. For example, in the current implementation there are three transformation choices at the Line-Set Level. We would like to define a control strategy that is capable of choosing between statVP, convVP and BP, noting that the choice is data-driven and dependent on type of image. If there are enough contributions from 3D parallel lines, then convVP should generate a VP hypothesis; statVP works if there are only three 3D parallel lines, and it is necessary to use BP if there are only two lines in a convergent group.

The knowledge rules also determine the values which are subsequently used in Priority Measure calculations by the Conflict Resolution Strategy. Determination of the probability in statVP has been discussed in Chapter 3. Evidential Support for a polygon to check estimated closure is shown in Figure 5.6. The Bounding Pair Overlap criterion is defined in Figure 5.7. The collinearity criterion in terms of distance between endpoints and orientation is shown in Figure 5.8; it also includes computation of average gray levels either side of candidate lines to determine whether edge gradient is consistent.

The final column in Table 5.1 refers to a filtering heuristic. It is necessary to filter using a criterion other than PM when a composite hypothesis (more than one candidate hypothesis) is being chosen since the constituent PMs cannot by themselves then be used by the CRS. For example at the Polylines-Level, the collinearity check is used to filter the candidates. In the case of the Bounding Pair the filtering heuristic and PM are the same, since there is no separate robust test for BP.

The general form of a knowledge rule is as follows:

IF Phase is L,
Focus of Attention Area is A,
Filtered Candidates from A are set F,
Candidate Hypotheses from F is set S,
Transformed Hypothesis from S at next Level is H,
Priority Measure of H is P

THEN Update Working Memory with H and P
In cases where the Priority Measure is expensive to calculate, for example statVP, the general rule is modified so that the calculation appears on the RHS rather than LHS, thus only being determined in the event of the rule firing. In such a case the Priority Measure is always available on the LHS for Conflicting Resolution. In the case of statVP, the filtering heuristic is derived from the estimated area of intersection of three lines. Use of PMs for Control is more fully discussed in the next section.

The modified rule becomes:

IF Phase is L,
Focus of Attention Area is A,
Filtered Candidates from A are set F,
Candidate Hypotheses from F is set H,
Priority Measure from filtering is PI
THEN Transform Current to New Hypothesis,
H Calculate Priority Measure,
P2 for H Update Working Memory with H and P2

5.4.4.2 Control Rules

As we have outlined the system so far, the control problem is to determine which hypothesis at which level should be pursued next in the context of the task that the system has been asked to perform. At any point in time the current system state represents partial hypotheses that have been propagated to various hierarchical levels. In common with other rule-based systems in vision, the non-deterministic aspect of control is implemented by the Conflict Resolution Strategy (CRS). The main advantage of using rules is that simple changes allow experimentation with a variety of different control strategies. It is the purpose of the control rules to specify the order in which rule sets are matched as well as to select specific items to be tested, classified as meta-rules and focus-of-attention (FOA) rules respectively.

From the recognition graph in figure 5.3 it can be seen that there are a variety of possible paths through the graph and some non-trivial control issues need to be solved. It should be remembered that this complexity has been achieved with only a few transformation choices between levels, and we have in mind a framework that could accommodate additional transformations (with their
5.4. Proposed Framework and Methodology

own distinctive properties). Furthermore there is no reason to restrict transformations to adjacent levels. For example, a closed-loop finder could be incorporated to create a hypothesis directly at the polygon level. In such a case it may be desirable to use the polygon hypothesis as focus-of-attention for the Vanishing Point finder in order to determine the associated Priority Measure. Or another example, if convVP has been run, its hypotheses could be used as focus-of-attention for statVP.

The intention here is not to explore all the control possibilities; using rules to experiment with alternative control strategies is well documented [17]. Rather we have sought to show that the basic paradigm of a hierarchical feature space with a hypothesis-and-test mechanism has a simple rule-based implementation which provides a useful prototyping framework for evaluating strategies dependent on the task at hand. The main requirement, as noted earlier, is that robust tests exist so that hypotheses are reliably selected. If the robust test is expensive to compute then an inexpensive heuristic can be utilised to make the selection, and the robust measure only calculated after the new hypothesis has been asserted. This was discussed in the previous section on Knowledge Rules in the context of whether Priority Measures should be calculated on the LHS or RHS of rules.

Conflict Resolution Strategies can be classified into general-purpose or domain-specific. The former include recency, specificity, refraction [13] among others and are used singly or in combination. While the general-purpose strategies can be useful in some situations, the control flow can be difficult to follow. We opt for conflict resolution based on Priority Measures as the dominant model. Priority Measures (PMs) at each level in the hierarchy are defined as normalised values between 0 and 1 which reflect confidence that the associated hypothesis will lead to a valid runway or runway/taxiway hypothesis. Effectively these PMs are attached to the rules as weights so that the rule with the highest PM fires.

The processing strategy is dependent on the task that the system is required to perform; we assume that we are seeking to find an instance quickly, and as processing continues we expect to find more solutions but likely to decrease in quality as processing continues. To implement this, the hierarchical level for the next phase of processing is chosen as the highest level which has a hypothesis with PM greater than a pre-defined threshold. In principle a strategy involving PMs could be used to determine the next rule set to match. However we choose a simpler alternative where each rule within a set is given the same pre-determined numerical weighting, with the weights given to each rule set ordered according to their Level. For example, a typical rule from the set of change-level rules at the Polygon-Level which has a pre-defined weighting greater than all other rule sets except the Polygon-structure-Level is as follows:

IF ‘phase’ is Change-Level,
A hypothesis at Polygon-Level has PM > Upper-polygon-threshold

THEN change 'phase' to Polygon-Level

After a Knowledge Rule has asserted a new hypothesis, it is necessary to modify 'phase' to Change-Level so that the set of Change-Level rules trigger. The change-level rule with the lowest weighting acts as a default to be fired if no other rule triggers.

Default rule is:

IF phase is change-level

THEN enlarge focus-of-attention-area

In order to keep the candidate hypotheses to a reasonable number, a lower threshold for PM is defined at each level as well as an upper threshold. If a hypothesis has a PM lower than the low threshold it is effectively removed from consideration. In fact it is placed in a tried-but-failed list in Working Memory, so that it could be retrieved at a later time.

FOA rules define a FOA area of the image as well as select data items within the FOA area. Searching for pairs and triples of symbolic items with specified properties is not particularly convenient using rules and can become quite inefficient; the advantage is more flexibility when mating with other control strategies. In our case, we keep sorted FOA Lists in Working Memory and available to be exploited by rule conditions. For example, lines are sorted by length at the Polyline-Level, and the first candidate line for a BP rule is the first element in the list and it is paired with all those below it. Those hypotheses with PMs greater than the lower threshold are asserted in Working Memory. Next time a hypothesis is generated at that level, the first candidate is the second in the list, and paired with all below it.

To perform an exhaustive search and find all instances, the lower threshold PMs are set to zero and the FOA becomes the entire image. As the lower threshold of PMs are set to zero all candidates are propagated to higher levels and this effectively brings the FOA to the entire image. However, most of the instances found are unlikely to be true positives. In such a case the reliability of the PMs affects efficiency only.
5.4.5 Implementation

The current implementation of the system is implemented in the production system CLIPS. The image processing components used for computation of average grey level values and region growing are all implemented in the C language. The following shows the excerpt of the CLIPS implementation:

```
(defunclass IMAGE-FEATURE
  (is-a USER) (abstract)
  (slot ID)
  (slot status (default ACTIVE)))

(defunclass EDGELIST
  (is-a IMAGE-FEATURE)
  (slot length)
  (slot edgelp (multiple))
  (message-handler output))

(defunclass LINE
  (is-a IMAGE-FEATURE)
  (slot startx)
  (slot starty)
  (slot endx)
  (slot endy)
  (slot rho)
  (slot theta)
  (slot l)
  (slot length)
  (slot origin-of (multiple))
  (slot hypo (default FALSE))
  (message-handler find-length)
```
(message-handler set-length)
(message-handler initialise)
(message-handler output))

(defclass POLYLINE
  (is-a LINE)
  (slot vertices (multiple))
  (slot lines (multiple))
  (slot vert-length-ratio)
  (slot bounding-pair (multiple) (default (mv-append -1)))
  (slot support)
  (message-handler find-lines)
  (message-handler set-lines)
  (message-handler delete)
  (message-handler initialise)
  (message-handler find-perimeter)
  (message-handler set-perimeter)
  (message-handler find-true-perimeter)
  (message-handler hypothesise polygon)
  (message-handler output))

(defclass POLYGON
  (is-a POLYLINE)
  (slot support)
  (slot area)
  (slot origin)
  (message-handler find-area)
  (message-handler set-area)
5.5 Results

All the test images have been shown not to succumb to detection strategies based on individual segmentation schemes. Thresholds are selected so as to minimize the possibility of false negatives, while filtering out most extraneous edges.

We present new results on test images that demonstrate operation of the production system in two different modes:

- **Polygonal Approximation plus Bounding Pair plus polygon closure**
- **Polygonal Approximation plus Bounding Pair plus polygon closure plus Region/Line verification**

The results collectively show that by combining the connectivity information preserved by polygonal approximation and priority measures such as evidential support, overlap criterion we can identify meaningful structures in the images. The polygon extracted near the centre of the image in fig.5.9 can only be discarded if higher-level information is available. The main runway shown in fig.5.10 was not identified as the edgelist information was discarded when propagated to a higher level. However, the framework proposed in this chapter permits extension to deal with situations like this by incorporating an appropriate edge detection module to the system.
Figure 5.9(Left) Edgelists input, (Right) Polygons output.

Figure 5.10(Left) Edgelists input, (Right) Polygons output.
5.5. Results

Figure 5.11 (Left) Edgelists input, (Right) Polygons output.

Figure 5.12 (Left) Edgelists input, (Right) Polygons output.
The results shown in fig.5.13 and fig.5.14 demonstrate the effectiveness of the framework proposed when a novel region growing mechanism is incorporated. Fig. 5.13(b) shows the result of applying polygonal approximation to the edge detection output. Fig. 5.13(c) shows that the system attempts to find the first bounding line of a runway/taxiway. The system starts from the longest line and attempts to lengthen it using the collinearity criterion and the grey level values consistency on either side of the line. The result of this is shown in fig. 5.13(c). Once the first bounding line is identified it then attempts to find the second bounding line by looking for lines with similar orientation. The system again attempts to lengthen any potential candidates with
similar orientation by applying the collinearity criterion. Two bounding lines are found if they meet the overlap criterion. The second bounding line is shown in fig. 5.13(d). Note that the bounding lines may or may not be the result of a grouping process based on collinearity constraints and grey level values consistency (this happen to be the case in fig. 5.13 and fig. 5.14). Once the bounding pair are located the system plants a seed near the middle of the polygon formed by the bounding pair for the region growing process to proceed. The idea is to identify structures which are linked to the identified polygon using a region growing technique. The region growing algorithm used here allows neighbouring pixels of the seed to be grouped together to form larger region if they meet all of the following criteria:

- The difference of grey level values is within a preset threshold.
- The region growing process in a certain direction will stop if the region hits a boundary. The boundaries are lines.
- The region growing process in a certain direction will stop if the region hits an 'opening' smaller than a preset threshold. The size of the 'opening' is defined as the shortest distance between boundaries.

Fig. 5.13(e) shows the result of the region growing process. The region growing process managed to identify the region connected to the left of the polygon. The grouped region on the left of the 'seed' located approximately at the middle of the polygon formed by the bounding pairs ceased to grow as it hits narrow openings whereas the region on the right of the seed ceased to grow as grey level values of neighbouring pixels exceed the preset threshold. Note that the last 'terminating' criterion of the region growing process is particularly important in heavily segmented images. Without it the region growing process could 'flood' the entire image. Fig. 5.13(f) shows the extracted polygons. The same reasoning applies to results shown in fig. 5.14.
5.6 Conclusions

Identification of runways in aerial imagery is cast as a problem of detecting rectangles under perspective projection from a set of scene segments. The Vanishing Point concept is used to define a feature cue and it is demonstrated that robust detection of such higher level features is an important aspect of system design. It is further demonstrated that by incorporating information derived from models of underlying noise in the image to perform feature detection it is possible to extract polygons using an efficient and simple rule-based classification system.

The runway recognition task is an interesting problem in that relationships between intermediate hypotheses at different levels of the hierarchy can be used as a basis for deriving heuristics to hypothesise missing data at the appropriate level. Furthermore it is possible to define these heuristics with very little knowledge of the expected scene contents. The resulting system successfully classifies a variety of taxiway/runway structures in the test images and, where only linear boundary segments are involved, this is accomplished without the need for further calls to low-level processing routines to confirm the hypothesised segments.

Fig.5.14 (a) Original image  

Fig.5.14 (b) Polylines
To machine vision researchers, the proposed framework represents a systematic methodology for combining different problem-solving strategies to solve a well-defined vision task. It is necessary to find reliable tests and a means of prioritising hypotheses to constrain the search for viable solutions. This work highlights a need for robust feature extraction procedures at each level in the feature hierarchy. For the domain of recognising man-made objects in aerial imagery, suitable
feature extraction processes did not previously exist, and it was necessary to devote a significant part of this project to developing robust methods before considering AI extraction of intermediate representations. Only the VP test was investigated thoroughly and it is anticipated that incorporating a similar approach to other levels in the hierarchy, particularly collinearity, would improve performance. For formal treatments to intermediate-level features please refer to Chapter 2. The current system should be treated as a prototype only and it is also important to note the limitations of earlier evaluation as insufficient data has been tested on the system. The future work on this system will include the formal treatment of current ad hoc PMs and extensive testing to check whether the proposed system can reliably be used to extract runways and taxiways.
5.7 References


Conclusions

6.1 Summary

The work carried out on the project can be broken down into five main sections: Feature definition and representation, hierarchical feature extraction, AI tool comparison, rule-based implementation and experimental evaluation.

The low level feature representation and extraction phases of the project were the necessary prerequisites for the AI extraction of intermediate representations. A literature survey of feature grouping criteria in chapter 2 revealed that existing schemes were mostly heuristic and failed to distinguish between various kinds of uncertainties. Consequently representations were inadequate in that they failed to address feature stability and nature of underlying noise. In particular it was necessary (also in chapter 2) to review line representation with respect to its parameters and associated error models. By analysing error distributions and conducting the Monte Carlo experiment to establish the error distribution of $\rho$ of the adopted line representation assuming that $x$, $y$ and $\theta$ are all normally distributed. This is necessary in order to design an "optimal" kernel function which is a function of both $\rho$ and $\theta$. It was concluded that a Gaussian
assumption was permissible for lines of sufficient length. This model was used as a basis for defining other higher level features in the hierarchy, namely parallelism, collinearity and junction.

The Vanishing Point (VP, the point of intersection in the image of those straight lines which are mutually parallel in 3D) proved to be a reliable feature cue for detecting runways/taxiways. However, existing VP detection schemes relied on accumulation of line pair junctions and ignored uncertainties of constituent line segments; they work well in well-structured scenes that have strong perspective, for example scenes containing office windows and corridors, but not in most of the DRA images. Consequently a novel probabilistic VP detection technique was developed in chapter 3 which takes into account underlying error models. Probabilities of line passing through a common point are computed, and subsequent combination of probabilities allows a confidence measure to be assigned, reflecting the likelihood of being a VP. In chapter 3 we also propose an adaptation of the standard accumulation method which takes proper account of the effects of noise and errors in line segment parameters. This is achieved by adjusting the way votes are cast during the accumulation. We further introduce a post-processing optimisation method that overcomes the shortcomings described above without incurring high costs due to high sampling frequency of the parameter space is. We describe the method and give estimates of the errors in vanishing point location due to under-sampling. In the third section we show results of our optimised vanishing point detector and show that we can overcome these sampling inaccuracies at cheap computational cost. We demonstrate the accuracy achieved using synthetic imagery and apply the method to two outdoor scenes. In contrast to the probabilistic based method proposed also in chapter 3 the modified accumulation approach is appropriate when a sufficient number of lines in the image intersect at the same vanishing point.

The goal of RBS is to take in edgelists as input and provide polygons as output, and for most of the DRA images one pass through the RBS is sufficient to recognise the runway/taxiway structure. Input pre-processing of the edgelists is kept to a minimum so as to preserve generality of the overall system. For example, the thresholds of the edge detector are kept fixed and chosen on the basis of a reasonable compromise between missing information and the number of non-significant lines.

The vision system presented in chapter 5 modelled runways and taxiways as rectangles under perspective projection, which enables use of what we term bounding pair as a cue for hypothesis generation (in reality the RBS uses polygons rather than rectangles in order to take into account the effects of corners and image imperfections). A bounding pair corresponds to the longer sides of a rectangle, and is only considered after line segments are linked by passing a collinearity criterion. There are, however, circumstances under which the collinearity constraint either fails to
link segments or provides improper re-construction. These conditions are detected by the RBS and a combined region growing/boundary detection approach is used to resolve ambiguities. A rule-based control scheme allows boundary detection to be integrated with a form of region growing such that linear segments define and constrain the region growing area. Parameters for seed position, region homogeneity and boundary gap threshold are dynamically determined within the RBS. Nevertheless, the system presented in Chapter 5 is only a prototype and the results presented only show the usefulness of the system for the dataset used. It is necessary to perform more extensive testing and complexity analysis to verify the usefulness of the system.

Being relatively expensive to compute, the system developed in chapter 5 allows VP to be used as part of the verification phase of the hypothesize-and-test framework after un-promising lines have been filtered out. The verification strategies proved to be robust so that heuristic generation of hypotheses could be quite liberally handled. Hypothesis generation is accommodated by heuristics which identify candidate features such as approximate parallelism as well as asserting missing data by end-point proximity or, for example, polygon closure. Priority Measures are designed to strike a balance between generating too many hypotheses (false positives) or too little hypotheses which result in loss of good hypotheses.

For the implementation of the RBS, three categories of tools were identified as candidates, C-based interpreters, Hybrid environments and AI Languages. One from each category, CLIPS, Knowledge Craft and Quintus Prolog was evaluated against an agreed set of criteria including maintainability, extensibility, knowledge representation, C-interface and run-time performance. Although CLIPS was chosen as most suitable, it is emphasised that the runway/taxiway problem did not exercise the tools across the whole range of intermediate level tasks. In particular, uncertainty handling and other knowledge representation issues associated with complexity of reasoning due to model matching might well have resulted in a different implementation choice. Chapter 4 briefly looks at the three categories of tools mentioned earlier. An extensive review of available AI tools as well as the framework of generic image understanding systems are presented.

6.2 Contributions

This thesis has had two main goals. First, we aim to build a rule-based system that is able to classify runways/taxiways in most of the DRA supplied imagery captured from unknown viewpoints. Complexity problems reported in previous work on RBS for low and intermediate level vision tasks are apparently overcome by identifying feature cues and developing extraction techniques which take into account the underlying uncertainties. The method handles
6.2. Contributions

uncertainties by hypothesis generation and hypothesis verification and can be regarded as a constrained search through the space of candidate hypotheses. Besides the advantages normally associated with RBS implementations, we single out two that are particularly relevant to the intermediate level vision task:

- a common framework for the integration and flexible control of different approaches to solve a problem.
- modular rule structure allowing the effect of parameter tuning to become more transparent, compared with embedding of heuristics in procedural algorithms where unpredicted interactions are more likely to occur.
- allow Priority Measures to be evaluated based on the performance of the resulting system.

Second, we set out to establish a formal approach to perceptual grouping. A review of existing perceptual grouping techniques in Chapter 2 identified the need to establish a formal approach to perceptual grouping. We tackled the problem by first identifying an appropriate line representation and then used the basic line model to define other perceptual groups which are generally regarded as important immediate level cues for vision systems. This work advanced the state of the art in perceptual group extraction as the existing techniques tend to be ad hoc. We have experimentally established the conditions under which the adopted line representation holds by conducting a Monte Carlo experiment. This was necessary in order to design an optimal kernel function for extracting perceptual groups such as vanishing points. Built upon the framework that we have established we developed the computational representation of higher level features such as junctions collinear line and parallel line groupings.

Two novel methods for vanishing point detection have been presented. The first takes a different perspective to detecting vanishing points as compared with the histogramming (accumulator based) methods. Instead of accumulating intersection, we compute the probability of a group of lines passing at the same point. This approach provides a probability measure for discriminating between completing hypotheses irrespective of the size of the vanishing group. In addition, its performance also degrades gracefully in noisy environments. The second novel approach is an extension of the accumulation idea which is applicable when a sufficient number of lines intersect at the same vanishing point. The main contribution of the method is that it improves the accuracy of the vanishing point estimate.

Chapter 4 reviews the various classes of expert systems building tools (ESBT). Classical topics such as knowledge representation and control mechanism have been discussed. Several commercially available ESBTs as well as the rationale behind the choice of tool for the
implementation of the vision system described in chapter 5 are discussed. The basic concept of
knowledge-based vision has been overviewed and a review of these knowledge-based vision
systems has been carried out.

In chapter 5 we presented a vision system which was capable of identifying aerial images
captured from unknown viewpoints. The system is based on the hypothesise-and-test framework.
This system differs from other hypothesised-and-test methods developed for aerial imagery in the
way verification tests have to be devised without structural constraints derived from a model of
the scene.

6.3 Limitations and Future Work

The perceptual grouping criteria, such as collinearity, parallelism and so on, currently
implemented in our vision system used ad hoc measures. It would be beneficial to replace these
grouping criteria with the computational measures developed in chapter 2.

The vision system developed in chapter 5 would, in future, be tested on more images to establish
the robustness of the system.

The vision system presented in this thesis performs satisfactorily, in that it recognised the
structures of the test images. However, in situations where the low level image processing
routines fail to detect any significant image features, the vision system is likely to miss out any
significant man-made structures. In order to cope with varying imaging conditions it would be
desirable to implement a feedback control to correct the result of edge detection, given prior
expectation of man-made structures like runways and taxiways. Some of the images contained
curved sections in the runway/taxiway structures and cannot be satisfactorily handled by current
implementation, which was designed for linear segments. In these cases the system outputs
potential start and stop points for hypothesised boundary segments with the intention that
candidate links could be confirmed by another pass through the low-level processing rather than
the bottom-up detection and linking which provided the original linear boundary segments. It
remains to be seen whether the existing hypothesise-and-test framework can accommodate
extensions such as curved links or whether to remain computationally feasible it is necessary to
introduce some explicit form of uncertainty handling into the intermediate-level processing
stage.

Nevertheless, we recognise that the proposed system only shows its effectiveness on the dataset
used and extensive testing needs to be done to verify the usefulness of the system. Also,
complexity analysis needs to be performed to identify the impact of Priority Measures.
Integrated Bibliography


