ACTIVE OBJECT RECOGNITION THROUGH NEXT VIEW PLANNING

SUMANTRA DUTTA ROY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, DELHI
INDIA

AUGUST 2000

ACTIVE OBJECT RECOGNITION THROUGH NEXT VIEW PLANNING

by

SUMANTRA DUTTA ROY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Submitted

in fulfillment of the requirements of the degree of

Doctor of Philosophy

to the

INDIAN INSTITUTE OF TECHNOLOGY, DELHI

INDIA

AUGUST 2000

Certificate

This is to certify that the thesis titled **"Active Object Recognition through Next View Planning"** being submitted by Sumantra Dutta Roy to the Department of Computer Science and Engineering, Indian Institute of Technology, Delhi, for the award of the degree of Doctor of Philosophy, is a record of bona-fide research work carried out by him under our guidance and supervision. In our opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted to any other university or institute for the award of any degree or diploma.

Dr. S. Banerjee
Professor
Department of
Computer Science & Engineering
Indian Institute of Technology
New Delhi - 110 016

Dr. S. Chaudhury
Professor
Department of
Electrical Engineering
Indian Institute of Technology
New Delhi - 110 016

To my family

Acknowledgements

There are some debts which cannot be paid back. A student owes one of this kind to his or her supervisors. Hence, it is a futile attempt on my part to express my deep sense of gratitude to Prof. S. Banerjee and Prof. S. Chaudhury. Their deep insight into problems, imaginative ideas, zest for work and rigour have been a source of inspiration and motivation for me. They have been very generous with their time, and have been a constant source of support and encouragement.

I am extremely grateful to members of the SRC – Prof. H. M. Gupta, Prof. S. N. Maheshwari, Prof. K. K. Biswas, Dr. P. K. Kalra and Prof. Shashi Kumar, for their helpful suggestions and advice. I must mention Prof. J. B. Srivastava for all his help, advice and constant encouragement.

I am indebted to Dr. S. B. Pathak and Dr. A. K. Jain of the I. I. T. Hospital, without whose efforts and constant care, I would not have been able to see the light of this day.

A very special thanks to former and present Research Scholars for all their help, academic, and otherwise – Dr. Neena Pahuja, Dr. A. R. Naseer, Dr. R. N. Moorthy, Dr. R. P. Rustagi, Dr. Rekha Goel, Dr. Ragini Choudhury, Dr. N. Rajpal, Dr. R. Garg; and Mr. R. S. Jadon, Ms. Shoma Chatterjee, Mr. D. V. L. N. Somayajulu, Mr. M. A. Sarwat, Mr. B. G. Prasad, Ms. Mona Mathur, Mr. H. Ghosh, Mr. A. S. Mandal, Mr. S. C. Jain, Ms. K. A. Nagamani, Mr. A. Rajawat, Mr. S. Baswana, Mr. M. K. Jain, and Mr. R. M. Khandekar. I have been very fortunate in having very friendly and helpful people in all the laboratories I have worked in. I am particularly

grateful to the Technical staff of these laboratories – Mr. K. R. Kaushik, Mr. D. Jaitly, Mr. F. Singh, Mr. G. Singh, Mr. J. S. Rawat, Mr. P. C. Prasad, and Mr. G. Narain for bearing with the strangest of my demands, and at the oddest hours. I will be failing in my duty if I do not acknowledge the contribution of some very cheerful and cooperative staff in the office, laboratories, and the libraries.

My words fail me when it comes to my family members - my parents Prof. S. C. Dutta Roy and Dr. Sudipta Dutta Roy, brother Shoubhik, and furry wet-nosed Lucy. This thesis is dedicated to them.

New Delhi Sumantra Dutta Roy

Abstract

Most model-based 3-D object recognition systems use information from a single view of an object. However, a single view may not contain sufficient features to recognize it unambiguously. Further, two objects may have all views in common with respect to a given feature set, and may be distinguished only through a sequence of views. A further complication arises when in an image, we do not have a complete view of an object. In this thesis, we propose two new on-line schemes for the recognition of an isolated 3-D object using reactive next view planning. The first one is based on aspect graphs, and the second is based on parts, and inner camera invariants. Both use an uncalibrated camera and simple features. We briefly describe them as follows:

Aspect Graph-based reactive object recognition An aspect constitutes a set of equivalent views of an object, with respect to a set of features. An aspect graph has nodes for an aspect, and links between adjacent aspects. We propose an integrated scheme for constructing aspect graphs from noisy feature detectors, and using them for recognizing isolated 3-D objects. Our aspect graph construction scheme accounts for errors in raw aspect data. Our system handles feature detection errors not only in the aspect graph construction process, but also in the object recognition stage, both of which use the same feature detectors. We propose a novel object recognition algorithm which uses the output of the aspect graph construction algorithm. It is not dependent on any specific feature set. The object recognition algorithm uses a probabilistic hypothesis generation mechanism. Our hierarchical knowledge representation scheme fa-

cilitates recognition and the planning process. The planning process is reactive - it utilizes the current observation and past history for identifying a sequence of moves to disambiguate between similar objects. We show results of aspect graph construction on more than 100 sets of noisy aspect data. The results of over 100 recognition experiments using the same noisy feature detectors, on two sets of models, show the effectiveness of our proposed scheme.

Part-based reactive object recognition For situations in which a complete view of a 3-D object may not be visible, we propose a new reactive, planning-based recognition algorithm which uses probabilistic reasoning. This algorithm is based on parts of an object and their relationships. We use a novel method of complete 3-D pose estimation using image-based measurements which are invariant to the internal parameters of a camera. While the earlier part of our work assumes an orthographic camera, we have formulated inner camera invariants for the more general projective case. For this problem also, we formulate a hierarchical knowledge representation scheme. The results of a large number of experiments with 2-D parts show the effectiveness of our approach in recognizing fairly complex 3-D objects, and localizing the camera with respect to the objects.

We demonstrate that by using reactive next view planning in conjunction with suitable representation schemes, it is possible to recognize 3-D objects with reasonably complex shapes, using simple features.

Contents

List of Figures

List of Tables

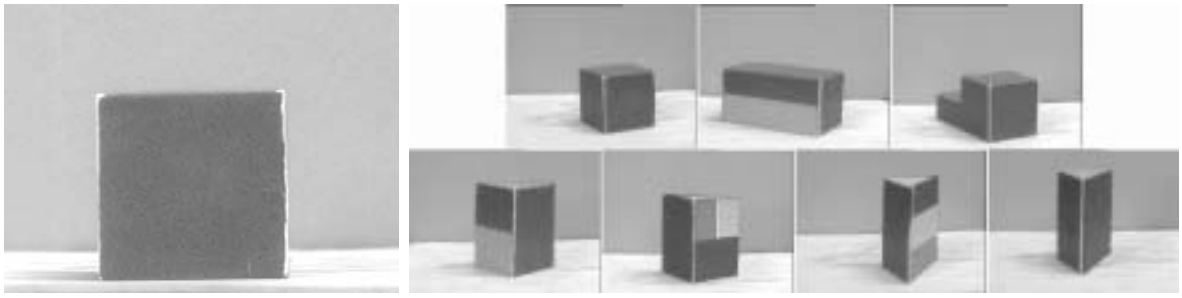
Chapter 1

Introduction

3-D object recognition involves identifying 3-D objects from a (2-D) image or a sequence of such images. Such a process involves extracting features from images, and comparing them with a stored representation of the object. Recognizing 3-D objects from images is a difficult problem, primarily because of the inherent loss of information in the projection from 3-D to 2-D. Further, the image of a 3-D object depends on factors such as the camera viewpoint and the viewing geometry. The difficulty of the recognition problem increases in the presence of noise in the feature detection process.

Fundamental to the 3-D object recognition problem is the recognition of a view of the given object. Thus, one needs features that are invariant to the camera viewpoint and the viewing transformation. For some special classes of objects such as rotationally symmetric objects, a single view may contain sufficient features for recognizing it. However, a single view-based approach may not be applicable for most objects due to the asymmetrical nature of their shape.

A further complication arises in the recognition problem if two or more objects have a view in common with respect to a feature set. Such objects may be distinguished only through a sequence of views. As a simple example, let us consider a set of 3-D objects. Let us consider the number of horizontal and vertical lines, as features.



(a) (b)

Figure 1.1: (a) The given complete view of an object, and (b) the objects which this view could correspond to

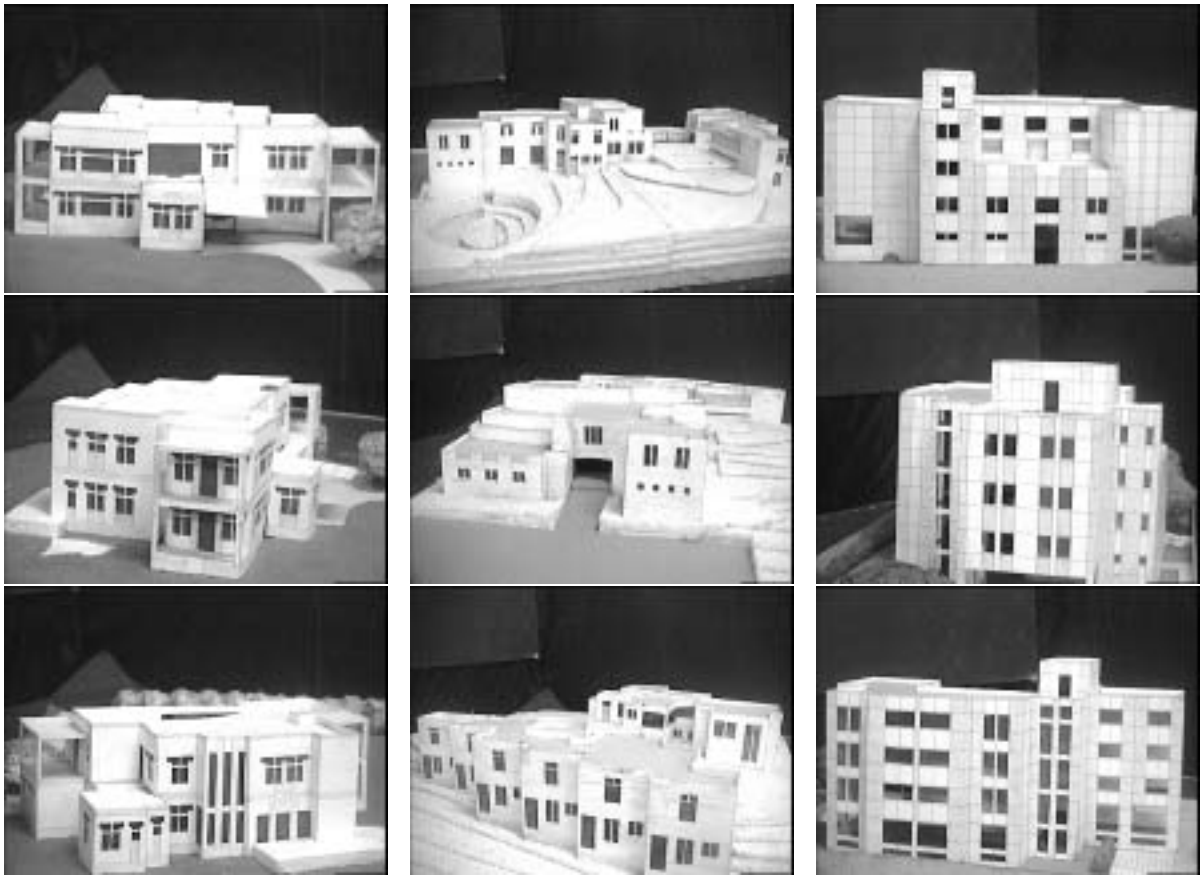
Figure 1.1(a) shows a given view. On the basis of the chosen features, this view could correspond to any of the objects in Figure 1.1(b). In other words, with each of the objects of Figure 1.1(b), it is possible to obtain a view in which we would detect only two horizontal and two vertical lines. Hence, it is not possible to determine which object the given view corresponds to, given only the single view in Figure 1.1(a).

A further complication arises when in an image, we do not have a complete view of an object. Figure 1.2(a) shows such an example. Such a view could have come from any of the three models, different views of which are shown in Figure 1.2(b), (c) and (d), respectively. Again, the identity of object cannot be uniquely determined from this one view. Further, even if the identity of the object were known, the same configuration of parts could occur at more than one place in the object. In that case, it is not possible to know the exact pose of the camera with respect to the object.

In these situations, multiple observation-based recognition strategies are needed. This may be important for a robot navigating in an environment with at least some known objects. In this thesis, we address this problem of recognizing 3-D objects through suitable planning of multiple observations.

A multiple observation-based strategy also requires a suitable representation scheme – with which the recognition scheme needs to match image-based information. One needs to maintain a relationship between different views of an object. Domain knowl-

(a)



(b) (c) (d)

Figure 1.2: (a) The given view of an object: only a portion of it is visible. This could have come from any of the models, different views of which are shown in (b), (c) and (d), respectively

edge representation, in turn depends on various factors – whether an image contains the complete object or only a portion of it, the number of degrees of freedom between the camera and the object, and the camera projection model assumed. In the context of planning the next view, one needs a sensor which can be positioned using vision-guided feedback. Such a sensor is called an active sensor, and recognition systems using such sensors are referred to as active recognition systems.

Generating hypotheses about the identity of a view, and next view planning – both form an integral component of an active recognition system. Both use the knowledge representation scheme and the information from the current view of the object to generate hypotheses about the likely identity of the object and its pose. The system plans the next move, if the current view does not have enough information for unambiguous recognition of the object. The input to an object recognition system is through sensors. Factors such as noise and non-adaptive thresholds may corrupt the output of a feature detector. Such a system should have some robustness to feature detection errors, built into it.

1.1 Problems under Investigation

In this thesis, we propose two new on-line schemes for the recognition of an isolated 3-D object using reactive next view planning. The first scheme is based on aspect graphs, and the second is based on parts, and inner camera invariants. Both use an uncalibrated camera and simple features. We briefly describe the specifications of the two problems, as follows:

1.1.1 Aspect Graph-based Reactive Object Recognition

Two or more objects could have a view in common, with respect to a set of features (as in Figure 1.1). Figure 1.3 shows a fixed camera observing an object kept on a turntable. We need to take at least one more view around the object, in order to

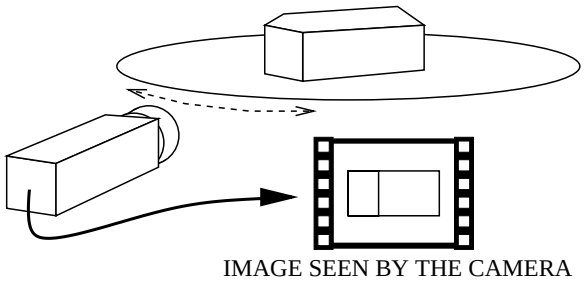


Figure 1.3: A fixed camera observing an object on a turntable

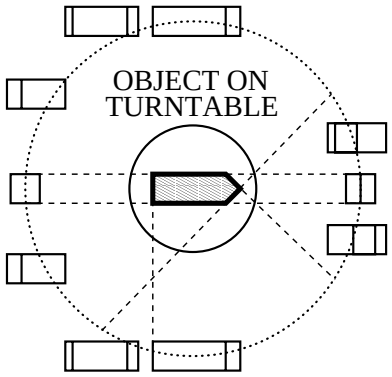


Figure 1.4: Regions corresponding to viewpoints around the object, where the view is the same with respect to a feature set.

recognize it unambiguously. However, a random sequence of views may not serve the purpose since these random views could also correspond to more than one object. This would incur a high image processing and movement cost. Figure 1.4 shows the partitioning of the space of viewpoints around the object. For all viewpoints in such a partition (an aspect), the view of the object is the same with respect to a feature set (the number of horizontal and vertical lines, in this example). It would be wasteful to have image processing operations for two or more viewpoints within the same aspect. One needs a representation scheme for an object, which accounts for the relationship between different appearances of an object – an aspect graph. The first task at hand is to construct an aspect graph for each object in the model base, given a set of feature detectors. Given an aspect graph for each such object, the recognition task involves planning a move to recognize the object unambiguously, subject to possible

memory and processing constraints.

For this problem, we present the following:

- A classification of errors in raw aspect data.
- A new algorithm to construct an aspect graph, given raw aspect data collected using noisy feature detectors.
- A function to evaluate the output of aspect graph construction algorithms.
- An evaluation function for the suitability of a feature detector.
- An aspect graph-based hierarchical knowledge representation scheme. This is not dependent on any particular feature set.
- A novel robust, reactive recognition algorithm based on next view planning. The algorithm uses probabilistic reasoning. For recognition, we use the same noisy feature detectors that were used at the aspect graph construction stage.
- Results of more than 100 experiments for both aspect graph construction, and recognition. These demonstrate the effectiveness of our proposed strategies.

1.1.2 Part-based Reactive Object Recognition

Not only could a view correspond to more than one object, the entire 3-D object itself may not lie in the camera's field of view. Only a portion of the entire object may be visible. Such a view may or may not contain any identifiable parts. Figure 1.2 shows an example where the given configuration of parts could correspond to more than one object. Figure 1.5 shows a robot with a camera fixed on it, observing a building. This robot can move translationally along all three axes, and also rotate along the **Y**-axis. Even if the identity of the object were known, the pose of the camera with respect to the object is not known uniquely. For this, the robot may position itself so as to observe the rightmost window. This sequence of moves may be unique to distinguish

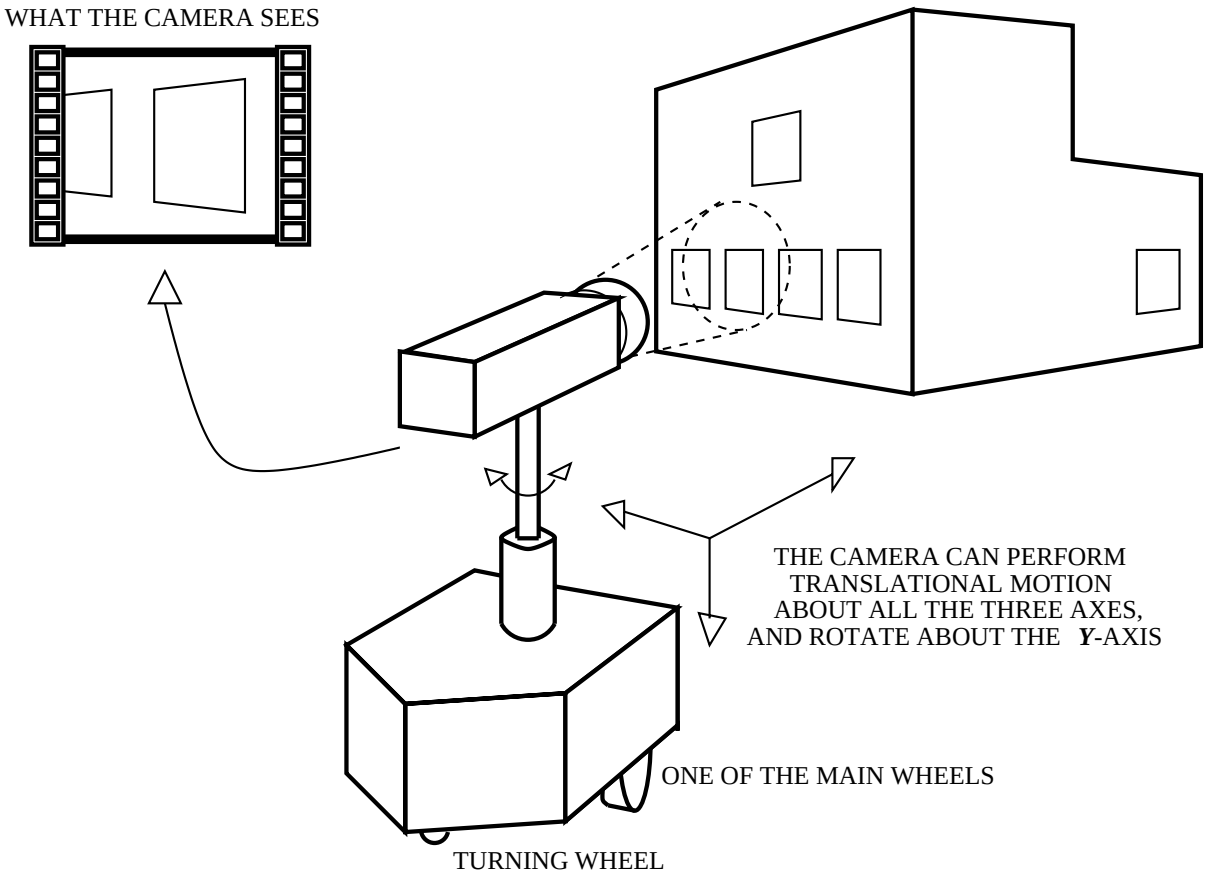


Figure 1.5: A robot with an attached camera, observing a building. The entire object does not fit in the camera's field of view. Not only is the identity of the object unknown, the robot also does not know its pose with respect to the object.

the building (from other buildings in the model base), as well as to determine its pose with respect to the building.

In this thesis, we present the following:

- Inner camera invariants: a powerful tool for pose estimation using an uncalibrated projective camera.
- A part-based hierarchical knowledge representation scheme.
- A novel part-based on-line recognition algorithm to recognize a 3-D object. The algorithm uses probabilistic reasoning. The strategy makes no assumptions about the field of view of the camera, and is robust to some feature detection errors and small errors in movement.
- Results of numerous experiments in support of our proposed strategies.

In this thesis, we show that the use of reactive next view planning and suitable representation strategies enables us to recognize 3-D objects with fairly complex shapes, even if we use simple features.

1.2 Outline of the Thesis

This thesis is divided into 7 chapters. Chapter 2 introduces the need for multiple view-based 3-D object recognition systems. Next, we classify various multi-view active 3-D object recognition systems with respect to their representation schemes and the scope or nature of their designated tasks. We then describe the motivation for the work in this thesis. We compare and contrast existing systems with our work. Finally, we summarize the salient features of our proposed strategies.

In Chapter 3, we first present a classification of different types of errors in raw aspect data. We consider the 1-DOF (single rotational degree of freedom) case, as well as the 3-DOF (all three rotational degrees of freedom) case. We propose a function to evaluate the output of different aspect graph construction algorithms. Finally,

we present a new approach to constructing aspect graphs from erroneous raw aspect data.

The aspect graph constructed using our algorithm in Chapter 3 serves as an input to our aspect graph-based object recognition algorithm. Chapter 4 describes our object recognition strategy in detail, which uses the same noisy feature detectors for the object recognition task. We propose a new hierarchical knowledge representation scheme. This plays an important role in the probabilistic hypothesis generation mechanism, as well as in planning the next view. The planning process uses information from the current observation, as well as the past history for planning a sequence of moves to identify the given object uniquely.

Chapter 5 introduces image-based measurements which are invariant to the internal parameters of the camera. We propose these inner camera invariants as a tool for pose identification and structure estimation. We use inner camera invariants for the situation when the internal parameters of the camera are not only unknown, but also may be varied – on purpose, or unintentionally.

These are used for our part-based object recognition algorithm, described in Chapter 6. Here, we consider the case when the entire object does not fit in the camera's field of view. This chapter presents a new on-line recognition scheme for the recognition and pose estimation of an isolated 3-D object, in such circumstances. The scheme also uses a probabilistic reasoning framework for recognition and planning. Our knowledge representation scheme encodes part-based information about objects as well as the uncertainty in the recognition process. This is used in the probability calculations, as well as in planning the next view.

Chapter 7 presents conclusions and scope for further work in the area.

Chapter 2

Recognition of 3-D Objects using an Active Sensor

2.1 Introduction

An active sensor is one which can be used in a purposive manner. In this chapter, we review two important applications of an active sensor. We first survey important approaches to active 3-D object recognition. Next, we review existing approaches towards another important application of an active sensor namely, that of scene analysis and interpretation.

2.1.1 3-D Object Recognition

3-D object recognition is the process of identifying 3-D objects from their images by comparing image-based features, or image-computable representations with a stored representation of the object. (For detailed surveys of 3-D object recognition and related issues, see [16], [45], [157] and [165].) Various factors affect the strategy used for recognition, such as the type of the sensor, the viewing transformations, the type of object, and the object representation scheme. Sensor output could be 3-D range images, or 2-D intensity images. 3-D range images can be obtained from the output

of a light stripe range finder, for example. 2-D images may be obtained from various means such as CCD cameras, infra-red devices, X-ray images, or from other devices operating on different ranges of the electromagnetic spectrum. 3-D objects may be classified as rigid, articulated, or deformable. In rigid objects, the distance between any two object points remains fixed. Articulated objects such as a pair of scissors, consist of various rigid parts. These objects, however permit the movement of one part of the object with respect to other parts. Deformable objects are those that are not covered under any of the two above categories. Examples include clouds, jelly, or clay.

3-D object recognition from 2-D intensity images is a difficult task, primarily because of the inherent loss of information between a 3-D object and its 2-D image. The appearance of the object depends on factors such as the viewing geometry, illumination and viewpoint. The presence of noise in the feature detection process increases the difficulty of the recognition problem.

In this survey, we primarily concentrate on 2-D intensity images taken with cameras. The work in this thesis is restricted to the recognition of rigid 3-D objects from 2-D intensity images.

2.1.2 The Need for Multiple Views

Most model-based 3-D object recognition systems consider the problem of recognizing objects from the image of a single view of an object ([16], [45], [213], [143]). Due to the inherent loss of information in the 3-D to 2-D imaging process, one needs an effective representation of properties (geometric, photometric, etc.) of objects from images which are invariant to the view point, and should be computable from image information. Invariants may be colour-based (*e.g.*, [84]), photometric (*e.g.*, [149]) or geometric (*e.g.*, [213]).

Burns, Weiss and Riseman prove a theorem in [30] that geometric invariants cannot be computed for a set of 3-D points in general position, from a single image.

Invariants can only be computed for a constrained set of 3-D points. One can impose constraints on the nature of objects to compute invariants for recognition [144] – this severely restricts the applicability of the recognition system to only a specific class of objects. Some classes of objects for which geometric invariants have been proposed, are:

- Canal surfaces [158], [159],
- Rotational symmetry [144], [199], [213],
- Repeated structures (bilateral symmetry, translational repetition) [213]

While invariants may be important for recognizing some views of an object, they cannot characterize all its views – except in a few specific cases, as mentioned above. We often need to recognize 3-D objects which because of their inherent asymmetry, cannot be completely characterized by an invariant computed from a single view. For example, certain self-occluded features of an object can become visible if we change the viewpoint. In order to use multiple views for an object recognition task, one needs to maintain a relationship between different views of an object, possibly described by image-computable invariants.

A single view may not be sufficient to recognize an object unambiguously. In fact, two objects may have all views in common with respect to a given feature set, and may be distinguished only through a sequence of views. In [94], the authors cite a simple example of a sedan and a station wagon having indistinguishable front ends, but different side views. A further complication arises when in an image, we do not have a complete view of an object. The partial view of the object could correspond to more than one portion of the object. Thus, it is not possible to know the pose of the camera with respect to the object. Figures 1.1 and 1.2 in Chapter 1 show an example each, of these two situations.

There may be another motivation for using multiple views in a recognition task. In recognizing 3-D objects from a single view, recognition systems often use complex

feature sets ([45]). Complex features such as 3-D projective invariants have been proposed only for special cases so far (*e.g.*, [213], [46]). In many cases, it may be possible to achieve the same, incurring less error and smaller processing cost using a simpler feature set and suitably planning multiple observations. A simple feature set is applicable for a larger class of objects than a model base-specific complex feature set.

A requirement for a Multi-view object recognition system is the availability of a sensor that can be purposively controlled – an active sensor. We discuss active sensors in the following section.

2.1.3 Active Sensors

An active sensor may be defined as follows:

Active Sensor An active sensor is one that can be purposively controlled. An **Active Vision** system has the ability to control the sensor parameters such as the orientation with respect to an object. Thus, vision-guided feedback may be used to position such a sensor. Such a system has other parameters that may be purposively varied, such as the focus, zoom, aperture and vergence (in two-camera system). Some specialized sensors have anthropomorphic properties, such as foveal attention mechanisms.

Some important papers proposing and elucidating the concepts of active vision and related paradigms, are the work of Aloimonos *et al.* [3], [1], [2]; Bajcsy *et al.* [7], [6]; Ballard [8]; Ballard and Brown [10]. Crowley [49] presents an on-line tutorial on active vision. A collection of representative papers in various areas of active vision is [18].

Swain and Stricker [189] survey a wide gamut of vision tasks which may be performed with an active sensor. They mention that active vision broadly encompasses attention, selective sensing in space, resolution and time. This may be achieved by

modifying physical camera parameters, or the way the data from the camera is processed.

The output of a camera usually contains a huge amount of data. An active sensor is capable of selective sensing. Hence, an attention mechanism may use an active sensor for signal selection in space, velocity and distance (*e.g.*, foveal processing, tracking an object, and focusing on regions of interest). Gaze control is a possible application – active manipulation of the imaging system in order to acquire images that are directly suited to the tasks being performed. Gaze control could be used for low level vision (*e.g.*, Murray *et al.* [147], Crowley *et al.* [51]), as well as for high level vision (*e.g.*, Rimey and Brown [172]). Thus, an active mechanism could be used to overcome a limited field of view of a camera. A related application is next view planning. Vision is a spatio-temporal process – events are distributed in time as well as space. Cost and complexity considerations often require a system to be focussed on restricted regions of a scene. Further, the current view available to a system may not even contain sufficient information for the vision task. Thus, deciding on where to look next may be task driven, feature driven, or context driven. Thus, a sequence of such sensing operations may be required. Sequential processing has the additional advantage of efficiency through directed analysis – results of each step guide subsequent steps. Object recognition and scene analysis are two example of such a vision task. Another example of an active vision task is eye-hand coordination [47]. An active vision system is often used in conjunction with robotic manipulators equipped with tactile and force sensors [189].

Tarabanis, Allen and Tsai [191] survey the field of sensor planning for vision tasks. They define the problem as follows: Given information about the environment (*e.g.*, the object and the sensors), and information about the vision task (*e.g.*, detection of certain object features, object recognition, scene analysis), the task at hand is to develop strategies to automatically determine parameter values in order to achieve the task, to a required degree of satisfaction. They classify problems into three classes:

1. object feature detection,

2. object recognition and localization, and

3. scene reconstruction.

We discuss these issues in the following sections.

Object Feature Detection

Object feature detection seeks to automatically determine vision sensor parameter values for which particular features satisfy particular constraints when imaged. These features belong to a known object in a known pose [191]. In addition to the general survey on sensor planning, the authors lay specific emphasis on systems for object feature detection systems. (A separate paper [192] presents the authors' own MVP system in detail.)

A related topic is planning for complete sensor coverage of 3-D objects. A recent work in the area is that of Roberts and Marshall [173], who present a viewpoint selection scheme for complete surface coverage of 3-D objects. They also propose a strategy for selecting a number of views that allow each object face to be viewed according to specified constraints on viewpoints and other features. The authors show results with a fixed camera and turntable. Some important earlier work in the area include those of Cowan and Kovesi [48], Tarbox and Gottschlich [193] and Mason and Grun [138].

Object Recognition and Localization, and Scene Reconstruction

In the above classification of Tarabanis, Allen and Tsai [191], the last two categories are more related to a recognition and interpretation task. We adopt a similar classification scheme, with a slightly different interpretation of the above two terms. Given an active sensor and a set of feature detectors, the fundamental problems involved in a multiple view-based recognition system are

- the design of a suitable modeling and representation scheme, and

- an identification mechanism which can exploit properties of the sensing process and the modeling scheme.

Based on the representation scheme and the scope or nature of the recognition strategy, we classify different multi-view recognition systems into two categories:

1. Object recognition systems, and

2. Systems for scene analysis

In the first class of systems, we consider systems whose aim is to primarily recognize a given object and its pose. Such systems typically assume that the entire object is visible in a given view. In the second class of scene analysis systems, we consider systems whose aim is to explore and and analyze a given scene. Such a scene may contain one or more objects, known or unknown. In such cases, the entire scene to be analyzed may not be visible in one view – the sensor may 'see' only a part of it at a time. While recognition may not be a primary aim of such systems, they may involve recognition of some components of a scene. We describe these two categories in detail in Sections 2.2 and 2.4, respectively.

2.2 Active Object Recognition Systems: A Survey

In this section, we survey different existing active object recognition systems. An active object recognition system needs to have a representation scheme to store information about the 3-D structure of an object. First, we review different representation schemes. Subsequently, we review different recognition schemes and their characteristics.

2.2.1 Representation Schemes

Object representation schemes used for model-based object recognition systems include ([115]): wire-frame representations, constructive solid geometry (CSG), spatial-occupancy representations (*e.g.*, voxels, octrees), surface boundary representations, generalized cone or sweep representation, skeleton representations, and aspect graphs. Appearance-based approaches (*e.g.*, [146]) to object recognition use appearance rather than shape, for matching. However, only a few of the above approaches have been used in multi-view object recognition systems. While wireframe models have an inherent ambiguity in interpretation, feature extraction is difficult in volume or surface-based approaches. Skeleton representations and generalized cones are applicable for recognition of only a specific class of objects.

We may also classify representation schemes on the basis of whether they represent the entire object, or model its *parts*. We first discuss two important representation schemes for active recognition systems, and then discuss part-based representation schemes. We conclude this section with a discussion on different methods used to handle uncertainty.

Most active object recognition systems consider either of the following three representation schemes, or their variants:

- Appearance-based parametric eigenspaces
- Multidimensional Receptive Field Histograms
- Aspect graphs

These three are *view-based* – they encode information about different 2-D views of a 3-D object. View-based recognition systems exist for recognizing objects from a single view of the object. Breuel [25], [27], [26] analyzes view-based recognition and compares its performance theoretically and empirically, with a common model for 3-D bounded error recognition [95]. He shows that the probability of false positive and false negative matches in view-based recognition systems is not substantially

different from the probability of simple errors in other commonly used recognition systems. Furthermore, he derives an upper bound on the number of views needed to be stored by a view-based recognition system in order to achieve zero probability of negative matches. The author describes simulations and experiments on real images to suggest view based recognition as a robust and simple alternative to 3-D shape-based recognition methods.

Appearance-Based Parametric Eigenspaces

Murase and Nayar [146] propose the idea of appearance-based methods using parametric eigenspaces. A basic observation is that the shape and reflectance are intrinsic properties, which are constant for a rigid object. However, the pose and illumination may vary from scene to scene. The authors propose a scheme to automatically learn 3-D objects from their appearance in 2-D images. An important advantage of this method is the ability to handle the combined effects of shape, pose, reflection properties and illumination. Furthermore, it is possible to learn appearance-based object representations off-line. For systems using such a representation, the recognition problem becomes one of appearance matching, rather than shape matching.

Appearance-based methods require a large number of images of the object – with different poses, and illumination conditions. The images of the objects are normalized with respect to size and illumination conditions. Both in the pose and illumination space, consecutive images are correlated to a large degree. Hence, a need arises to compress the large set of images to a low-dimensional representation of object appearance. Each normalized image is written as a column vector in raster scan order. Next, the normalized images are stacked together , and the covariance matrix is calculated. The first k eigenvectors are used to represent the stacked matrix of images. This process is discussed in detail, in [146]. In the recognition phase, the image vector is projected to the eigenspace. The object which has a minimum distance between the projected image vector and its manifold, is considered to be present.

Multidimensional Receptive Field Histograms

Multidimensional Receptive Field Histograms [179] are based on the idea that local structure is an important component of the appearance of an object. The local structure can be characterized by a vector of local features measured by local operators such as Gaussian derivatives or Gabor filters. This technique represents appearances of objects by the joint statistics of such local neighbourhood operators.

Aspect Graphs

Aspect graphs are a popular representation tool for 3-D object recognition systems. Aspects represent object appearances that are equivalent with respect to a particular set of features. An aspect graph consists of nodes which correspond to aspects. Links between nodes represent transitions from one aspect to another. We examine a number of aspect graph-based and related representations in Chapter 3 (*e.g.*, [40], [82], [58], [60], [59], [177], [93], [109], [110], [174], [135], [136], [36], [37], [145]). We consider different kinds of aspect graphs and examine the advantages and disadvantages of each. We also analyze existing approaches for handling errors in aspect graphs.

The aspect graph-based approach is more general than the other two approaches in that appearance-based information may be used to construct an aspect graph.

Part-Based Representations

Object recognition systems use representations for the complete object. However, there have been some instances where systems consider the representation of an object in terms of its *parts*. Existing part-based recognition systems typically consider the object to be wholly composed of identifiable parts. Here, we review two part-based approaches. The first is based on volumetric primitives, and the second on appearance-based parts. Existing part-based recognition systems usually use information from only a single view. The works of Dickinson *et al.* [56], [57] are significant examples of active recognition systems using a part-based representation.

Here, we look at two part-based representations namely, geons and appearance-based parts:

1. **Geons**

   Biederman's *Recognition by Components* theory [17] proposes the concept of volumetric primitives, called geons (short for 'geometric ions'). The Cartesian product of contractive shape properties gives rise to this set of volumetric primitives. Bergevin and Levine [13], [14], [15] propose methods of automatically extracting geons from relatively perfect 2-D line drawings. Single view-based recognition systems such as [15], [58], [60] and [59] use geons as representation tools. In [15], Bergevin and Levine propose a system for generic object recognition from a single line drawing of an object, by parts. Dickinson and co-workers [58], [60], [59] use an augmented aspect hierarchy using geons as the basic volumetric primitives. They use this augmented aspect hierarchy for active 3-D object recognition in [56] and [57].

2. **Appearance-based parts**

   Another approach to part-based representation is that of Huang, Camps and Kanungo [105], [35]. The authors define appearance-based parts as "polynomial surfaces approximating closed, non-overlapping image regions that optimally partition the image in a minimum description length (MDL) sense." Their single view-based recognition systems consider the advantages of appearance-based representations. Additionally, the idea of recognizing parts and not the whole object gives the system robustness to occlusion and segmentation variations.

Methods for Representing Uncertainty

Common methods for representing uncertainty are probability theory, the Dempster-Shafer theory [55], [181], and fuzzy logic [210], [211]. A representation scheme based on probability theory is a Bayes Net. (Bayes nets, and their variants are also known as Belief networks, Bayesian networks, and probabilistic networks.) However, a Bayes

net is a far more general AI-based representation scheme (as against the above schemes specifically used for modeling 3-D objects). A Bayes net (first proposed by Pearl in [156]) is a graph which represents the joint probability distribution of a set of variables. Nodes in the graph represent variables, and directed links represent conditional probabilities. The Bayes rule is used for updating the probabilities of nodes having a particular label, given that successor nodes have particular labels. References [156], [150], [176] describe Bayes nets in detail. Dickinson *et al.* [56], [57] use a variant of a Bayes net for their recognition system (Rimey and Brown [172], [170], [171], [200] use Bayes nets for scene analysis), while Hutchinson and Kak [106] use the Dempster-Shafer theory to represent uncertainty. Some scene analysis systems use fuzzy logic (*e.g.*, [153]).

In the following section, we discuss more about different representation schemes, in conjunction with the recognition strategies.

2.2.2 Recognition Strategies

We now present recognition strategies for some important active 3-D object recognition schemes. We classify these on the basis of the next view planning strategy:

1. Systems incorporating explicit planning algorithms, and
2. Systems which take the next view to minimize an ambiguity function

We discuss different schemes as follows. While the first two belong to the first category above, the rest belong to the second.

Goldberg and Mason

Goldberg and Mason [92] investigate the problem of determining object pose. They explore this problem in a Bayesian framework, using an object diameter function. For polygonal objects, this is piecewise sinusoidal. During a squeeze grasp, the object rotates to reduce its diameter, terminating in a local minimum. Such an operation

reduces uncertainty. This also converts the diameter function into a piecewise constant function, which is more amenable to analysis. They use breadth-first search to expand the state space. Assuming a uniform distribution of initial poses and a frictionless parallel jaw gripper, they demonstrate the automatic planning of a sequence of grasps that optimize a robot's expected throughput.

Gremban and Ikeuchi

Gremban and Ikeuchi [94] present a scheme for planning multiple views in an object recognition task. They use Aspect-Resolution Trees built on the basis of aspect diagrams for planning multiple observations for object recognition. The authors show results for a vision-based sensor, and a haptic sensor, and give examples of recognition of objects based on sensors to detect specularities and their properties. These determine the aspects of the object. They consider this to be a challenging domain, since a single image of a specular object yields very little information about the overall object shape (specularity being a local phenomenon). Further, many different object poses can yield the same pattern of specularities. The authors mention that it is possible to configure parallel jaw grippers as sensors that report when a jaw has contacted an object, and the distance between the jaws. With such a configuration, the gripper can be used to measure object diameters. This information can be associated with object poses.

Hutchinson and Kak

In their work on planning sensing strategies in a robot work cell with multi-sensor capabilities, Hutchinson and Kak [106] use an aspect graph to represent information about the objects in their model base. They present a system for dynamically planning sensing strategies, based on the current best estimate of the world. They automatically propose a sensing operation, and then determine the maximum ambiguity which would remain if the operation were applied. The system then selects the

operation which minimizes the remaining ambiguity. They use the Dempster-Shafer theory to combine evidence and analyze proposed operations.

Liu and Tsai

Liu and Tsai [134] describe a multiple view-based 3-D object recognition system. Their setup has two cameras and a turntable. They use silhouettes as features. The off-line model base construction involves taking silhouette views from a set of fixed camera views. The system first reduces ambiguity by taking images from above the turntable to normalize the shape of the top view, position the object centroid, and align the principal axis of the object. The system then takes a side view, and analyzes its features. The object is repeatedly rotated by 45^o and this system repeats the above process, till the object is recognized.

Callari and Ferrie

Callari and Ferrie [33] base their active object recognition system on mode-based shape, pose and position reconstructions from range data. They estimate Bayesian probabilities with neural networks. Their system takes the next view based on the move which minimizes the expected ambiguity in terms of Shannon entropy.

Schiele and Crowley

Schiele and Crowley [180] develop an analogy between object recognition and the transmission of information through a channel based on the statistical representation of 2-D object appearances. They use multidimensional receptive field histograms. Transinformation enables the determination of the most discriminative viewpoints. The proposed strategy moves the camera to the most discriminant viewpoint of the hypothesized object.

Dickinson and co-workers

Dickinson and co-workers [56], [57] present an active object recognition scheme which integrates attention and viewpoint control. Their representation scheme is similar to that of Biederman [17]. The Cartesian product of contractive shape properties gives rise to a set of volumetric primitives called *geons*. The authors consider three properties:

1. cross-section shape,
2. axis shape, and
3. cross-section size variation

The earlier works of Dickinson *et al.* [58], [60], [59] consider a similar framework in recognizing an object from a single view. The Cartesian product of the dichotomous and trichotomous values of these properties give rise to a set of 10 volumes. They consider 10 modeling primitives namely, 1. Block, 2. Tapered block, 3. Pyramid, 4. Bent block, 5. Cylinder, 6. Tapered cylinder, 7. Cone, 8. Barrel, 9. Ellipsoid, and 10. Bent cylinder. To construct objects, the volumes are simply attached to one another.

The above recognition schemes use an augmented aspect hierarchy as their data structure. Aspects are used to model the (typically small) set of volumetric part-classes from which each object in the database is constructed. The augmented aspect hierarchy considers relations between boundary groups (representing all subsets of contours bounding the faces), the faces, the aspects, and finally, the volumetric primitives themselves. The entities at each level are linked with one another. Each link is associated with a conditional probability.

Dickinson *et al.* present a case for using regions. They use conditional probabilities captured in the augmented aspect hierarchy to define a measure of average inferencing uncertainty. On the basis of this, they conclude that the value of this parameter for faces is less than that for boundary groups. It is pointed out that the advantage

would be realizable if the cost of extracting the features corresponding to the two are comparable.

Their attention mechanism exploits the augmented aspect hierarchy to map target objects down to target faces. Target faces are in turn, compared to image faces. In selecting which recovered face to focus attention on, they use a decision theoretic approach using a Bayesian framework. They use a structure known as the aspect prediction graph to drive the sensor to a new position from which an object's part can be disambiguated.

Borotschnig and co-workers

Borotschnig *et al.* [21] present an active 3-D object recognition system that uses appearance-based information. They extend the idea of the off-line system of Murase and Nayar [146] to an on-line case. They use a parametric eigenspace, and augment it with probability distributions – to capture possible variations in the input images due to errors. Their system chooses as the next view a move, which minimizes the average entropy.

2.3 A Comparative Analysis of Active Object Recognition Systems

Active recognition systems have been proposed which can work with different assumptions about the nature of the sensors and the environment, the degrees of freedom between the object and the sensor, and the object models themselves. We discuss the following issues with respect to different active 3-D object recognition systems:

1. *Features used for modeling and view recognition*

   While many approaches such as those of Hutchinson and Kak [106] and Liu and Tsai [134] use geometric features, the scheme of Gremban and Ikeuchi [94] is independent of the features used. The latter present results with geometric

and photometric information. Goldberg and Mason [92] use haptic features. Appearance-based methods such as that of Borotschnig *et al.* [21] use pixel information from an entire image. Dickinson *et al.* [56], [57] use volumetric primitives, which are associated with a high feature extraction cost. The same is true for the super-ellipsoids of Callari and Ferrie [33].

2. *The system setup and viewing geometry*

   Existing systems such as those of Hutchinson and Kak [106], Liu and Tsai [134], Callari and Ferrie [33], Dickinson *et al.* [56], [57], Gremban and Ikeuchi [94], and Borotschnig *et al.* [21] assume that the object completely fits into the camera's field of view. Borotschnig *et al.* [21] assume a single degree of freedom between the object and the sensor. While Gremban and Ikeuchi [94] have experimented with such a case, they propose extensions to higher degrees of freedom. The viewing geometry used for recognition using a single view is usually affine (such as in [107]), or projective (as in [213]). Most multiple view-based approaches using geometric features, implicitly or otherwise, assume the camera model to be orthographic.

3. *Efficient representation of domain knowledge*

   The knowledge representation scheme should support an efficient mechanism to generate hypotheses on the basis of the evidence received. It should also play a role in optimally planning the next view.

   Dickinson *et al.* [56], [57] use a hierarchical representation scheme based on volumetric primitives. Due to the non-hierarchical nature of Hutchinson and Kak's system [106], many redundant hypotheses are proposed, which have to be later removed through consistency checks. Borotschnig *et al.* [21] use a parametric eigenspace-based representation, which is associated with a high storage and processing cost.

4. *Speed and efficiency of algorithms for both hypothesis generation and next view*

planning

Hypothesis generation should be fast, and incur minimal error. The next view planning strategy acts on basis of these hypotheses.

In Hutchinson and Kak's system [106], the polynomial-time formulation overcomes the exponential time complexity associated with assigning beliefs to all possible hypotheses. However, their system still has the overhead of intersection computation in creating common frames of discernment. Consistency checks have to be used to remove the many redundant hypotheses produced earlier. Though Dickinson *et al.* [56], [57] use Bayes nets for hypothesis generation, their system incurs the overhead of tracking the region of interest through successive frames.

5. *Nature of the next view planning strategy*

   The planning scheme should ensure adequate discriminatory ability between views common to more than one object in the model base. The cost incurred in this process should also be minimal. The system should, preferably be on-line and reactive – the past and present inputs should guide the planning mechanism at each stage.

   While schemes such as those of Borotschnig *et al.* [21] are on-line, that of Gremban and Ikeuchi [94] is not. An off-line approach may not always be feasible, due to the combinatorial nature of the problem. An on-line scheme may result in significant reduction of the search space. An on-line scheme has the additional capability to react to unplanned situations, such as errors.

6. *Uncertainty handling capability of the hypothesis generation mechanism*

   Approaches such as those of Goldberg and Mason [92], Gremban and Ikeuchi [94], and Liu and Tsai [134] are essentially deterministic. An uncertainty-handling mechanism makes the system more robust and resistant to errors compared to a deterministic one. Dickinson *et al.* [56], [57] and Borotschnig *et al.* [21] use

Bayesian methods to handle uncertainty, while Hutchinson and Kak [106] use the Dempster-Shafer theory. In the work of Callari and Ferrie [33], the ambiguity in super ellipsoid-modeled objects is a function of the parameters estimated, on the basis of which the next move is determined. Schiele and Crowley [180] use a transinformation-based mechanism to propose the next move.

2.4 Active Scene Analysis Systems: A Survey

The aims and domains of scene analysis systems are extremely diverse – even though active sensing and recognition usually form a common thread in each of them. Given their diverse natures, systems for scene analysis generally use specialized schemes for knowledge representation. They use these in conjunction with the recognition and analysis strategies. In this section, we review some important classes of scene analysis systems, and their data representation and control schemes.

Next View Planning for Data Acquisition: Range Images

Maver and Bajcsy [141] present a strategy for next view planning which exploits occlusion information. They consider a data acquisition problem where a sensor is placed in an unknown environment it has to investigate with its sensors. Their system works with range images obtained using a laser-camera triangulation system that can measure the distance only of those portions of a 3-D scene that are simultaneously illuminated by laser light, and visible to the camera. The system exploits characteristics of the sensing process, to acquire yet-unknown 3-D information about the scene of interest. The foci of attention are the occluded regions. Two types of occlusions can be encountered in their system: either when the reflected laser light does not reach the camera, or when direct laser light does not reach the scene surface.

A related approach is that of Massios and Fisher [139]. The authors also use range images, and propose a quality criterion. This quality criterion aims at obtaining views that improve the overall range data quality of the imaged surfaces.

Another recent approach is that of García, Velázquez and Sappa [85]. They present a two stage algorithm for determining the next view, using range images. The first stage applies a voting scheme that considers occlusion edges. Most of the surfaces of the scene are recovered this way. The second stage fills up the remaining holes through a scheme based on visibility analysis. The idea of having the expensive visibility computations at the end of the exploration process, is to increase the efficiency of the scheme.

A related vision-based approach is one of recovering the surface shape with an active sensor. We discuss this in the following section.

Active Recovery of Surface Shape using a Vision-Based Sensor

Kutulakos and Dyer [126] present an approach for recovering surface shape from an occluding contour of an object, using an active sensor. They use the fact that if the viewing direction is along a principal direction for a surface point whose projection is on the contour, it is possible to recover the surface shape (curvature). In their strategy, an observer purposefully changes its viewpoint in order to achieve a well-defined geometric relationship with respect to a 3-D shape. They use only curvature measurements on the occluding contour to recover qualitative shape information. Similar work of this group may be found in [122], [128], [123], [127], [129], [124], [125].

Scene Geometry Interpretation and Exploration

Whaite and Ferrie [190] present a system for the interpretation of scene geometry in the form of parametric volumetric models. They describe ambiguity as a local probabilistic property of the misfit error surface in the parameter space of super-ellipsoid models. This is an ellipsoid of confidence in which there is a given probability that the parameters can be found. They project back the ellipsoid of confidence into 3-D space in order to obtain the shell in which the true 3-D surface most probably lies. The construction of an *uncertainty image* demonstrates the notion of uncertainty as a

local property of the fitted model's surface. They propose a technique that uses this information to plan for the next view – which minimizes the ambiguity of subsequent interpretation. Their system uses 3-D range data collected with a laser range finder.

Marchand and Chaumette [137] present an autonomous active vision system for 3-D reconstruction of static scenes. They do not assume any prior knowledge of the number, localization, and the dimension of the different objects in the given scene. The authors handle the perception-action cycles at various levels: from the definition of perception strategies for scene exploration, down to the automatic generation of camera motions using visual servoing. A controlled structure-from-motion method is used for reconstruction. This allows an optimal estimation of parameters of geometrical primitives (the authors use the scene to be composed of polyhedra, cylinders and line segments). The authors propose perceptual strategies which are able to appropriately perform a succession of such individual primitive reconstructions in order to recover the complete spatial structure of the scene. They present two algorithms to ensure exploration of the scene. The first is an incremental reconstruction algorithm based on the use of a prediction/verification scheme involving decision theory and Bayes nets. Such an arrangement allows the system to get a high-level description of the observed part of the scene. The second algorithm is based on the computation of new viewpoints for the complete reconstruction of the 3-D scene. The authors show the results of experiments in a robotic cell in support of their proposed algorithms.

Systems for 'Finding Waldo': Incorporating Colour and Other Cues

Grimson and co-workers [96] present an attentive active visual system which integrates visual cues to fixate candidate regions in which to recognize a target object. The system combines colour and stereo cues to perform figure/ground separation. The input to the system is a representation of the target model, which includes both colour and shape information. The system scans the entire field of view with minimum overlap of images. It takes a stereo pair of images and subsamples them to support

a wide field of view with moderate resolution. Edge detection is the next step. A knowledge of colour primitives of the target object helps in selecting salient regions in each image – the system retains edges in only those regions. The next step is stereo edge matching. The disparity associated with the match is used to verge the cameras and extract a high-resolution pair of images. The matched edges, and those which lie within a small distance of the matched ones, are input to an alignment-based recognition method [107]. The authors demonstrate this system to quickly find a small target in a cluttered environment by focusing its resources in areas most likely to contain that target.

While the system of Ennesser and Medioni [76] is not an active recognition system, some ideas they present may be used to advantage in a active vision system. They present a method for selecting a set of likely locations of an object in a colour image. It is based on matching local histograms with the model. The idea of interest is their method of handling scale. Each local histogram begins with an initially small size, and is intersected with the model according to its current size. If the initial intersection is promising enough, the histogram is sequentially allowed to grow by a fixed amount in each cardinal direction. This is done as long as this process does not decrease the normalized intersection measure, after updating and rescaling to the new size. The final area is removed from the image and memorized as a possible candidate, while a new local histogram starts to grow next to it. The algorithm uses heuristics to avoid local minima.

Systems for 'Finding Waldo' essentially rely on active visual search. The following section describes an important paradigm in visual search namely, using intermediate objects.

Using Intermediate Objects to Enhance Visual Search

Wixson and Ballard [207], [208] describe an active vision system that use intermediate objects to improve the efficiency of visual search. They show examples of trying to

search for an object using an active camera, whose internal and external parameters can be varied, and which is also capable of foveal processing. They propose *indirect searches* to be more efficient as compared to direct searches for an object, in two cases. The first is when intermediate objects can be recognized at low resolutions and hence found with little extra overhead. The second is when they significantly restrict the area that must be searched for the target. Indirect searches use spatial relationships between objects to repeatedly look for intermediate objects, and look for the target object in the restricted region specified by these relationships. The authors present a mathematical model of search efficiency that identifies the factors affecting efficiency and can be used to predict their effects. They report that in typical situations, indirect search provides up to about an 8-fold increase in efficiency.

Selective Attention for Scene Analysis

Rimey and Brown [170], [200], [171], [172] suggest the use of Bayes Nets for scene analysis through selective attention. They describe a select active vision, TEA-1 in [170], [171], [172]. They mention that the efficiency of a selective vision system comes from processing the scene only where necessary, to the level of detail necessary, and with only the necessary operators. TEA-1 uses not only the prior knowledge of a domain's abstract and geometrical structure, but is also reactive – it also uses information from a scene instance gathered during analysis. The knowledge representation is through 4 kinds of Bayes Nets, (the PART-OF net, the expected area net, the IS-A tree, and the task net) which are used to store different kinds of domain knowledge. TEA-1 uses benefit-cost analysis for the control of visual and non-visual actions. The authors show the application of TEA-1 in analyzing dinner table scenes. In [200], simulation results are shown on a train scene with a train on a track, a train station, herds of cows and barns.

Jensen, Christensen and Nielsen [114] adopt a similar approach. The conditional probabilities for their Bayesian network is obtained by subjective assessment. They

show results on a network for discrimination between a British and a Continental breakfast table scene.

Dynamic Relevance in a Vision-Based Focus of Attention Strategy

Baluja and Pomerleau [11] use the concept of Dynamic relevance in their vision-based focus of attention strategy. The system ascertains the relevance of inputs by exploiting temporal coherence. In their system, relevance is a time-varying function of the previous and current inputs. It dynamically allocates relevance to inputs by using expectations of their future values. The expectation of what features will be there in the next frame decides which portion of the next visual scene will be focussed on. The system uses a neural network with an input layer, a hidden layer and two sets of units in the output layer: one for the output, and one for the reconstructed inputs. The weights between the input layer and the hidden layer, and those between the hidden layer and the outputs are trained to reduce the task error alone. The weights between the hidden layer and the reconstructed inputs are trained to reduce prediction error only. The architecture further has a feedback between the reconstructed 'next' inputs, and the input layer. The input layer actually uses the concept of a saliency map to make the system use filtered inputs. Thus, the information that is not relevant to the task will not be encoded in the hidden layer. The authors demonstrate the application of their ideas in various environments – vision-based autonomous control of a land vehicle, vision-based hand tracking in cluttered scenes, and the detection of faults in the plasma-etch step of semiconductor wafers.

Visual Surveillance

Buxton and Gong [31] present a visual surveillance system for tracking moving objects and interpreting their patterns of behaviour. They use conceptual knowledge of both the scene and the visual task to provide constraints to the problem. The control of the system is through dynamic attention and selective processing. The authors use

belief networks to model dynamic dependencies between parameters involved in visual interpretation. They present experimental results on a traffic surveillance application, using a fixed pre-calibrated camera model and pre-computed ground plane geometry. To recognize different objects in the scene, they use volumetric models. The system tracks objects across frames.

Environment Map Building

Nashashibi and co-workers [148] describe a system for indoor scene terrain modeling using multiple range images. This relies on two grid-based representations: the local elevation map, and the local navigation map. The authors describe their interpolation method to build their grid-based representation of the terrain – the local elevation map. Elevation data are used to build a symbolic grid representation call the local navigation map. Here, each terrain patch is assigned to a pre-defined class of terrain. They do not assume any *a priori* world model or landmarks to be available.

Lebègue and Aggarwal [131] propose a scheme for the extraction an interpretation of of semantically significant line segments for a mobile robot. Their scheme is particularly suited for environments which can be described by lines of particular 3-D orientations. The low-level processing stages are designed to increase the usefulness and the quality of the extracted features for a semantic interpretation. The detection and interpretation processes provide a 3-D orientation hypothesis for each 2-D segment. This is used to estimate the robot's pose, and delimit the free space visible in the image. A motion stereo algorithm (3D structure from motion) uses the orientation data to fully estimate the 3-D Euclidean structure of the scene. The authors use a similar approach in their later work [132] to build a CAD model of the environment.

Faugeras, Ayache and Faverjon [79] also use visual cues for map-building. This paper proposes a method to build visual maps by combining noisy stereo measurements. The authors propose the idea of a Realistic Uncertain Description of the Environment

(RUDE) which incorporates local information – it is attached to a specific reference frame, and incorporates both geometrical information, as well as the related uncertainty information. They relate this to pixel uncertainty, and show how the RUDE corresponding to different frames can be used to relate them by a rigid displacement, and a measure of its uncertainty. Finally, they use the relations between frames to update the associated RUDE and decrease the uncertainty. In a more recent work, Faugeras [78] describes deterministic computational geometry-based methods for map building.

Elfes [75] describes a sonar-based system for map building and navigation. The system uses sonar data to build a multilevel description of the robot's surroundings. The system uses probability profiles to determine empty and occupied regions. The author proposes a robust method to combine sensor information that may be erroneous, as well as have an uncertainty associated with them. This enables range measurements from multiple points of view to be integrated into a sensor-level sonar map. These are used for path planning and navigation. The probabilistic sensor-level sonar maps serve as the basis of a multilevel description of the robot's operating environment.

Pagac, Nebot and Durrant-Whyte [155] propose an algorithm for map-building using evidential reasoning. They use the Dempster-Shafer theory to fuse new sensor information into the map. This evidential approach with its multi-valued hypotheses allows quantitative analysis of the quality of the data. The authors show examples on real world data in support of their algorithm.

While the above two approaches model uncertainty using probability theory and the Dempster-Shafer theory, respectively, the approach of Oriolo, Ulivi and Vendittelli [153] uses fuzzy logic. They present a system for real-time map building and navigation for autonomous robots in completely unknown environments. The approach of the authors is based on the alternate execution of two fundamental processes – map building, and navigation. Map building consists of collecting range measures using the robot's ultrasonic sensors and processing it in order to build a local representa-

tion of the surrounding area. This representation is then integrated in the global map constructed thus far, by filtering out insufficient or conflicting information. In the navigation phase, an A^*-based planner generates a local path from the current robot position to the goal. The robot follows the path up to the boundary of the explored area, terminating its actions if unexpected obstacles are encountered. The authors present results with a $NOMAD$ 200 mobile robot.

Zelinsky [212] presents an algorithm for mobile robot exploration in an unknown environment. The robot maps the environment only to the extent necessary to achieve the goal. The paper assumes tactile sensors. The algorithm uses a quad-tree data structure to to model the plan of the environment, and uses the distance transform methodology [20] to generate paths for the robot. The algorithm generates paths by treating unknown regions in the environment as free space. As and when the robot encounters obstacles, it updates its environment and the planning process modified. The algorithm successively ensures that the lengths of the paths are optimized. The author presents simulation results of the action of the algorithm.

Asada [5] extends the work of Elfes [75] and proposes a method for building a 3-D world model for sensory data from from outdoor scenes. His system allows for other sources of input data, such as range and video data. First, a range image ('physical sensor map') is transformed to a height map ('virtual sensor map') relative to a mobile robot. the height map is segmented into unexplored, occluded, traversable and obstacle regions from the height information. The system classifies obstacle regions into artificial objects or natural objects according to their geometrical properties such as slope and curvature. Height maps are integrated into a local map by matching geometrical parameters and updating region labels.

Thrun [194] presents an approach that allows mobile robots to automatically select landmarks. Landmarks are chosen based on their utility for localization. He achieves this task by training landmark detectors so as to minimize the *a posteriori* localization error that the robot is expected to make after querying its sensors. The system trains a set of neural networks, each of which maps sensor input to a single value estimating

the presence or absence of a particular landmark. He shows that these approaches outperform approaches in which a human operator hand-selects landmarks and trains neural networks to recognize them. The author also applies the Bayesian approach to control the direction of the robot's camera. He shows that using active perception helps in faster localization than with a static camera configuration.

In [196], Thrun, Burgard and Fox address the problem of building large-scale geometric maps of indoor environments with mobile robots. In their experiments, they investigate a restricted version of the map-building problem, where a human operator tele-operates the robot through an environment. They pose the map-building problem as a constrained, probabilistic maximum-likelihood estimation problem. They present an algorithm for getting the most likely map from the data, as well as the most likely path taken by the robot. They demonstrate experimental results in cyclic environments of sizes up to 80 by 25 metres.

Map building strategies use two major paradigms to represent the environment – grid-based, and topological. While grid-based methods produce accurate metric maps, their complexity often prohibits efficient path planning. (Schiele and Crowley [178] examine the problem of pose estimation using occupancy grids.) Topological maps do not suffer from this problem. However, accurate and consistent topological maps are often difficult to learn and maintain in large-scale environments. Thrun [195] proposes an approach that integrates both paradigms. The approach learns grid-based maps using artificial neural networks and Bayesian integration of sensor output. Topological maps are generated on top of the grid-based maps by partitioning the latter into coherent regions. The paper presents results for autonomous exploration, mapping and operation of a mobile robot in populated multi-room environments.

Reactive Robot Navigation

Crowley [50] describes a navigation system based on a dynamically maintained model ('composite local model') of the local environment. The system uses ultrasonic sen-

sors. The composite local model integrates information from the ultrasonic range sensors, the robot's touch sensor, and a pre-learned global model, as the robot moves through its environment. The system includes a technique for correcting odometric errors. The robot learns the global model of its environment, and uses it for global path planning.

Basri and Rivlin [12] present a method of representation that may be useful for a reactive vision-based navigating robot. The authors extend the work of Ullman and Basri [198] on recognition by a linear combination of models. They analyze three basic tasks in autonomous robot navigation namely, localization, positioning and homing. They define localization as the act of recognizing the environment *i.e.*, assigning consistent labels to different locations. Positioning is the act of computing the coordinates of the robot in the environment. Homing is the task of returning to a previously visited position. The authors represent a scene as a set of 2-D views and predict the appearances of novel views by linear combinations of the model views. They assume weak perspective projection. For the case when the weak perspective assumption is invalid, they propose using either a larger number of models, or an iterative solution for perspective distortions. They present a method for localization from only a single 2-D view without calibration. They have a similar method for positioning, and a simple qualitative algorithm for homing.

Kosaka and Kak [120] present a fast vision-guided robot navigation system FINALE using model-based reasoning and the prediction of uncertainties. Although this system is primarily meant for a path planning task, many ideas presented here are relevant for scene interpretation as well. The vision system maintains a model of uncertainty and keeps track of the growth of uncertainty as the robot travels towards the goal position. For example, the uncertainty with respect to a line is modeled as the convex hull for the two ellipses of uncertainty at the end-points of the line. These ellipses of uncertainty depend on the mean vector and covariances matrices of the uncertainty in position associated with the end points of the line. The system uses these uncertainty estimates to predict bounds on the locations and orientations of

landmarks expected to be seen in a monocular image. This reduces the complexity of searches for establishing correspondence between landmarks and image features. There is a sequential reduction in uncertainty as each image feature is matched successfully with a landmark, allowing subsequent features to be matched more easily.

Fennema *et al.* [81] describe an autonomous robot navigation system at the University of Massachusetts, Amherst. Model-based processing of the visual sensory data is the primary mechanism used for controlling movement through the environment, measuring progress towards a given goal, and avoiding obstacles. They assume a partially modeled unchanging environment containing no unmodeled obstacles. the system integrated perception, planning and execution of actions. The system models the environment using a CAD modeler. The system uses reactive planning processes that reason about landmarks that should be perceived at various stages of task execution. The correspondence information between image features and expected landmark locations (the system uses line features) is used at several abstraction levels to ensure proper plan execution. For example, when the image of a landmark moves differently from what is expected, the system makes corrections to the motor system. The system proposes partially-developed tentative plans about what action to take next. These are developed depth-first with less developed details away from the current location. Failures trigger changes in plans at various levels. Landmarks selected from the model are used to steer the robot.

Chenavier and Crowley [44] describe a method for position estimation for a mobile robot, using vision-based and odometric information. The system uses landmarks for correcting the position and orientation of a robot vehicle. There are numerous other examples of landmark-based navigation strategies *e.g.*, Levitt and Lawton [133], Onoguchi *et al.* [152],

Mataric [140] integrates a map representation into a reactive, subsumption-based mobile robot [28]. She presents an implementation on a mobile robot equipped with a ring of sonar sensors and a compass, and programmed with a collection of simple and incrementally designed behaviours. The robot performs collision-free navigation,

dynamic landmark detection, map construction and maintenance, and path planning. Given any known landmark as a goal, the robot plans and executes the shortest known path to it. If the goal is not reachable, the robot detects failure, updates the map, and finds an alternative route.

Burgard *et al.* [29] present a modular and distributed software architecture of an autonomous interactive tour-guide robot. The architecture integrates localization, mapping, collision avoidance, planning and various modules concerned with user interaction and Web-based tele-presence. The authors demonstrate results of the deployment of their system in a densely populated museum for a period of six days.

Chen and Tsai [42] present a incremental-learning-by-navigation approach to vision-based autonomous land vehicle (ALV) guidance in indoor environments. The approach consists of three stages – initial learning, navigation and model updating. In the initial learning stage, the ALV is driven manually, and environment images and other status data are recorded automatically. The authors then build the initial environment model off-line. In the navigation stage, the ALV moves along the learned environment automatically. It locates itself by model matching (using multi-weighted generalized Hough transform), and records necessary information for model updating. The approach uses information about vertical lines. In the model-updating stage, the system refines the learned model off-line. A more precise model is obtained after each navigation-and-update iteration. The authors show results on a real ALV in indoor corridors.

2.5 An Analysis of Scene Interpretation Systems

Similar to our analysis of object recognition schemes, we discuss some issues in the context of scene analysis systems.

1. *Features used for modeling and view recognition*

   Existing scene analysis systems primarily work with geometric features, irrespective of whether they are obtained from a vision-based sensor, a range sensor, a haptic sensor, or ultrasonic sensors. Systems such as that of Grimson *et al.* [96] additionally use colour information.

2. *The system setup and viewing geometry*

   Object data acquisition systems, and systems for recovering surface shape, both assume that the object completely fits into the sensor's field of view. For the other application areas, the entire scene may not fall within the sensor's field of view. The aim of these systems is to use the sensor in a purposive manner, to fulfill its task. The sensors for scene analysis applications typically have three translational and one rotational degree of freedom (*e.g.*, navigational applications as in the system of Kosaka and Kak [120]). Some systems such as those of Rimey and Brown [170], [200], [171], [172] do not make any explicit assumptions about the viewing geometry. Systems such as that of Basri and Rivlin [12] explicitly assume weak perspective projection, while those of Lebègue and Aggarwal [131], [132] assume perspective projection.

3. *Representation of domain knowledge*

   Different scene analysis applications need different representation schemes to fulfill their requirements. Rimey and Brown [170], [200], [171], [172] use Bayes nets to represent domain knowledge, and encode task specifications. In their system for 3-D reconstruction of static scene, Marchand and Chaumette [137] propose a prediction/verification scheme using decision theory and Bayes nets. The visual surveillance system of Buxton and Gong [31] uses many different representations for its components, such as Bayes nets and ground plane maps. Artificial neural networks form the architecture of systems that use some form of learning, such as those of Baluja and Pomerleau [11], and Thrun [194].

   As mentioned in Section 2.4, Active map-building strategies primarily consider

grid-based maps (*e.g.*, Nashashibi *et al.* [148], Elfes [75], Orioli *et al.* [153]) as against topological maps (*e.g.*, [209]). Thrun [195] proposes an approach that integrates both paradigms. Basri and Rivlin [12] represent a scene in terms of 2-D views as against the representation of Marchand and Chaumette [137], who use explicit 3-D geometric models.

4. *Algorithms for hypothesis generation and next view planning*

   Algorithms vary according to the nature of the application. Systems may use explicit scene information to compute the next view. The approach of Maver and Bajcsy [141] uses occlusion information, while that of Kutulakos and Dyer (*e.g.*, [126] uses curvature measurements on the occluding contour. The strategy may be based on minimizing an uncertainty function as in [190]. Grimson and co-workers [96] use colour and stereo features in their multi-stage algorithm. Rimey and Brown [170], [200], [171], [172] use a benefit-cost analysis to plan actions. There may be a high-level general control paradigm, as in the approach of Wixson and Ballard [207], [208]. Map-building algorithms primarily focus on algorithms for integrating evidences taken at different points in space and time, such as that in [196]. Reactive navigation strategies primarily focus on reaching a goal, subject to positional uncertainty and navigational obstacles. The system of Oriolo *et al.* [153] for example, is designed for a completely unknown and unmodeled environment.

5. *Nature of the next view planning strategy*

   All systems described in Section 2.4 have an on-line component. The on-line nature of such systems illustrates their reactive property – an essential requirement of an active scene analysis system.

6. *Uncertainty handling capability*

   Some approaches such as that of Maver and Bajcsy [141] are deterministic. Most systems handle uncertainty explicitly. Uncertainty representation schemes

include probability theory (as in the work of Elfes [75]), Dempster-Shafer theory (as in the system of Pagac *et al.* [155]), and Fuzzy logic (*e.g.* the real-time map building and navigation system of Oriolo *et al.*).

2.6 The Thesis in Perspective

Sections 2.2 and 2.3 survey and analyze different active 3-D object recognition systems. We repeat the process for different scene analysis systems (Sections 2.4 and 2.5) due to the commonality of many issues in the two problems. Based on this survey and analysis, we draw the following conclusions:

- Geometric features are useful in a recognition task. We may supplement them with other features such as colour and photometric information. Some recognition systems are tightly coupled with the properties of the particular features they use. However in some cases, we may have a system that is not explicitly based on any particular set of features.

- The 1-DOF (rotational) case between the object and an orthographic camera is an important and fairly complex problem. The complexity of the recognition task increases with the number of degrees of freedom between the object and the camera, and the increasing generality of the camera model – from orthographic to projective.

- The knowledge representation scheme plays an important role in both generating hypotheses corresponding to a given view, as well as in planning the next view.

- The domain of application influences the design of the recognition algorithm. In general, the system should plan a minimal number of steps (each step corresponds to an epoch where sensor data is processed) in order to achieve its goal. Such a process is subject to memory and processing limitations, if any.

- The next view planning strategy should preferably be on-line. The system should balance plan-based schemes and pure reactive behaviour. A pure reactive behaviour may veer a system away from its goal. On the other hand, the reactive nature of a system allows it to handle unplanned situations.

- A system with uncertainty handling capability gives it an edge over one that uses a pure deterministic strategy – the former is more robust to errors.

Now, we look at some issues that existing object recognition systems do not address.

- Accounting for noise at the model-building stage

- The use of the same noisy feature detectors at the recognition stage

- Recognition of a 3-D object when the object does not fit in the camera's field of view

- Addressing cases when there is no assumption about a knowledge of internal parameters of the camera, or their constancy. The internal parameters of a camera may be changed either involuntarily, or done on purpose.

Motivated by these observations, we address two problems in the recognition of a 3-D object using a planned sequence of multiple views. Both cases assume an uncalibrated camera.

The first problem relates to the case of an aspect graph-based recognition scheme, using noisy feature detectors. For this case, we assume that the entire object fits in the camera's field of view.

- We propose a new aspect graph construction scheme which accounts for errors in raw aspect data. We handle both the single DOF (rotational) case, as well as the 3-DOF one, in which all three rotations are permitted. For this problem, we assume the camera model to be orthographic.

- We propose a function to evaluate the output of an aspect graph construction algorithm.

- Our system handles feature detection errors not only in the aspect graph construction process, but also in the object recognition stage, which use the same feature detectors. To the best of our knowledge, no related work addresses these issues.

- We propose a novel object recognition algorithm which uses the output of the aspect graph construction algorithm, for the 1-DOF case.

- The object recognition algorithm uses a probabilistic hypothesis generation mechanism. The hypothesis generation mechanism has low-order polynomial-time complexity.

- Our hierarchical knowledge representation scheme facilitates recognition and the planning process. The hierarchy itself enforces different constraints to prune the set of possible hypotheses.

- The planning process is reactive - it utilizes the current observation and past history for identifying a sequence of moves to disambiguate between similar objects.

- An important feature of our system is that it is independent of the type of features used.

- We show experimental results of recognizing fairly complex objects with a set of simple features, and suitably planned multiple views.

For situations in which a complete view of a 3-D object may not be visible, we propose a new recognition algorithm based on parts of an object and their relationships.

- We use a novel method of complete 3-D pose estimation using inner camera invariants.
- Such an approach enables us to work with situations in which camera internal parameters (such as the focal length, on which the field of view depends) may change.
- While the earlier part of our work assumed an orthographic camera, we have formulated inner camera invariants for the projective case, with a commonly used camera model.
- For this problem also, we formulate a hierarchical knowledge representation scheme.
- Our probabilistic recognition algorithm accounts for uncertainty in pose estimation, movement, as well as in the feature detection process.
- For this problem, we use geometric features in conjunction with other features. The parts may be 2-D or 3-D entities.
- We show the results of a number of experiments in support of our proposed strategy. We demonstrate its robustness to certain classes of feature detection errors, and small movement errors.

In this thesis, we show that using a combination of reactive next view planning and suitable knowledge representation schemes, we can effectively recognize fairly complex 3-D shapes using simple features.

Chapter 3

Constructing Aspect Graphs for Object Recognition

Many 3-D object recognition strategies use aspect graphs to represent objects in the model base. A crucial factor in the success of these object recognition strategies is the accurate construction of the aspect graph, its ease of creation, and the extent to which it can represent all views of the object for a given setup. Factors such as noise and non-adaptive thresholds may introduce errors in the feature detection process. This chapter presents a characterization of errors in aspect graphs, as well as an algorithm for estimating aspect graphs, given noisy sensor data. We present extensive results of our strategies applied on a reasonably complex experimental set. We demonstrate its application to a robust 3-D object recognition method in Chapter 4.

3.1 Aspect Graphs

Any object recognition strategy, based on single or multiple views, requires robust identification of a view of the given object. Aspect graphs are a convenient tool for multi-view, viewer-centric representation of a 3-D object. Many 3-D object recognition strategies use aspect graphs (*e.g.*, [40], [106], [94], [56], [57]). In this section, we

discuss aspect graphs and related issues.

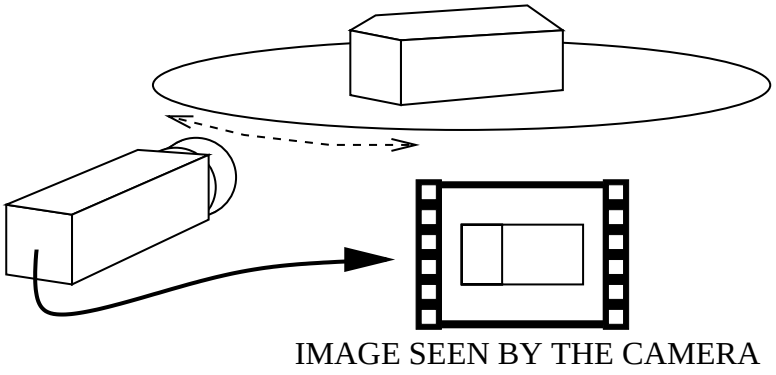
3.1.1 Aspects and Classes

We start with the definitions of the terms 'Aspect' and 'Class':

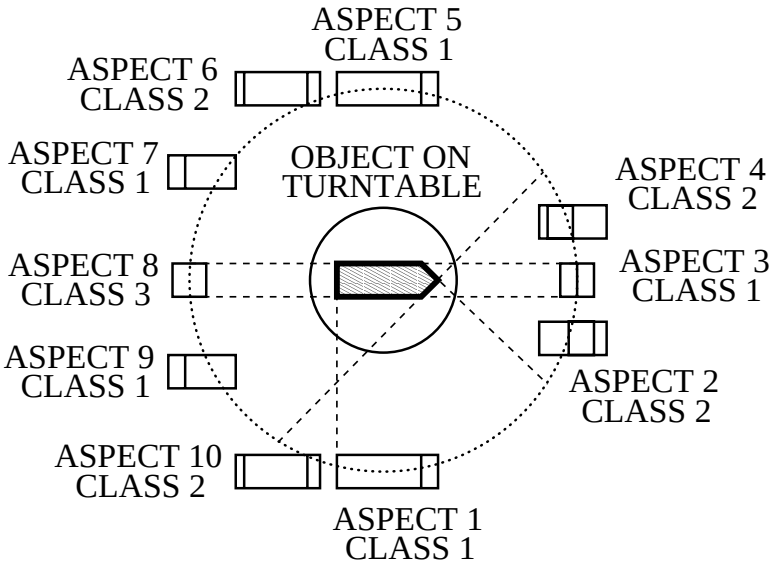
Aspect Koenderink and van Doorn [117], [118] define aspects as topologically equivalent classes of object appearances. Chakravarty and Freeman [40] adopt a similar approach in their definition of the 'Characteristic Views', and their uses in object recognition. Since sensors may be of different types (geometric, photometric, etc.), Ikeuchi and co-workers generalize this definition – Object appearances may be grouped into equivalence classes with respect to a feature set. These equivalence classes are aspects [93]. Thus, an aspect is a collection of contiguous sites in viewpoint space which correspond to the same set of features.

Class A Class (or Aspect-Class) is a set of aspects, equivalent with respect to a feature set. Thus, a class represents a set of features.

We illustrate these definitions with two simple examples. In these examples, we consider the classical definition of the term aspect *i.e.*, the one based on topology. For the most general case, there may be 6 degrees of freedom (hereafter, DOF) between the object and the sensor – three translational, and three rotational. We consider two cases – one for a single rotational degree of freedom, and the second for all three rotational degrees of freedom between the object and the sensor. *Throughout this thesis, we refer to these as the '1-DOF' and the '3-DOF' case, respectively.* Figure 3.1(a) shows an example of the 1-DOF case. The setup has an object on a turntable, which can rotate about its spindle (the central axis). Let us consider as features, the number of horizontal and vertical lines in an image of the object. We assume an orthographic camera. For such a camera model, the polyhedral object has 10 aspects belonging to 3 classes. Figure 3.1(b) shows these aspects along with their angular extents in



(a)



(b)

Figure 3.1: (a) An example of the 1-DOF case (a single rotational degree of freedom between the object and the camera), and (b) the object with its aspects and classes

the viewing space around the object. In this example, the viewing space around the object is a circle with the centroid of the object at its centre. The radius is the the distance between the centre and the centre of the camera coordinate system.

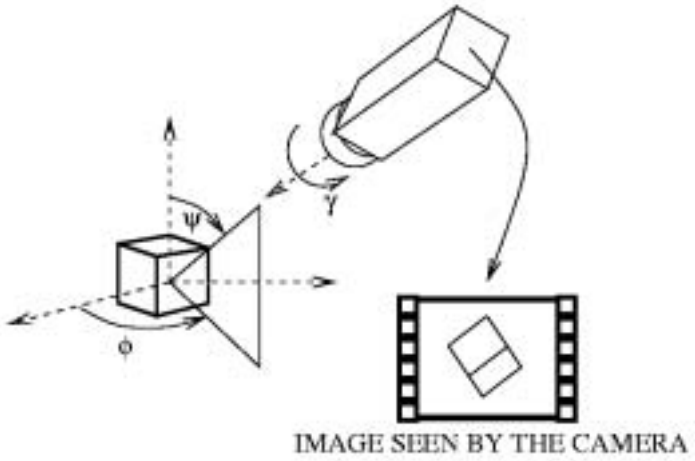
Figure 3.2 shows the corresponding examples for the 3-DOF case. Figure 3.2(a) shows the three rotational degrees of freedom between the object and the camera – the latitude and longitude angles ψ and ϕ respectively, and the rotation angle γ of the camera about its own axis. We assume an orthographic camera again. The viewing space around the object is a sphere, with its centre coinciding with the centroid of the object O. The radius of the sphere is the distance between the centre of the sphere, and the centre of the camera coordinate system. Figure 3.2(b) shows some aspects and classes of this object for a quadrant of the viewing space. The angles corresponding to these aspects in this case are solid angles. Points A, B and C on the viewing sphere represent aspects of one class. All points such as D which lie on the equator of the viewing circle, (with the exception of points such as B and C which view a face of the cube head-on) represent another class. Points such as F also belong to the same class. E and its neighbouring points belong to another class. An aspect corresponding to this configuration would correspond to the solid angle subtended by the surface of the sphere bounded by (but not including) arcs AB, AC and BC.

3.1.2 Aspect Graphs and their Classification

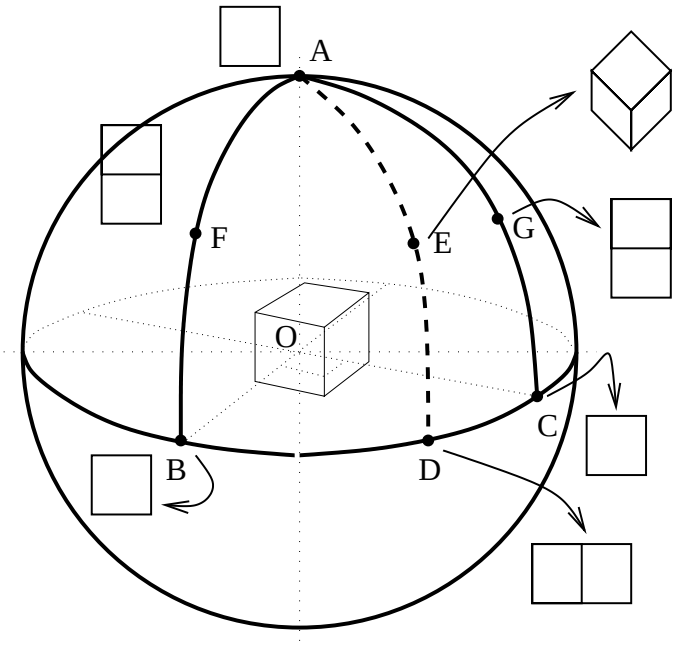
We define an aspect graph as follows:

Aspect Graph An aspect graph consists of nodes which correspond to aspects. Links between nodes represent transitions from one aspect to another. A link is often referred to as an *accidental view*, or a *visual event* ([73], [74]). An aspect graph has a node for each aspect of the object, and a link for each possible visual event.

An aspect graph partitions the space of viewpoints around an object into maximal regions. Every viewpoint in each such region (an aspect) gives the same view of the



(a)



(b)

Figure 3.2: (a) An example of the 3-DOF case (all three rotational degrees of freedom between the object and the camera), and (b) the object with its aspects and classes

object with respect to a particular feature set. An aspect is a *general viewpoint* ([73], [74]) – one from which an infinitesimal movement in any possible direction in viewpoint space, results in a view that is equivalent to the original view. Changes in the aspect (visual events) take place at the boundaries between regions. Two aspects are connected by a visual event if and only if their corresponding regions are adjacent in the viewpoint space. A link represents an *accidental viewpoint* – one for which there is at least one direction in which an infinitesimal movement results in a view that is different from the original.

Depending on the method used to construct them, aspect graphs may be classified as:

1. Exact aspect graphs, and

2. Approximate aspect graphs (hereafter, AAGs)

We discuss these two types of aspect graphs in the following sections.

3.1.3 Exact Aspect Graphs

Analytical approaches are used to construct exact aspect graphs – directly from object shapes and surface characteristics. A limitation of such an approach is its applicability to only a specific class of objects. Existing approaches are primarily limited to geometric features obtained from CAD models of objects – changes in topological properties of object appearances determine the visual event catalog for the object. Depending on the projection method used, analytical approaches can be further classified as orthographic and perspective. Algorithms exist for specific classes of objects (not all of these classes are mutually disjoint) –

- polygons ([86]),

- transparent smooth objects ([117]),

- piecewise-smooth objects and algebraic surfaces ([43], [166], [162], [184], [167], [168], [164], [169]),
- curved objects ([32], [121], [162], [184], [163], [164], [72], [104], [22]),
- smooth shapes from volumetric data ([151]),
- polyhedral objects – 2.5-D polyhedra under orthographic projection ([38]),
- solids of revolution under orthographic projection ([71], [154]),
- arbitrary polyhedra under orthographic projection ([89], [90], [88], [130], [160], [201], [182]),
- convex polyhedra under perspective projection ([185], [186], [187], [202]),
- arbitrary polyhedra under perspective projection ([186]), and
- objects with moving parts ([23]).

The above were examples of vision-based approaches for aspect graphs. In [116], the authors present an example of a *Haptic aspect graph* for 3-D object shapes.

We now examine three related approaches.

Finite Resolution Aspect Graphs Shimshoni and Ponce [182] address the problem of computing finite-resolution aspect graphs of polyhedral objects. They assume an orthographic camera with limited spatial resolution, and simple geometric features.

The Scale Space Aspect Graph Eggert *et al.* [73], [74] propose the concept of a *Scale Space Aspect Graph*. The aim of their work is to reduce the large set of aspects (obtained from analytical methods) to a smaller set of "most important" aspects. In a more recent work on scale space aspect graphs, Pae and Ponce [154] extend the concept to a class of solids of revolution.

PREMIO and Related Approaches Lu, Shapiro and Camps [135], [136] present a relational pyramid approach to class (view-class/aspect-class) determination. For modeling objects in their object recognition system PREMIO, Camps, Shapiro and Haralick [36], [37], [34] use CAD models, surface reflectance properties, light sources, sensor characteristics and performances of feature detectors to build a *Vision model*. PREMIO has a feature prediction module. The match between a prediction and a relational structure based on image data is a relational matching problem. The authors use an IDA^* search on an interpretation tree.

3.1.4 Approximate Aspect Graphs (AAGs)

AAG construction approaches usually tessellate the viewpoint space into uniform partitions. Adjacent viewpoints which give the same appearance of the object with respect to a feature set, are grouped together to form an aspect. Each partition is represented by a site. A site is a representative viewpoint for a partition, at which sensor data is collected. Examples of uniform partitioning strategies, and related issues are the schemes of Horn [102], Korn and Dyer [119], Fekete and Davis [80], Goad [91], Ikeuchi and co-workers [108], [109], [94], Chen and Kak [41], Ballard and Brown [9], Jain and Hoffman [113], and Flynn and Jain [83].

The uniform partitioning approach is independent of the object shape and structure, the sensor, and the feature set. In other words, one can use the same algorithm to construct an AAG, irrespective of the object, the sensor, or the feature set. The effects of finite resolution are generally not considered in analytical approaches (exceptions exist, *e.g.*, [182]). We consider the uniform partitioning approach in this thesis.

We now review some related approaches.

Geometric Features: The Use of CAD Modeling Some authors use CAD modeling for a uniform partitioning approach. For example, the BONSAI sys-

tem of Flynn and Jain [82] identifies and localizes 3-D objects in range images by comparing relational graphs extracted from CAD models to relational graphs extracted from range image interpretation. In the more recent work of Munkelt [145], the author uses CAD modeling to generate a set of normalized synthetic views for his image interpretation system Aspik.

Volumetric Primitives and the Augmented Aspect Hierarchy Dickinson and co-workers [58], [60], [59], [56], [57] use a set of volumetric primitives to compose different objects. They consider the uniform partitioning approach to generate AAGs of each volumetric primitive. They propose an augmented aspect hierarchy encodes relations between boundary groups (representing all subsets of contours bounding different faces), the faces, the aspects, and finally, the volumetric primitives themselves.

The Use of Specular Features The above three were examples of using geometric information. Sato, Ikeuchi and Kanade [177] present a system for the recognition of specular objects. In an off-line phase, the system generates synthetic images from a representative set of viewpoints.

The Use of Physical Modeling An interesting variant of the uniform partitioning approach is the work of Robey, West and Venkatesh [174]. The authors suggest the use of physical modeling for the prediction of various feature types visible from different viewpoints. An advantage of their method is that their technique is independent of the shape of the object. Their only assumption is that it should be possible to represent the object using a boundary representation. A second advantage of their method is that it can make use of not only geometric features, but also other feature types such as specularity. Other authors such as Ikeuchi and co-workers (*e.g.*, [94]) also use similar methods (using colour and reflectance).

3.1.5 Geometric Features: Exact Aspect Graphs, and High-Resolution AAGs

As mentioned earlier, exact aspect graphs have been proposed for specific classes of objects, and are based on geometric features alone. Let us now compare AAGs of such objects with exact aspect graphs. The resolution of the tessellation of the viewing space determines the extent to which an AAG will approximate an exact aspect graph better. *However, such an AAG and an exact aspect graph may still not correspond with each other.* The reasons could be one or all of the following:

- The actual 3-D object may not exactly correspond to the geometric CAD model. For example, a model edge may not be perfectly straight, or angles may be different. Small changes in object shape can drastically alter the set of visual events. This affects the partitioning of the viewpoint space into aspects. Such discrepancies would show up in an AAG. Thus, an AAG can prove to be more useful than an exact aspect graph in a recognition task with the actual object.

- An exact aspect graph gives equal importance to each visual event. Thus, an exact aspect graph may include details which an observer may almost never see in practice. Certain events may exist due to fragile alignments, but may occupy a negligible fraction of viewpoint space. The term *View degeneracy* refers to the accidental alignment of spatially distinct scene parts as seen from a camera viewpoint [206], [61].

- Exact aspect graphs assume infinite resolution in the projected image. Even for an AAG with a fine degree of tessellation, limitations of the feature detectors could preclude the detection of some aspects with small extents in viewpoint space. The same could also occur for visual events. Thus, two views that differ by such an aspect would be the same, as observed using an experimental setup.

- There may be another consequence of view degeneracy. Given arbitrarily high resolution of the imaging process, view degeneracy may not be a problem. This

is so because such accidental alignments would occur for a vanishingly small fraction of the possible viewpoints. For an experimental setup, the set of such viewpoints is much larger due to the finite resolution of cameras and feature extraction operators.

- An exact aspect graph idealizes a camera as a point object. For a given experimental setup, certain views are unlikely to be witnessed because of their location in space [74].

In the following section, we examine different approaches to these problems, and the related issue of errors in aspect graphs.

3.1.6 Errors in Aspect Graphs: Related Work

This section first explains the concept of an 'error' in the context of aspect graphs. An error is a non-conformance of the observed feature data at a position in the viewpoint space with the one predicted by the aspect graph. This could be the result of the inability of an exact aspect graph to model all observable details of an object, or the effect of noise in the feature detection process. Algorithms for aspect graph construction from CAD data generally do not address the issue of errors in aspect graphs. For analytical approaches, one requires very precise models of not only the noise process, but also the sensors and detectors, and the imaging process. In the following sections, we examine some important approaches in the area.

Feature Detector Detectability and Reliability

Ikeuchi, Kanade and co-workers define the terms detectability and reliability for a feature detector [109], [110], [93], [111], [177]. In [110] they derive analytical expressions for two cases – a light stripe range finder, and a photometric stereo detector. However, no related work (including the work of Gremban and Ikeuchi on object recognition [94]) uses such models for analyzing errors in AAGs.

PREMIO and related approaches

Lu, Shapiro and Camps [136] consider a very restricted class of feature detection errors in [136]. They assume the only variants of a perfect line drawing of a view class to be those with at most two internal line segments missing. Further, they assume the presence of all boundary line segments, and the presence of no extra line segments. The system PREMIO considers incorporates a frequency statistics-based concept of feature detectability. PREMIO's feature prediction module also uses feature detectability estimates to a predicted view of an object, given a set of sensor and lighting conditions.

The Scale Space Aspect Graph

The scale space approach of Eggert *et al.* [73], [74] and Pae and Ponce [154] model some types of errors through the notion of scale. The modeling of scale is with respect to geometric features alone. Further, the techniques for some simple examples are not easy to generalize for more complicated shapes.

View Degeneracy

Wilkes, Dickinson and Tsotsos [206], [61] show that degenerate views occupy a significant fraction of the viewing space surrounding an object. Accidental alignment of parts causes view degeneracies. The finite resolution of cameras and feature extracting operators enlarges the set of such erroneous viewpoints. The authors present a computational model for view degeneracy. They assume a perspective projection model. They assume geometric features, and assume the knowledge of camera parameters such as the focal length and the nodal point.

View Likelihood and Stability

Weinshall and Werman [203] define two measures on views: view likelihood and view stability. View likelihood measures the probability that a certain view of a

given 3-D object is observed. View stability measures how little the image changes as the viewpoint is slightly perturbed. They analytically derive the stability and likelihood measures for two feature-based 2-D metrics. The authors analyze this case for geometric features, with a calibrated setup. Further, the authors do not account for noise in the image formation process.

Finite Resolution Aspect Graphs

The work of Shimshoni and Ponce [182] is probably the only work in exact aspect graphs which account for finite-resolution effect – a cause of error in aspect graphs. The authors present an algorithm for constructing aspect graphs of polyhedral objects with an orthographic camera with limited spatial resolution.

3.2 The Proposed Approach: Motivation and Rationale

In this chapter, we propose a new approach to aspect graph construction using noisy sensors. Rather than using the output of a CAD modeling scheme, we use image-based information to construct an AAG. We consider simple image-based features to characterize a class. We assume the availability of an active sensor. (For our experimentation, we use an active orthographic camera.) We tessellate the viewing space around an object. We use the active sensor to collect data from all viewpoints in the tessellated viewing space. These sensors could be noisy. This leads to the detection of an incorrect class, at a viewpoint. This data ('raw aspect data') is the input for our AAG construction algorithm.

We first present a classification of all types of errors in raw aspect data. We look at errors from the point of view of the raw aspect data available with us. We examine the statistical nature of the occurrence of these errors. Our AAG construction algorithm eliminates errors from raw aspect data in an unsupervised manner. The algorithm

maintains estimates of these errors – we use this to advantage to impart robustness to our object recognition algorithm (Chapter 4), which uses the same set of sensors.

In this section, we present the motivation and rationale behind our approach. We base this on the following issues:

- The work on exact aspect graphs concentrates solely on their construction. To quote Flynn and Jain [83]:

  > "... the current work on "formal" aspect graphs has emphasized their construction from geometric models, rather than their use in recognition."

  Exact aspect graphs use topological information for defining aspects. However, it is difficult to extract topological information from images. For example, to quote Mundy in [24],

  > "On the one hand, these topological structures are well defined and are well developed within the mathematical literature. On the other hand, there is little reason to believe that topological relations can be reliably retrieved from an image, even without considering occlusion."

- Nearly all object recognition systems which use aspect graphs (or related models) with *simple image features*.

  1. Hutchinson and Kak [106] use faces of a polyhedral object which are segmented from range images.
  2. Ikeuchi and Kanade [109], [110] define aspects based on visible faces with photometric stereo. Suppose there are n faces S_1, S_2, ... S_n, where one face corresponds to either a plane surface, or a curved surface which will be detected as a single surface patch in photometric stereo. They use a $0-1$ variable X_i to have a value 1 if face S_i is visible and 0, otherwise. They use an n-tuple $(X_1, X_2, \ldots X_n)$ to represent an aspect.

3. Sato, Ikeuchi and Kanade [177] use information about specularities.
4. Gremban and Ikeuchi [94] use the following as features for their recognition system (this has one rotational degree of freedom between the object and the sensor):
   - the number of specularities,
   - the area of the largest specularity,
   - the dominant eigenvalue of the largest specularity,
   - the maximum distance between specularities, and
   - the minimum distance between specularities

   For their example using a finger-gap sensor, they use the distance between the fingers of a parallel jaw gripper, as a feature.
5. Chakravarty and Freeman [40] use the 8 types of junctions proposed in [39]. As features, they use a feature vector, whose components are the number of junctions of each type.

PREMIO [36], [37], [34] , and the work of Dickinson *et al.* [58], [60], [59], [56], [57] are probably the only systems which derive topological information from an image. However, their method suffers from the overhead of having to match two hierarchical topological descriptions. The authors perform this relational matching problem as an IDA^* search on an interpretation tree. [56], [57] additionally use an aspect prediction graph, which is based on an aspect graph. Further, the work on PREMIO shows experimental results with only two polyhedral objects, *Cube3Cut* and *Fork*. It is not very clear how these relational matching procedures perform under noise, and feature detection errors.

Dickinson and co-workers use information from an aspect graph to construct an *Aspect Prediction Graph*. There are two important shortcomings of their system. The first is the use of volumetric primitives. It is not very easy to extract knowledge about volumetric primitives from images, reliably. Secondly, their

object recognition strategy (which uses the aspect prediction graphs) has to track the region of interest through successive views. In addition to the tracking overhead, the system performs many redundant image processing operations – in many successive images, when the aspect of the object remains unchanged.

- An aspect graph is often used for recognizing a given object from its image. In such a case, factors such as noise and non-adaptive thresholds may affect the output of a feature detector at a position in viewpoint space. While authors account for effects such as finite resolution and varying the distance between the object and the sensor, no approach addresses the above issue. During the course of AAG construction, such effects may be directly observed. Thus, it is possible to get information about noise and other feature detection errors from observed data directly.

- *In general, it is difficult to find models that model the sensing and imaging processes very accurately.* For example, in spite of the elaborate vision model in PREMIO, Camps mentions [34]:

  > "At times, edges that do not correspond to any boundary or limb can appear in an image due to a particular material property or lighting configuration. Those edges represent intensity discontinuities caused by shadows or highlights in shiny surfaces and can be derived from the surface characteristics. However, incorporating this feature into the prediction module constitutes a major research effort ... "

Error-modeling approaches for exact aspect graphs explicitly aim at modeling some causes of errors – either along aspect boundaries, or finite resolution, or accidental alignments. Though there has been a significant amount of work on sensor modeling by proponents of the uniform partitioning approach – sensor detectability and reliability ([109], [110], [93], [111]), no related work accounts for feature detection errors in aspect graph construction. The authors in [94]

mention only one type of error – those on aspect boundaries. No approach – either for exact aspect graphs or AAGs, accounts for the effects of noise and related feature detection errors in aspect graph construction.

The organization of the chapter is as follows: Section 3.3 examines different issues in AAG construction. We then present our classification scheme for different types of errors in AAGs, in Section 3.4. We present our algorithm for aspect graph construction in Section 3.5, as well as our evaluation function for comparing aspect graph construction algorithms. Section 3.6 discusses the suitability of a feature detector for aspect graphs, and related issues. We present results of extensive experimentation in Section 3.7. In the concluding section, we summarize the salient features of our scheme.

3.3 AAGs, Errors in Raw Aspect Data

Any object recognition strategy requires robust identification of a view of the given object. The success of such a strategy crucially depends upon its ability to handle feature detection errors – the ability of the aspect graph construction algorithm to model and account for such errors, as well as the mechanism in the view-identification algorithm to handle them. This chapter presents a new algorithm for aspect graph generation with active sensors. We show its applicability for robust 3-D object recognition in Chapter 4.

With the uniform partitioning approach, construction of an AAG requires visiting each site in the tessellated viewing space around the object. In the most general case, there may be six degrees of freedom between the object and the sensor – three each for translation and rotation. In this thesis, we consider the 1-DOF and 3-DOF cases. In the 1-DOF case, the sensor can move around the object in a circle, at a fixed distance. Figure 3.1(a) shows an example of the 1-DOF case, along with an image taken by the (orthographic) camera. The viewing space is a tessellated circle

(Figure 3.3(a), shown flattened out in Figure 3.3(b)). Figures 3.3(c) and 3.3(d) depict two ways of representing a 1-DOF aspect graph. Figure 3.3(c) shows a Gantt chart - each aspect of the object is represented by a shaded rectangle, proportional to its angular extent on the flattened-out perimeter of the viewing circle. Different shading patterns represent different classes. We may also represent an aspect graph as a plot of the class at each viewing position. The **X**-axis represents different viewing positions on the flattened-out perimeter of the viewing circle, while the non-metric **Y**-axis represents different classes as different heights. Figure 3.3(d) shows an AAG of the example object represented as a class-distribution. In this example, tessellated sites are 3^o apart. We use a circular linked list as the data structure to represent an AAG. Each node (an aspect) stores information such as its associated class and its angular extent.

Figure 3.4 shows the corresponding examples for the 3-DOF case. The viewing space is a tessellated sphere (Figure 3.4(a)). We use the tessellation algorithm of Chen and Kak [41]. The authors claim their method to be superior to other existing methods – [102], [108], [80], [119] and [91]. To obtain a geodesic tessellation, we inscribe an icosahedron in a unit sphere. Each edge of a triangular face is divided into Q sections (Q is the frequency of geodesic division). Finally, the subdivided faces are projected onto the unit sphere. Each of the $10 \cdot Q^2 + 2$ vertices represents a viewpoint. Chen and Kak present a scheme to represent a vertex as $[i, j, k]$ (Figure 3.4(b)) and give rules to find the neighbours of each vertex. Vertices which correspond to those of the original icosahedron have 5 neighbours, whereas the rest have 6 each. They express the direction of each tessel in terms of latitude and longitude angles, $\psi(i, j, k)$ and $\phi(i, j, k)$. The radial angle between adjacent sampling points ('inter-site angular distance') is roughly $atan(2)/Q$. We use the above scheme to compute the distance between two sites A and B in units of the inter-site distance, as follows (Figure 3.5).

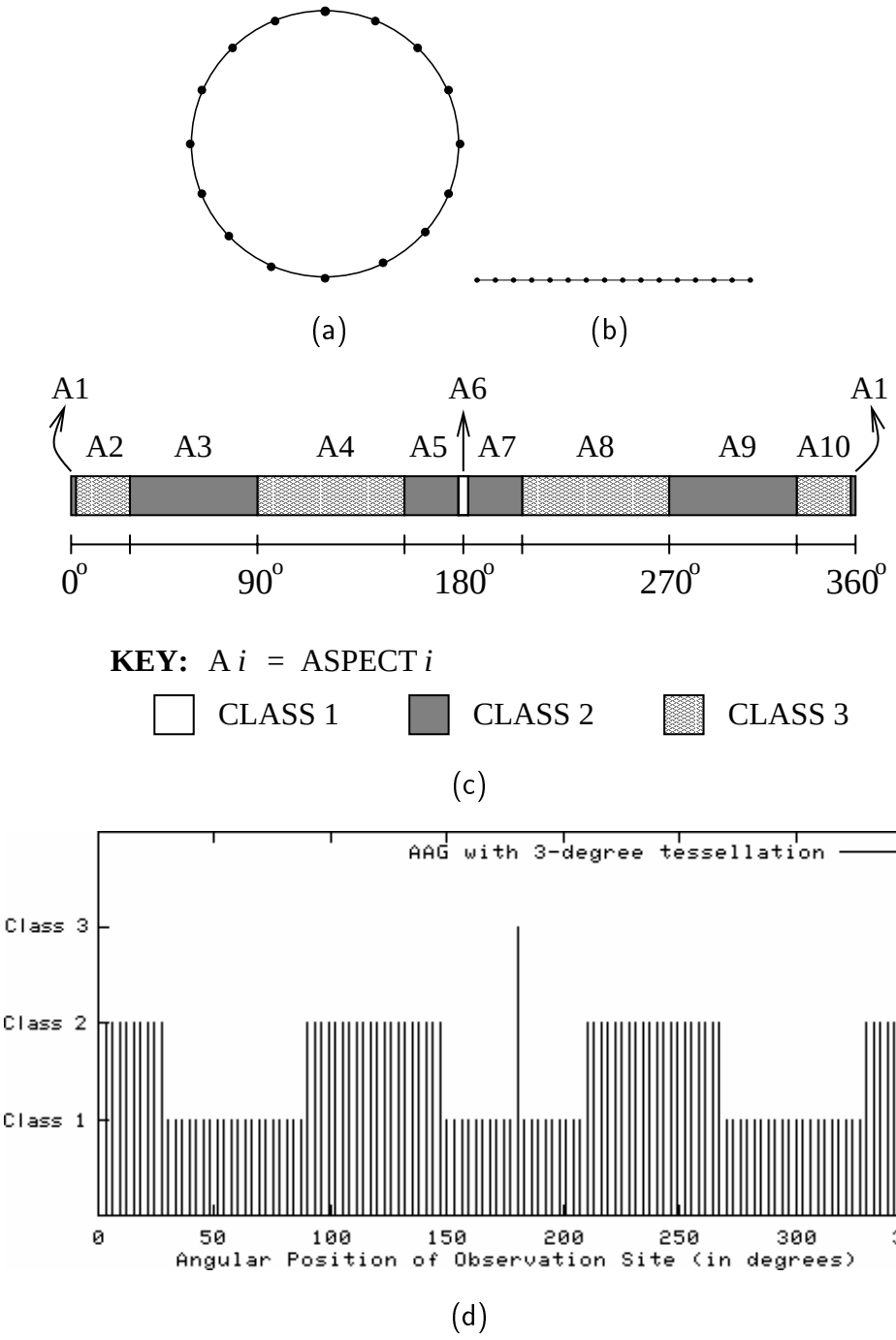


Figure 3.3: 1-DOF viewpoint space tessellations and aspect graphs: (a) The tessellated viewing circle, and (b) its flattened-out representation; and corresponding to the object in Figure 3.1, (c) a Gantt chart representation of the aspect graph, and (d) an AAG of the object, shown as a class-distribution

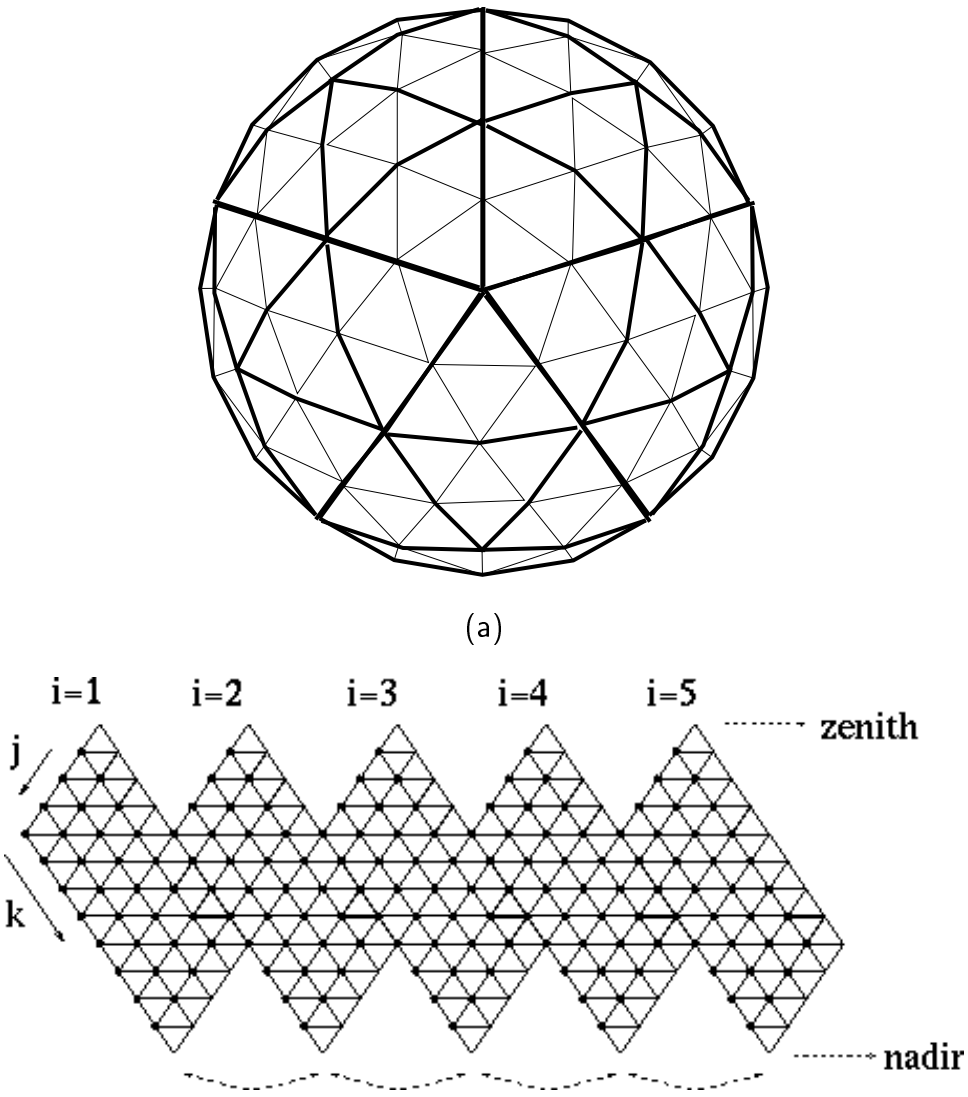


Figure 3.4: 3-DOF viewpoint space tessellations: (a) The tessellated viewing sphere, and (b) its flattened-out representation

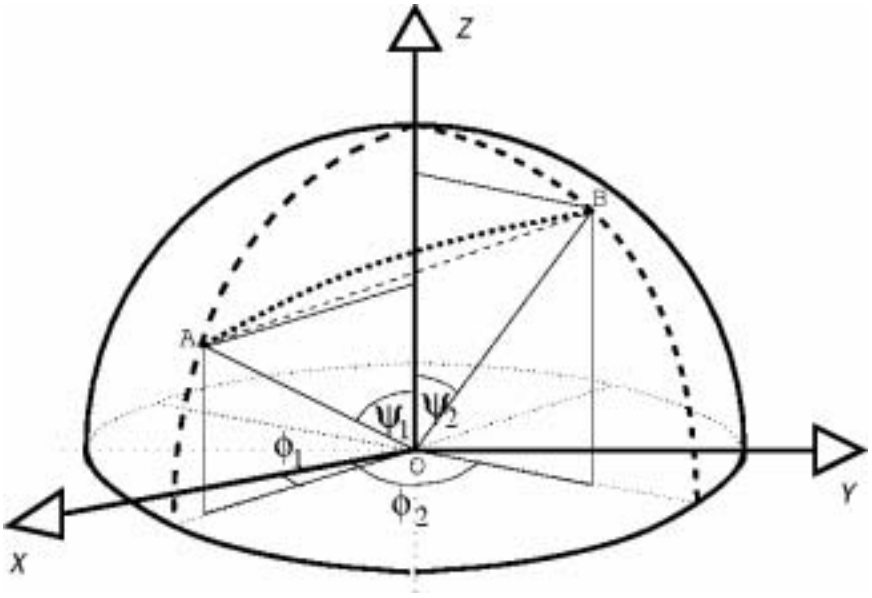


Figure 3.5: Distance between two tessellated points: the 3-DOF case (details in text)

Since $cos\angle AOB$ is $\vec{OA} \cdot \vec{OB}$,

$$l_{A,B} \approx (Q/atan(2)) \cdot arccos(sin\psi_1\ cos\phi_1\ sin\psi_2\ cos\phi_2\ + \\ sin\psi_1\ sin\phi_1\ sin\psi_2\ sin\phi_2 + cos\psi_1\ cos\psi_2)$$

We use a weighted graph as the data structure to represent a 3-DOF AAG (Figure 3.6). Each node represents an aspect. An adjacent node is one with which it shares a common boundary (in the tessellated viewing space). The weight of the link between them is the length of this common boundary.

3.3.1 Raw Aspect Data, Aspect-Candidates, Class-Candidates

Let us assume a set of n feature detectors for the set of features $\mathbf{F} = \{F_1,\ F_2,\ \ldots\ F_n\}$. We use this set of feature detectors on the image of the object taken at a site in the viewpoint space. We denote the output of the set of feature detectors as an n-tuple, $\langle\ f_{1j_1},\ f_{2j_2},\ \ldots\ f_{nj_n}\ \rangle$. Thus, each site in the viewpoint space is associated with a feature tuple label. Let the term **'Raw aspect data'** denote the collection of feature tuples obtained at the set of sites in the tessellated viewing space.

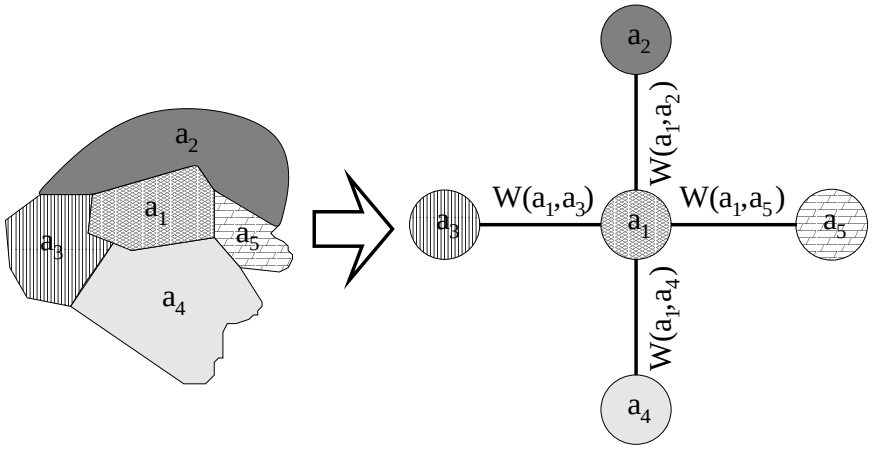


Figure 3.6: Representation of a 3-DOF AAG: for aspect a_1

For a given set up with an object and suitable sensors, uncertainty in sensor measurement can arise due to variation in brightness values (intensity-based sensor) and variation in light source directions (position-based sensor) [110]. Errors can also arise due to physical characteristics of not only the sensor, but also the illumination source (for light-based feature detection) and the digitization process. Non-adaptive thresholds in algorithms for processing sensor data are also a cause of feature detection errors. Thus, factors such as noise and non-adaptive thresholds may produce an incorrect set of features at a viewpoint.

We refer to aspects and classes obtained from raw aspect data as **aspect-candidates** and **class-candidates**, respectively. A feature tuple $\langle\ f_{1j_1},\ f_{2j_2},\ \ldots\ f_{nj_n}\ \rangle$ characterizes a class-candidate. (As an example, f_{ij_i} could represent the number of junctions of a particular junction type: V-junction, T-junction, etc. Chakravarty and Freeman [40] use this method to represent a view class.) A class-candidate is related to the set of features describing an aspect-candidate. An aspect-candidate has a class-candidate associated with it, along with information about its corresponding set of points in the viewpoint space around the object. Thus, we can have erroneous aspect-candidates and class-candidates, but no erroneous aspects and classes.

The following section presents a classification of errors in raw aspect data.

3.4 A Classification of Errors in Raw Aspect Data

In this section, we characterize different types of errors that can occur in raw aspect data – due to noise in the sensing process, and errors in feature detection. First, we present a general discussion on errors in AAGs. Subsequently, we formalize this in our classification scheme. We perform this analysis for both 1-DOF and 3-DOF cases. We assume that all the objects in the given model base, and feature detectors are suitable for aspect graph-based object recognition. Loosely speaking, the object should have aspects which are not too small in size, or too large in number. If aspects are very small in size, it may be difficult to tell them apart from errors. Too many aspects increases the memory requirements for an object recognition task – in the search for a view that distinguishes between different objects having many views in common.

An error-free AAG is characterized by piecewise continuity in the feature tuple labels at its sites. Such an AAG has aspects, whose corresponding angular extents are not too small in size. The discontinuities in the smoothness correspond to aspect boundaries. We base our error classification on experimental observations – the feature data obtained with an active sensor. Inherent in our classification scheme is the statistical nature of occurrence of errors in the raw aspect data. Intuitively, an error corresponds to a small region whose feature tuple label is different from the labels in its neighbourhood. (We use the notation $\mathcal{G}_\alpha$ to denote the set of aspect-candidates in the neighbourhood of aspect-candidate α.) The following two factors form the basis of error characterization in an AAG:

- the position (site) at which the error is introduced, and
- the 'value' or feature tuple label at that site.

At this point, we define the term **Valid class-candidate**. Given a set of raw aspect data, let us consider aspect-candidates with large extents (in terms of the number of sites they occupy in viewpoint space). These are more likely to either constitute an

aspect by themselves, or be part of some aspect in the corresponding error-free aspect graph. We would like to identify those class-candidates which correspond to aspects in an error-free aspect graph. We characterize a valid class-candidate in terms of a parameter N_{min}:

N_{min} : the minimum total number of sites at which a class-candidate should be present in an object's viewpoint space to be called a 'Valid class-candidate'

At the end of our AAG construction algorithm (Sections 3.5.3 and 3.5.4 for the 1-DOF and 3-DOF case, respectively), the only class-candidates left in the aspect data are all valid class-candidates. These constitute the aspect-classes for the resultant AAG. An aspect-candidate having a small extent constitutes an error. However, its corresponding class-candidate may not necessarily be an erroneous class-candidate. As an example, there may be other aspect-candidates in its neighbourhood corresponding to the same class-candidate. Thus, these aspect-candidates may be a part of an aspect in the corresponding error-free aspect graph. Such an aspect could correspond to a unique class, that is not present in any other object in the model base. If the number of sites corresponding to the class-candidate is not less than N_{min}, it will not be treated as an erroneous class-candidate.

In what follows, we formally present our classification scheme. We give a mathematical characterization of each type of error. Let us use the terms $\mathcal{A}$ and $\mathcal{C}$ to denote the set of all aspect-candidates and the set of all class-candidates, for a particular model base, respectively. We define the following function:

$$CLASS\_CAND : \mathcal{A} \rightarrow \mathcal{C}$$

Thus, for an aspect-candidate α, the function $CLASS\_CAND(\alpha)$ returns its corresponding class-candidate. We use the notation $\mathcal{A}_c$ to denote the set of all aspect-candidates corresponding to class-candidate c. Formally,

$$\mathcal{A}_c \triangleq \{\ \alpha\ \in \mathcal{A} \mid CLASS\_CAND(\alpha) = c, \qquad c \in \mathcal{C}\}$$

Association Errors

An important point here is that due to characteristics of the particular experimental setup – sensor response characteristics, sensor positioning, lighting arrangements, the imaging process and the feature detection mechanism, a particular type of error may be *associated* with a particular class-candidate. In other words, some erroneous class-candidate c may occur as error regions exclusively in regions corresponding to a particular class c' of an exact aspect graph. We term this phenomenon as an **association error**. To illustrate an example of such an association, let us consider a 1-DOF setup, with a camera going round a polyhedral object in a tessellated circle. A light source is placed just on top of the camera. Let us consider an edge of the polyhedral object and the two plane faces meeting at the edge. Let δ denote the angle at which light is incident on the edge. When δ is not small, these two faces are illuminated differently. In the image, there is a prominent discontinuity in the grey levels corresponding to the object edge. Hence, it is possible for an edge detector to easily identify an edge in the image. However, when δ is close to zero, the two faces are illuminated almost uniformly. Thus, an edge detector fails to detect the corresponding edge in the image. Let us assume the corresponding aspect to have class c'. The range of viewing positions where δ is nearly zero will have one edge less – this corresponds to another class c. Thus, this class c is *associated* with class c'. The situation is the same for aspect-candidates and class-candidates.

We use the notation $P(c'_{actual} \mid c_{observed})$ to denote the probability of the class-candidate actually being c', given that class-candidate c has been observed, for the given model base. For such an association error, $P(c'_{actual} \mid c_{observed})$ is expected to have a high value. (For a valid class-candidate c', $P(c'_{actual} \mid c'_{observed})$ has a high value.) In such cases, we use this information to advantage in two ways. First, this gives an indication about the correct class corresponding to the region – during AAG construction. We need to prune out such regions from the aspect data in order to construct an error-free AAG. Our AAG construction algorithm (Sections 3.5.3 and

3.5.4 for the 1-DOF and the 3-DOF cases, respectively) makes estimates of these probabilities during the course of processing the raw aspect data. *It is important to note that these are unsupervised estimates.* Second, an object recognition algorithm may use this information to advantage, to recover from such a feature detection error.

3.4.1 The 1-DOF Case

For the 1-DOF case, the space of viewpoints is a circle. Angular extents corresponding to aspects are all planar angles. We define the following terms:

θ_α : angular width of aspect-candidate α, in terms of the number of sites it occupies

Θ_{min} : the minimum extent which an aspect-candidate must have, for it to be called a **Valid aspect-candidate**

Θ_p : the minimum extent which an aspect-candidate must have, for it to be called a **Prominent aspect-candidate** ($\Theta_p \geq \Theta_{min}$)

$\mathcal{A}_c^g \stackrel{\triangle}{=} \{\ \alpha \mid \theta_\alpha \geq \Theta_{min}$, where $\alpha \in \mathcal{A}_c\}$

We explain the above terms, and their significance as follows. Three parameters characterize an aspect-candidate α – its corresponding class-candidate $CLASS\_CAND(\alpha)$, its angular extent θ_α, and its position in the aspect graph. An aspect-candidate can either correspond to a part of an aspect in an error-free aspect graph, or constitute an error. Intuitively, a small region whose class-candidate is different from those in its neighbourhood is likely to be an error. Thus, we characterize an aspect-candidate as 'valid' if its angular extent is greater than or equal to Θ_{min}. Any aspect-candidate whose angular extent is less than Θ_{min} corresponds to an error.

There is a difference between the terms 'aspect' and 'prominent aspect-candidate'. At the end of the aspect graph construction algorithm, all aspect-candidates left are aspects. At this stage, a prominent aspect-candidate will either constitute an aspect by itself, or have other aspect-candidates (prominent, or otherwise) integrated into it to constitute an aspect. In other words, a prominent aspect-candidate is sure to be part of an aspect at the termination of the AAG construction algorithm.

The set $\mathcal{A}_c^g$ is another parameter for a class-candidate c, on the basis of which we may infer whether it is a **valid class-candidate**, or an erroneous one. $\mathcal{A}_c^g$ is the set of valid aspect-candidates corresponding to this class-candidate. If a class-candidate c is *not* a valid class-candidate, it satisfies the following two conditions:

1. $\sum_{\alpha \in \mathcal{A}_c} \theta_\alpha < N_{min}$, where $CLASS\_CAND(a_k) = c$, and
2. $\mathcal{A}_c^g = \phi$

In other words, it corresponds to less than N_{min} sites (Section 3.4, page 72). Further, it has no valid aspect-candidate corresponding to it.

Here, we clarify the significance of having two separate parameters N_{min} and Θ_{min}. Any aspect-candidate α having a corresponding angular extent $\theta_\alpha \leq \Theta_{min}$ is an error. Its corresponding class-candidate may not always be an erroneous class-candidate. Let us consider an object in the model base, which has an aspect corresponding to a unique class-candidate (c, say) – which is not present in any other object in the model base. If detected, this aspect uniquely identifies the object. Noise may affect the aspect data corresponding to this aspect in such a way that there are many closely-spaced fragments (aspect-candidates) with class-candidate (c). The angular extent of each of these aspect-candidates may be less than Θ_{min}. However, the presence of closely-spaced fragments having the same class-candidate is an indication of the presence of an aspect in the corresponding error-free aspect graph. If the number of sites at which this unique class-candidate occurs exceeds N_{min}, our algorithm (Section 3.5.3) is able to recover the aspect.

We explain the significance of these terms further in our classification scheme. We classify errors in aspect data into five categories. The classification is based on the width of the region $\mathcal{R}$ between two valid aspect-candidates, their properties, and the properties of the (non-valid) aspect-candidates present in region $\mathcal{R}$.

Type I Error A Type I error is present as a small transition region between two aspects of the corresponding error-free aspect graph. This corresponds to the

"border effect" in [94]. In the raw aspect data, a Type I error can be described as follows: (Figure 3.7(a) illustrates an example of this type of error)

$$\theta_{a_i}, \theta_{a_j} \geq \Theta_{min}$$

$$CLASS\_CAND(a_i) \neq CLASS\_CAND(a_j), \text{ and}$$

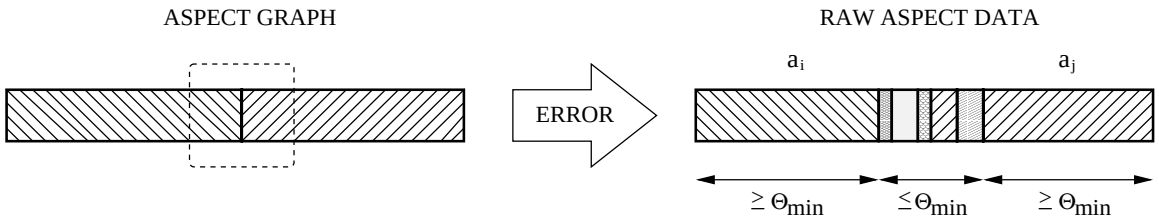
$$\sum_{a_k} \theta_{a_k} < \Theta_{min}, \ \forall \ a_k \text{ between } a_i \text{ and } a_j$$

Here, a_i and a_j are two valid aspect-candidates belonging to different class-candidates such that there is a small region of width $\leq \Theta_{min}$ between them. For taking aspect-candidates between two other aspect-candidates lying on the circle, we consider the smaller gap between the two.

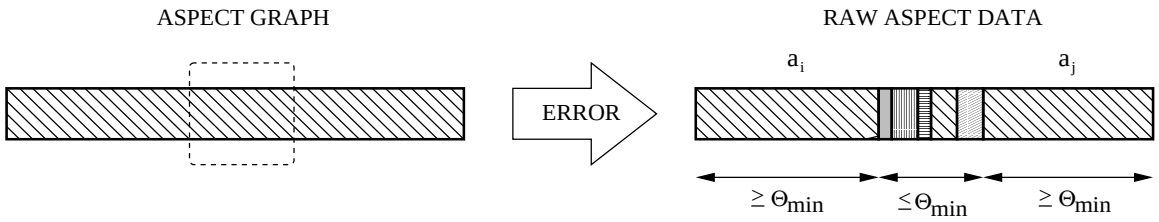
Type II Error A Type II error corresponds to the **association error** in Section 3.4. First, we characterize the region in which a Type II or Type III error can lie. Type II and Type III errors can be present in a small isolated error region inside an aspect of the corresponding error-free aspect graph. Figure 3.7(b) shows an example of such a region. In the raw aspect data, we observe a small region of width $\leq \Theta_{min}$ in between two valid aspect-candidates a_i and a_j. The two enclosing aspect-candidates a_i and a_j correspond to the same class-candidate (say, c'). Formally, we may describe the above error conditions as follows:

1. $\theta_{a_i}, \theta_{a_j} \geq \Theta_{min}$
2. $\sum_{a_k} \theta_{a_k} < \Theta_{min} \ \forall \ a_k$ between a_i and a_j
3. $CLASS\_CAND(a_i) = CLASS\_CAND(a_j) = c'$

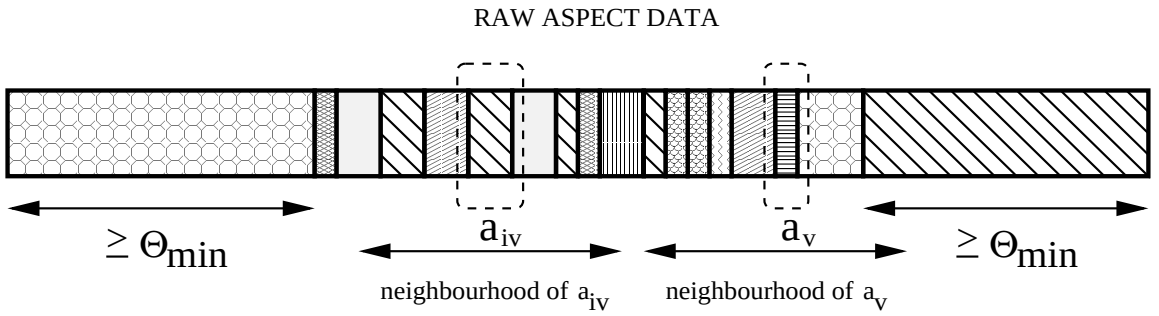
We now define a Type II (association) error. For the small enclosed error region, those aspect-candidates constitute a Type II error, whose class-candidates observed are associated with the class-candidates of the enclosing aspect-candidates. Let us consider aspect-candidates a_k in the small enclosed region (described above). We may describe this as follows:



(a) Type I Error



(b) A Region having Type II and III Errors (details in text)



a_{iv} AND a_v ARE EXAMPLES OF TYPE IV & TYPE V ERRORS, RESPECTIVELY

(c) Types IV and V Errors

Figure 3.7: A pictorial representation of some types of errors (Different shading patterns represent different aspect candidates): the 1-DOF case

1. $\sum_{\alpha \in \mathcal{A}_c} \theta_\alpha < N_{min}$, where $CLASS\_CAND(a_k) = c$
2. $\mathcal{A}_c^g = \phi$, and
3. $P(c'_{actual} \mid c_{observed}) \geq$ a threshold T_1, $0 < T_1 < 1$

($\mathcal{A}_c$ represents the set of all aspect-candidates corresponding to class-candidate c, and $\mathcal{A}_c^g$ represents the set of all *valid* aspect-candidates corresponding to this class-candidate.)

We explain the above three error conditions as follows. For an aspect-candidate a_k to constitute a Type II error, its corresponding class-candidate c should not be a valid class-candidate. In other words, the class-candidate c is observed at less than N_{min} sites in the tessellated viewing space. Further, the relative extent to which its corresponding valid aspect-candidates (if any) occupy the viewpoint space is also low. The most important condition for a Type II error is the third one – given that this class-candidate c is observed, the probability that it actually is another class-candidate c', is high. Our algorithm for AAG construction (Section 3.5.3) describes the process of keeping estimates of association error probabilities.

A particular object in the model base may have a distinctive feature, or a combination of features that permit it to be clearly distinguished from other models. Additionally, this may be present over a small range of viewing angles. Conditions 2 and 3 taken together allow for such cases to be treated as aspects, and not as errors. For a distinctive feature, one would expect either the total number of sites at which the class-candidate is observed, to be greater than the N_{min} threshold; or the class-candidate to have most of its aspect-candidates with extent not less than Θ_{min}. Both the above conditions may also be simultaneously true. If none of these conditions hold, the distinctive aspect will be treated as an error condition.

Type III Error A small isolated error region inside a region corresponding to an

aspect of an error-free aspect graph (as defined above) may contain either Type II or Type III errors. For a Type III error, the only requirement is $P(c'_{actual} \mid c_{observed}) < T_1$, *i.e.*, the probability of recognizing class candidate c as c' is minimal.

Type IV Error Due to noise, an aspect of an error-free AAG could be fragmented such that the raw aspect data has closely spaced aspect-candidates with the same class-candidate (Figure 3.7(c) shows an example of this situation). Such regions contain Type IV and Type V errors. Formally, Type IV and Type V errors can be present in a 'large' region between two valid aspect-candidates:

1. $\theta_{a_k} < \Theta_{min}$, $\forall\ a_k$ between a_i and a_j
2. $\sum_{a_k} \theta_{a_k} \geq \Theta_{min}$

In other words, no aspect-candidate a_k in the region is a valid aspect-candidate. However, the size of the entire region $\geq \Theta_{min}$.

As aspect-candidate α constitutes a Type IV error if the following error conditions hold:

1. $\max_{\beta \in \mathcal{A}_c} \theta_\beta \ \geq \Theta_{min}$, where $CLASS\_CAND(\alpha) = c$
2. $\exists \gamma \in \mathcal{G}_\alpha$ for which $CLASS\_CAND(\gamma) = CLASS\_CAND(\alpha)$

(The term $\mathcal{A}_c$ denotes the set of all aspect-candidates corresponding to class-candidate c. $\mathcal{G}_\alpha$ denotes the set of all aspect-candidates in the neighbourhood of aspect-candidate α.)

This type of error considers those aspect-candidates which themselves do not have enough extent to be considered valid aspect-candidates. However, there are other valid aspect-candidates associated with this class-candidate, elsewhere. In Figure 3.7(c), aspect-candidate a_{iv} shows an example of a Type IV error. A Type IV error indicates that the aspect-candidates with the same class-candidate may be part of a single aspect.

Type V Error Like Type IV errors, Type V errors too can be present in a 'large' region between two valid aspect-candidates (as defined above). An aspect-candidate α constitutes a Type V error if either

- $\forall \gamma \in \mathcal{G}_\alpha,\ CLASS\_CAND(\gamma) \neq CLASS\_CAND(\alpha)$, or
- $\exists \gamma \in \mathcal{G}_\alpha$ for which $CLASS\_CAND(\gamma) = CLASS\_CAND(\alpha)$, but $\max_{\beta \in \mathcal{A}_c} \theta_\beta \ < \Theta_{min}$, where $CLASS\_CAND(\alpha) = c$

In Figure 3.7(c), aspect-candidate a_v illustrates an example of a Type V error. Type V errors are very difficult to correct since they give very little indication as to which aspect (in the corresponding error-free aspect graph) they might have come from.

3.4.2 The 3-DOF Case

Similar to the 1-DOF case, we classify errors in the 3-DOF case into five categories. An error region $\mathcal{R}$ consists of either a single small aspect-candidate or a group of adjacent small aspect-candidates. In each of the Figures 3.8(a), (b) and (c), the region $\mathcal{R}$ consists of those aspect-candidates which are fully enclosed inside the boundary drawn with a thick dashed curve. The collection of small adjacent aspect-candidates $\mathcal{R}$ is like a 'supernode', in the graph representation of an AAG (Section 3.3). It stores information such as its boundary sites, number of sites, and links to other valid aspect-candidates.

Some definitions for the 3-DOF case are completely analogous to the 1-DOF case (Section 3.4.1). For the 3-DOF case, ω_α denotes the number of sites aspect-candidate α occupies. The terms corresponding to Θ_{min} and Θ_p in the 3-DOF case are Ω_{min} and Ω_p.

For the error classification, we need to introduce a few more terms. Section 3.3 mentions that we follow the tessellation scheme of Chen and Kak [41]. This section additionally gives details of the tessellation scheme. We recall that each site in the

tessellated space has either 5 or 6 neighbours (Page 66). (Vertices which correspond to those of the original icosahedron have 5 neighbours, whereas the rest have 6 each.) We use the term $NEIGH(s)$ to denote the set of neighbours of site s. In the 3-DOF case, each aspect-candidate α corresponds to a region of sites in the viewpoint space. We define $\mathcal{B}(\alpha)$ to be the set of sites on the *boundary* of aspect-candidate α. In this context, we define the following terms:

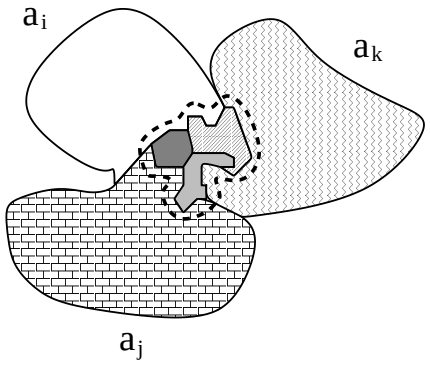
$$
\begin{aligned}
&W(\alpha_1, \alpha_2) \stackrel{\triangle}{=} \min(|S_1|, |S_2|), \text{where } |S_i| \text{ denotes the cardinality of set } S_i, \\
&\quad S_1 = \{s \mid s \in NEIGH(s'), s' \in \mathcal{B}(\alpha_1)\} \cap \mathcal{B}(\alpha_2), \text{and} \\
&\quad S_2 = \{s \mid s \in NEIGH(s'), s' \in \mathcal{B}(\alpha_2)\} \cap \mathcal{B}(\alpha_1) \\
&ADJ(\alpha) \stackrel{\triangle}{=} \{\beta \mid W(\alpha, \beta) \neq 0\}, \textit{i.e.}, \text{ the set of nodes linked to } \alpha \\
&DIST(\alpha_1, \alpha_2) \stackrel{\triangle}{=} \min l_{s',s''}, \ \forall s' \in \mathcal{B}(\alpha_1) \text{ and } s'' \in \mathcal{B}(\alpha_2)
\end{aligned}
$$

Here, $l_{s',s''}$ is the distance between two sites in the tessellated viewpoint space (Section 3.3, page 69).

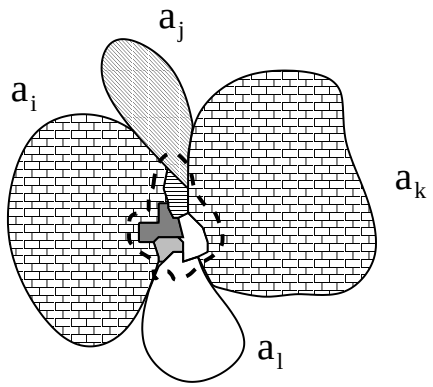
An error free 3-DOF AAG is characterized by piecewise continuity (in two dimensions, in the flattened out spherical array representation) in the feature tuple labels. Intuitively, a region of small and mutually adjacent aspect-candidates constitutes an error. The region itself could be large or small. Further, just like the 1-DOF case, we could have association errors. The classification considers the size of the error region, the properties and statistics of the aspect-candidates comprising the error region, and those aspect-candidates with which the region shares a common boundary.

Type I Error A Type I Error can be present as a small transition region between aspects of an error-free aspect graph (Figure 3.8(a) illustrates an example of a Type I error). A small region $\mathcal{R}$ constitutes a Type I error if the valid aspect-candidates having the same class-candidate do not share a large common boundary:

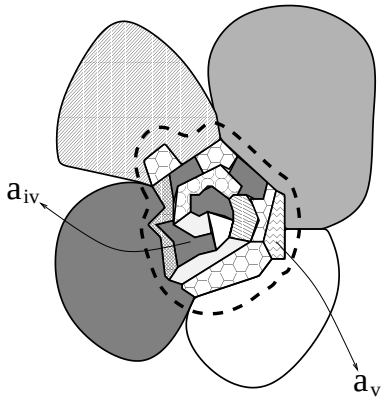
1. $\omega_\alpha \geq \Omega_{min}, \ \forall \alpha \in ADJ(\mathcal{R})$,



(a) Type I Error



(b) A Region having Type II and III Errors (see text)



(c) Types IV and V Errors

Figure 3.8: A pictorial representation of some types of errors(Different shading patterns represent different aspect candidates): the 3-DOF case. The thick dashed curve encloses aspect-candidates in the region $\mathcal{R}$ (details in text).

2. $\sum_{\beta \in \mathcal{R}} \omega_\beta < \Omega_{min}$, and

3. $\frac{\sum_{\alpha \ with \ the \ same \ class-candidate} W(\mathcal{R},\alpha)}{\sum_{\alpha} W(\mathcal{R},\alpha)} <$ a threshold Υ, $\forall\ \alpha \in ADJ(\mathcal{R})$

Type II Error Type II and Type III errors can be present as a small isolated region in the interior of an aspect in the corresponding error-free aspect graph. In the raw aspect data, we observe a small region $\mathcal{R}$ enclosed by valid aspect-candidates. For Type II and III errors - most of the common boundary consists of valid aspect-candidates having the same class-candidate, say c'. The region in which a Type II and III errors can lie is described as (Figure 3.8(b)):

1. $\omega_\alpha \geq \Omega_{min},\ \forall \alpha \in ADJ(\mathcal{R})$,

2. $\sum_{\beta \in \mathcal{R}} \omega_\beta < \Omega_{min}$

3. $\frac{\sum_{\alpha:\ CLASS\_CAND(\alpha)=c'} W(\mathcal{R},\alpha)}{\sum_{\alpha} W(\mathcal{R},\alpha)} \geq \Upsilon,\ \forall\ \alpha \in ADJ(\mathcal{R})$

A Type II error is characterized by the association of the particular aspect-candidate $\alpha \in \mathcal{R}$ with the class-candidate c':

1. $\sum_{\xi \in \mathcal{A}_c} \omega_\xi < N_{min}$, where $CLASS\_CAND(\alpha) = c,\ \alpha \in \mathcal{R}$

2. $\mathcal{A}_c^g = \phi,$ and
3. $P(c'_{actual} \mid c_{observed}) \geq$ a threshold $T_1,\ 0 < T_1 < 1$

(The term $\mathcal{A}_c$ represents the set of all aspect-candidates corresponding to class-candidate c. Among these, $\mathcal{A}_c^g$ considers those that are valid aspect-candidates.)

Type III Error A Type III error is also present as a small isolated region inside a region corresponding to an aspect of an error-free aspect graph. For a Type III error aspect-candidate α however, $P(c'_{actual} \mid c_{observed}) < T_1$ (*i.e.*, there is no association of a class-candidate with another).

Type IV Error Type IV errors are scattered in a region corresponding to an aspect of an error-free aspect graph. In the raw aspect data, they can be present in a 'large' region $\mathcal{R}$ between valid aspect-candidates:

1. $\omega_\alpha < \Omega_{min}, \alpha \in \mathcal{R}$
2. $\sum_\alpha \omega_\alpha \geq \Omega_{min}$

As for the 1-DOF case, a Type IV error indicates that those aspect-candidates could be part of a single aspect of the corresponding error-free aspect graph:

1. $\max_{\beta \in \mathcal{A}_c} \omega_\beta \geq \Omega_{min}$, where $CLASS\_CAND(\alpha) = c$
2. $\exists \gamma \in \mathcal{G}_\alpha$ for which $CLASS\_CAND(\gamma) = CLASS\_CAND(\alpha)$

(We recall that $\mathcal{G}_\alpha$ denotes set of aspect-candidates in the neighbourhood of aspect-candidate α. $\mathcal{A}_c$ represents the set of all aspect-candidates with class-candidate c.) Figure 3.8(c) shows an example of such an error.

Type V Error Type V errors, too can be present in a 'large' region between valid aspect-candidates (as defined above for a Type IV error). An aspect-candidate α constitutes a Type V error if either

- $\forall \gamma \in \mathcal{G}_\alpha$, $CLASS\_CAND(\gamma) \neq CLASS\_CAND(\alpha)$, or
- $\exists \gamma \in \mathcal{G}_\alpha$ for which $CLASS\_CAND(\gamma) = CLASS\_CAND(\alpha)$, but $\max_{\beta \in \mathcal{A}_c} \omega_\beta < \Omega_{min}$, where $CLASS\_CAND(\alpha) = c$

Figure 3.8(c) shows an example of such an error. Just as mentioned for the 1-DOF case, Type V errors give very little indication as to which aspect (in the corresponding error-free aspect graph) they could have come from.

3.5 AAG Construction from Erroneous Raw Aspect Data

In this section, we present our algorithm for construction of an AAG from noisy raw aspect data. First, we describe the problem, and various related issues. Next, we present a new function for evaluating the output of different AAG construction algorithms. We present our algorithm for AAG construction for the 1-DOF, as well as the 3-DOF cases.

3.5.1 Statement of the Problem

We are given an instance of raw aspect data for each object in the given model base. The raw aspect data is obtained from all viewing positions in the uniformly tessellated viewpoint space around the object. Noisy feature-detectors are used to collect the raw aspect data. The task at hand is to construct AAGs of the objects in the model base for robust class recognition. This problem is similar to one of data clustering. According to Jain and Dubes [112], there is no single "best" criterion for obtaining a partition for clustering data. We use clustering techniques for filtering the raw aspect data such that the distribution of aspects satisfies a smoothness criterion.

Let us define the terms 'smoothness' of model base data ($\mathcal{S}(A)$), and the total model base error ($\mathcal{E}(A)$) as follows:

$$\mathcal{S}(A) \triangleq (1/M) \cdot \sum_{i=1}^{M} \sum_{j=1}^{G} d(c_{ij}, c_{ij+1})$$
$$\mathcal{E}(A) \triangleq (1/M) \cdot \sum_{i=1}^{M} \sum_{j=1}^{G} d(c_{ij}, D_{ij})$$

where M is the number of objects in the model base and G is the number of tessellated viewpoints for the aspect data. Here, D_{ij} refers to the original raw aspect data at the jth site in model number i. c_{ij} is the corresponding class-candidate label assigned to it by the AAG construction algorithm.

In order to account for the smoothness and error criteria, we need to consider the distance between two class-candidates, $d(\ ,\ )$. We recall the feature tuple rep-

resentation of a class-candidate (Section 3.3.1). For instance, features could be the number of junctions of each type present in a view (V-junctions, T-junctions, etc.), as in Chakravarty and Freeman's representation of a view class [40]. For distance computation, we use the Euclidean Distance between two normalized class-candidate vectors. Each feature corresponding to a class-candidate is normalized with respect to its mean and standard deviation. We thus normalize both the input and output aspect data with respect to their respective mean vector and standard deviation vector. The error at a site is the Euclidean distance between the normalized original raw aspect data feature vector and the normalized vector corresponding to the label assigned by the algorithm. Evaluation of the smoothness at a site requires the distance between two adjacent normalized class-candidate feature vectors.

For the problem of AAG generation from noisy aspect data, one has to consider the following factors:

- the algorithm should be reasonably fast (*i.e.*, have polynomial time complexity in the size the AAG)
- the algorithm should give similar results for two 'similar' instances of raw aspect data
- the algorithm should give rise to prominent aspects, not too 'large' in number. The resulting AAG class-distribution plot should be piecewise smooth, with discontinuities only aspect boundaries.
- the AAG should be close to the original raw aspect data *i.e.*, the AAG construction algorithm should also ensure fidelity to the original aspect data.

Many of these requirements are conflicting – optimality in terms of one compromises optimality in terms of the other. For example, minimum error will be incurred if the output of the algorithm is the same as the raw aspect data. However, the resulting aspects may be small, and too many in number. If we impose the condition of a minimum allowable aspect size, the complexity required is exponential in the size of

the AAG (Proof in Appendix for Chapter 3: pages 124 – 125). Further, this approach may not satisfy the requirement of having a small number of prominent aspects. In the following section, we propose a new function to evaluate the output of different AAG construction algorithms.

3.5.2 A New Evaluation Function for AAGs

In view of the above desirable characteristics, we propose a new coefficient to evaluate the output of AAG construction algorithms. We define the **Demerit Coefficient** for the AAG of model i in model base $\mathcal{M}$ as follows:

$$\eta(\mathcal{M}, i, \tau) \triangleq \mu \sum_j (1 - \rho_{ij})\ d(c_{ij}, c_{ij+1}) + \nu \sum_j d(c_{ij}, D_{ij}) + \sigma \sum_j \rho_{ij}$$

where D_{ij} is the original raw aspect data item at site j of model i, c_{ij} is the class-candidate assigned to site j by the AAG construction algorithm, $d(\ ,\ )$ denotes the Euclidean distance operator for two normalized feature vectors, and μ, ν and σ are constants. (For our experimentation, we have chosen the constants μ, ν and σ such that all the three terms have the same order of magnitude.) ρ_{ij} is defined to be 1 at if $d(c_{ij}, c_{ij+1})\ >$ threshold τ, and 0 otherwise. (Such objective functions have been used in image processing [87], [19]. The book [100] cites examples of fitting a curve to given data and image restoration, where Hopfield Nets are used to minimize such a term.)

The first term takes the piecewise smoothness criterion into account – if there is a prominent discontinuity between two adjacent class-candidates, then the $(1 - \rho)$ term will not allow the distance between the two class-candidates to penalize the Demerit Coefficient. The second term considers the fidelity between the original class-candidate at a site and the one assigned to the site by the algorithm. The last term considers the number of prominent discontinuities in the aspect data.

In case we have more than one instance of raw aspect data for a particular model, we consider the Demerit Coefficient for the AAG of the model as the average of

the Demerit Coefficients computed for each given instance. We define the Demerit Coefficient for the set of AAGs for the entire model base as the average of the Demerit Coefficients of the M individual models' AAGs, using the same constants and thresholds for each model:

$$\eta_{model\ base}(\mathcal{M},\tau) \triangleq (1/M)\ \sum_{i=1}^{M} \eta_{model\ base}(\mathcal{M},\tau) \triangleq (1/M)\ \sum_{i=1}^{M} \eta(\mathcal{M},i,\tau)$$

To evaluate the performance of an AAG construction algorithm, we consider the values of the Demerit Coefficient before and after the application of the algorithm. To calculate the Demerit Coefficient for the raw aspect data, we make the following observation. We can consider the raw aspect data itself as the output of an AAG construction algorithm. Hence, c_{ij} is the same as D_{ij} for this case. The second term of $\eta(\mathcal{M},i,\tau)$ is zero, while the presence of errors in the raw aspect data causes the first and third terms to have large values. The output of a good aspect graph construction algorithm is smooth, piecewise continuous, and is close to the original data. Hence for such a case, all the three components of $\eta(\mathcal{M},i,\tau)$ are expected to have low values.

3.5.3 Algorithm for 1-DOF AAG Generation

This section proposes a low order polynomial time-complexity algorithm (polynomial in the size of the AAG) for building an AAG from noisy aspect data in the 1-DOF case. Our algorithm incorporates clustering heuristics for modifying the raw aspect data in order to reduce its Demerit Coefficient. The input to the algorithm is noisy raw aspect data. The aim of the algorithm is to construct AAGs of the objects in the model base for robust class recognition. The algorithm has the following output:

- An AAG for each model in the model base
- A list of all classes and feature-classes
- statistics about the suitability of a feature detector (Section 3.6).

Extensive experiments with two model bases (Section 3.7) shows that the output of the algorithm has far lower values of the Demerit Coefficient compared to the input noisy data. The regions where our algorithm requires execution time quadratic in the size of the region, are a small part of the total size of the AAG. The rest is done in linear time.

The *ASSOC_TABLE* and Association Probability Estimates

The algorithm maintains estimates of the probability with which one class-candidate is observed as another. For example, one expects valid class-candidates to be observed correctly with a high probability. To keep estimates of these association values, the algorithm uses an $N_{\mathcal{C}} \times N_{\mathcal{C}}$ matrix, the *ASSOC_TABLE*. The [i][j]th entry stores the probability of the class-candidate actually being c_j, given that class-candidate c_i is observed. The *ASSOC_TABLE* stores these values as counts of the number of times one class-candidate appears as another. We use the *ASSOC_TABLE* estimates for our robust 3-D object recognition strategy (Chapter 4).

We have a chicken-and-egg problem here. Proper *ASSOC_TABLE* estimates of $P(c'_{actual} \mid c_{observed})$ would be available only after the completion of the algorithm execution. However, the algorithm needs proper (qualitative) estimates of $P(c'_{actual} \mid c_{observed})$ for various processing operations. Hence at each stage, we need a good current estimate of $P(c'_{actual} \mid c_{observed})$ values. Therefore, we structure the phases in our algorithm in such a way that we fulfill this requirement.

Our algorithm is divided into three phases:

Algorithm Phase I

Phase I of our algorithm is primarily concerned with *identification of valid class-candidates*. Figure 3.9 gives an overview of this phase. The input to the algorithm is the feature tuple data from all sites in the tessellated viewpoint space around an object. The first phase clusters the data into aspect-candidates and forms a cir-

ALGORITHM: Phase_I

```
FOR the raw aspect data of each model IN model_base REPEAT Step 1:
1.  FOR EACH site IN raw aspect data DO
      IF (CLASS_CAND(current_site) == CLASS_CAND(previous_site)) THEN
         assign current_site to the previous_site's aspect_candidate;
      ELSE
         start a new aspect_candidate from the current_site;
2.  Assign the status field of each class_candidate c as follows:
      IF the extent of any aspect_candidate associated with c ≥ Θ_p
         THEN c.status := VALID;
      ELSE c.status := UNASSIGNED;
```

Figure 3.9: The AAG Construction Algorithm for the 1-DOF Case: Phase I

cular doubly linked list (Section 3.3 describes the representation of an AAG). For class-candidates c corresponding to prominent aspect-candidates, $P(c_{actual} \mid c_{observed})$ values are expected to be high. Prominent aspect-candidates correspond to valid class-candidates. We use this fact to advantage – we initialize the $ASSOC\_TABLE$ with the corresponding extents of these aspect-candidates. For all class-candidates corresponding to prominent aspect-candidates, we set their $ASSOC\_TABLE$ entries (*i.e.*, only those diagonal entries $ASSOC\_TABLE[i][i]$ corresponding to the corresponding class-candidates) to be the sum of the extents of all prominent aspect-candidates corresponding to this class-candidate. The $P(c_{actual} \mid c_{observed})$ for such class-candidates will not change much during the course of the algorithm execution. We explain the main steps in this phase of the algorithm as follows.

The algorithm clusters the raw aspect data into aspect-candidates using a 1-D version of Horn's sequential labeling algorithm ([103]). Comparing two class-candidates amounts to finding the distance between two class-candidate labels, and checking if this is zero or negligible, depending on the definition of a class-candidate (Section 3.5.1). This phase creates the aspect-candidate list for each object, and the class-candidate list for the model base. For an AAG which is not heavily corrupted with errors, the class-candidates corresponding to prominent aspect-candidates are expected to occupy more than N_{min} sites. Hence for such an AAG, this phase iden-

ALGORITHM: Phase_II

```
FOR EACH model IN model_base REPEAT Step 1:
1.  FOR EACH pair of proximal valid aspect_candidates a_i AND a_j DO
      IF CLASS_CAND(a_i) == CLASS_CAND(a_j) AND
      gap(a_i,a_j) < Θ_min THEN
         integrate all aspect_candidates from a_i to a_j into one;
2.  FOR EACH class_candidate c with (c.status == UNASSIGNED) DO
      IF the number of sites at which c is observed < N_min
         THEN c.status := INVALID;
```

Figure 3.10: The Algorithm: Phase II

tifies most of the valid class-candidates for the given model base. The algorithm sets the status field for each such class-candidate accordingly. Setting the status label is done to speed up the search for a class-candidate with a particular label – VALID, INVALID or UNASSIGNED.

Phase I does not remove any errors from the raw aspect data. At the end of the entire algorithm, all valid class-candidates constitute the list of all classes for the given model base. A class-candidate nodes have the property that its label does not change from VALID to INVALID, or vice versa during the course of the algorithm.

This phase requires one pass through the raw aspect data at each site for each model in the model base.

Algorithm Phase II

Phase II is primarily concerned with identification of prominent aspect-candidates after removing interspersed errors. In this phase, we consider small isolated regions between two valid aspect-candidates. We specifically consider the case when the two aspect-candidates correspond to the same class-candidate. We handle Type II and Type III errors in this phase. Figure 3.10 presents the steps in this phase.

In this phase, we consider pairs of **proximal valid aspect-candidates**, with the same class-candidate, say c. We define a pair of valid aspect-candidates (a_i, a_j) as

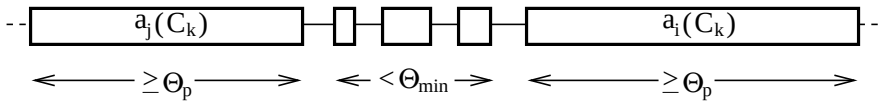


Figure 3.11: The situation handled by Phase II

proximal valid aspect-candidates if

$0 < \theta_{a_k} < \Theta_{min}$, $\forall\ a_k$ lying in between a_i and a_j in the direction of traversal of the aspect-candidate list.

Figure 3.11 shows an example of this condition. For each pair of proximal valid aspect-candidates with the same class-candidate separated by a gap of width $\leq \Theta_{min}$, we integrate both the valid aspect-candidates and those in between them, into one.

The correct class-candidate for the aspect-candidates in between the valid aspect-candidates a_i and a_j, is considered to be that of a_i and a_j. The algorithm updates the *ASSOC_TABLE* with the information about the class-candidates of the valid aspect-candidates and those in between them. For the class-candidates corresponding to the valid aspect-candidates, we increment the $ASSOC\_TABLE[i][i]$ count by the size of the aspect-candidates. For the aspect-candidates a_k between a_i and a_j, the algorithm updates the entry corresponding to $CLASS\_CAND(a_k)$ being observed as $CLASS\_CAND(a_i)$ by the size of aspect-candidate a_k. After we perform the above operations for each model in the model base, we update the status labels of each class-candidate labeled UNASSIGNED, as in the first phase. This is due to the property that no class-candidate changes its status label from VALID to INVALID, or vice versa during the course of the algorithm.

Phase II of our algorithm removes Type II and Type III errors. (Whether the isolated error removed is a Type II or Type III error will be clear from the *ASSOC_TABLE* conditional probabilities).

As mentioned earlier in the introduction to the use of the *ASSOC_TABLE*, we structure the phases of our algorithm in such a way that the association estimates are expected to be correct at each stage. In the first phase, we initialized

the *ASSOC_TABLE* for class-candidates which are most likely to be valid class-candidates. In this phase, we store estimates to specifically capture Type II errors. In the subsequent Phase III, the algorithm uses the *ASSOC_TABLE* to account for association errors. If the value of $P(c_{actual} \mid c'_{observed})$ is above a particular threshold (c and c' are not necessarily different), the algorithm interprets an instance of c' to be c. Results of over 100 experiments with two model bases shows that using this association information reduces the error between the raw aspect data and the AAG created by the algorithm (Section 3.7). Our experimentation also shows that these estimates of association errors made in Phase II are reasonably correct. Section 3.7 gives details about a comparison of the estimates of association errors after Phase II to those after the completion of the algorithm.

Algorithm Phase III

The third phase of our algorithm handles the rest of the raw aspect data. There are two passes through Phase III. The first is a logical pass, done in order to get further (better) estimates for *ASSOC_TABLE* entries. At the end of the logical pass, we keep just the *ASSOC_TABLE* unchanged and undo all other changes made in the logical pass. The actual pass (which follows the logical pass) uses *ASSOC_TABLE* estimates made during Phase II as well as the logical pass of Phase III. This phase concerns the removal of Type I, Type IV and Type V errors. Figure 3.12 gives an outline of the steps performed in this phase.

In Phase III again, we consider pairs of proximal valid aspect-candidates a_i and a_j. Depending on the gap between a_i and a_j in the direction of traversal, we consider two cases:

Case 1: gap(a_i,a_j) $< \Theta_{min}$

In such a case, we cannot have any valid aspect-candidate in between a_i and a_j. For this reason, we obtain the minimum square-error decision boundary for the region of gap δ ($\delta < \Theta_{min}$). To get the minimum square-error decision boundary, we place the

ALGORITHM: Phase_III

```
(* Two passes are made through Phase_III. *)
(* After the first(LOGICAL) pass, keep ASSOC_TABLE unchanged but UNDO *)
(* all other changes. Then, make the second (ACTUAL) pass. *)
FOR EACH model IN model_base REPEAT the following step
   FOR EACH pair of proximal valid aspect_candidates a_i & a_j DO
    (* consider region from a_i to a_j *)
        IF gap(a_i, a_j) < Θ_min THEN
           BEGIN (* create single decision boundary *)
           a. get the min-error decision boundary from raw aspect data
           b. get the min-error decision boundary considering
                 ASSOC_TABLE info for each class_candidate
           c. of the two decision boundaries, select the one which
                 incurs min error
           d. integrate aspect_candidates from a_i till just before
                 the decision boundary, into a_i;
              integrate aspect_candidates between the decision boundary
                 & a_j, into a_j;
           END (* create single decision boundary *)
        ELSE
           BEGIN (* handle large region *)
           1. construct normalized freq histogram for class_candidates
           2. (*  — First Rule —  *)
            IF max(histogram) ≥ T_2 AND the corresponding class_candidate
                == CLASS_CAND(a_i) == CLASS_CAND(a_j) THEN
                 integrate all aspect_candidates from a_i to a_j into a_i;
                 RETURN;
           3. (*  — Second Rule —  *)
            FOR EACH histogram entry ≥ T_3 DO
                 IF the corresponding class_candidate == CLASS_CAND
                 of any/both surrounding aspect_candidates THEN
                    integrate aspect_candidates in between to that/those
                    surrounding aspect_candidate(s); RETURN;
           4. (*  — Third Rule —  *)
            FOR EACH class_candidate c whose histogram entry ≥ T_4 DO
                  FOR EACH pair of aspect_candidates (a_k, a_l) between
                  a_i & a_j with class_candidate c
                     IF gap(a_k,a_l) ≤ THEN
                        integrate aspect_candidate from a_k to a_l into a_k;
                  ELSE
                     IF current_size(a_k) < Θ_min THEN
                        UNDO all changes to a_k;
           5. FOR EACH remaining region between a_i & a_j DO
                  create single decision boundary (* as in steps a - d *)
           END (* handle large region *)
```

Figure 3.12: The Algorithm: Phase III

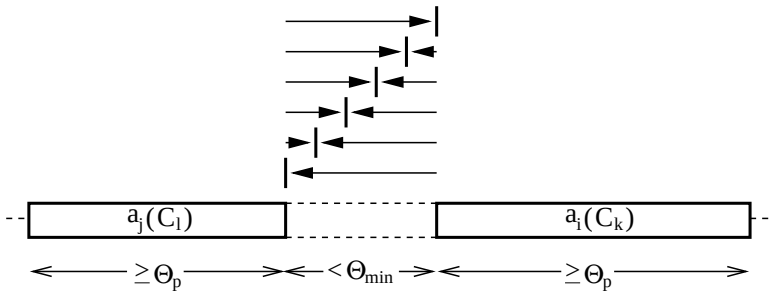


Figure 3.13: Getting a single decision boundary between two valid aspect-candidates

decision boundary at each position between the end point of a_i and the start point of a_j. Figure 3.13 illustrates this point. We consider the part of the gap till the decision boundary assigned to the class-candidate label $CLASS\_CAND(a_i)$ and the rest, $CLASS\_CAND(a_j)$. The error for a decision boundary is the sum of Euclidean distances between the original class-candidate label and the one just assigned for the entire region, both suitably normalized. We select the one which incurs minimum error. This process has quadratic ($O(\delta^2)$) time complexity as one has to calculate the error for the entire region considering all possible positions of the decision boundary. We can thus remove Type I errors.

This is one point where we use the association of one class-candidate with another. We now consider the minimum-error decision boundary taking into account the association information collected thus far. For each class-candidate in the region of width δ, if its probability of being some other class-candidate is above a particular threshold, we replace it with this class-candidate for the purpose of getting an alternative decision boundary. Of the two decision boundaries calculated so far, the algorithm takes the one with the minimum error. We update the raw aspect data, the class-candidate list and the $ASSOC\_TABLE$ with this information.

Case 2: gap(a_i,a_j) $\geq \Theta_{min}$

We now consider the case when the gap between valid aspect-candidates a_i and a_j is $\geq \Theta_{min}$. While we would like our algorithm to incur as small error as possible between the raw aspect data and the original class-candidate labels, we would also

like our algorithm to be fast. For a small region (width $< \Theta_{min}$), we can justify taking a minimum square-error configuration since we can have one decision boundary, the calculation of which takes quadratic time. Thus, we have to make a trade-off here: the optimality in terms of one parameter affects the optimality in terms of another. Further, taking a single decision boundary for a large region may not just be time-consuming, but also may not be optimal since there may be one or more aspects in the region under consideration. To handle these cases, we try the following heuristic rules 1 – 3. If one of them succeeds, we exit and consider the next pair of proximal valid aspect-candidates. Otherwise, we try the next one. These rules aim at removing Type IV and Type V errors.

We construct a normalized histogram for the class-candidates in the region between valid aspect-candidates a_i and a_j. (For each class-candidate between a_i and a_j, the normalized histogram stores its relative frequency of occurrence in the region.)

1. **First Rule:**

   This is applied if the following conditions are satisfied:

   (a) The surrounding valid aspect-candidates a_i and a_j have the same class-candidate

   (b) The maximum histogram value exceeds a threshold T_2

   (c) The maximum histogram entry is the same as the class-candidate of the surrounding aspect-candidates a_i and a_j

   If the above conditions hold good, we integrate the region from a_i to a_j into one aspect-candidate and suitably update the aspect data, class-candidate list and *ASSOC_TABLE*. We now go back to consider the next pair of valid aspect-candidates.

2. **Second Rule:**

   If the conditions for the above heuristic are not satisfied, we try our second rule for all histogram entries above another threshold T_3. (Our implementation has

$T_2 = 0.75$ and $T_3 = 0.4$). If the histogram entry is the same as the class-candidate of one/both of the surrounding valid aspect-candidate a_i or/and a_j, then we integrate the entire enclosed region with that/those valid aspect-candidates(s), and go back to consider the next pair of valid aspect-candidates. If not, we try the next histogram entry satisfying the above criterion. If we exhaust all such possibilities, we go in for the region-growing heuristic described below.

3. **Third Rule (Region Growing):**
   For region-growing, we consider each histogram entry (class-candidate) whose value is above a particular threshold T_4. We consider pairs of aspect-candidates (a_k, a_l) belonging to that class-candidate lying between a_i and a_j. If the gap between a_k and a_l is less than or equal to the current size of a_k, we integrate the region from a_k to a_l into one, and continue with the next pair. If not, then we check the current size of a_k. If it is below Θ_{min}, we undo all changes made to it and try another pair, till we are done with all suitable pairs between a_i and a_j. Before working on aspect-candidates between a_i and a_j, we first try a similar region growing on a_i. After performing region-growing for aspect-candidates between a_i and a_j, we repeat the exercise for a_j.

After trying out region growing, we may still be left with regions between a_i and a_j unaccounted for. We get a single minimum square-error decision boundary for all such regions, as in Case I above.

3.5.4 Algorithm for 3-DOF AAG Generation

Our algorithm for the 3-DOF case follows on the same lines as the 1-DOF case algorithm. Hence the descriptions are brief, with only the salient points mentioned. It has three phases:

Phase I

We adapt Horn's sequential labeling algorithm ([103]) for the spherical array representation of the geodesic tessellated viewing space (Figure 3.4(a), (b)). This phase creates the graph representation for the raw aspect data. The following pseudo code shows the portion of the sequential labeling algorithm where the raster scan of the spherical array is performed. The function `examine([i, j, k])` considers all immediate neighbours of the tessel [i, j, k] which have already been examined earlier in the raster scan order, if any. If the class-candidate at tessel [i, j, k] is different from that of the neighbours seen earlier, then it is assigned a new aspect-candidate. Else, it is assigned the aspect-candidate of a neighbour with whom it shares the same class-candidate. In case there is more than one such neighbour, an equivalence table records the fact that the aspect-candidates corresponding to these are equivalent.

```
examine([0, 0, 0]); (* The zenith tessel *)
(* For the upper portion *)
FOR r:= 2 TO Q DO
  FOR i:= 1 TO 5 DO
   FOR l:= (r-1) DOWNTO 1 DO
    examine([i, l, r-l]);
(* For the middle portion *)
FOR r:= (Q+1) TO (2*Q+1) DO
  FOR i:= 1 TO 5 DO
   FOR l:= Q DOWNTO 1 DO
    examine([i, l, r-l]);
(* For the lower portion *)
FOR r:= (2*Q+2) TO (3*Q) DO
  FOR i:= 1 TO 5 DO
   FOR l:= Q DOWNTO (r-2*Q) DO
```

```
    examine([i, l, r-l]);
examine([-1, 0, 0]);  (* The nadir tessel *)
```

This phase requires two passes through the raw aspect data. All equivalent aspect-candidates are grouped into one in the second pass. We now group all adjacent 'small' aspect-candidates into regions $\mathcal{R}$. We represent these regions as supernodes in the graph data structure (Section 3.4.2). As in the 1-DOF case, this phase is primarily concerned with identification of valid class-candidates.

Phase II

Phase II removes Type II and III errors from the raw aspect data. In this phase, we consider all supernodes in the graph data structure corresponding to regions where Type II and III errors may be present. We integrate such regions with the aspect candidates which belong to class-candidate c (Section 3.4.2). This phase updates the *ASSOC_TABLE*, just as in Phase II of the 1-DOF algorithm. The time complexity for Phase II is linear in the number of nodes in the graph data structure.

Phase III

Just as it was for the 1-DOF algorithm, Phase III has a logical and an actual pass. In Phase III, we consider all supernodes representing regions $\mathcal{R}$. We construct a normalized histogram for the relative frequency of occurrence of a class-candidate for the region $\mathcal{R}$. Depending on the size of region $\mathcal{R}$, we consider two cases:

Case 1: $|\mathcal{R}| < \Omega_{min}$

We consider Type I errors here. For the 1-DOF case, we could consider taking an optimal decision boundary (quadratic time complexity). For the 3-DOF case however, the time complexity is exponential in the size of the region $\mathcal{R}$. As a compromise between optimality and high time complexity, we take the minimum error configuration out of the following cases: First, we consider $\mathcal{R}$ to have a class-candidate c – where the

extent to which the boundary $\mathcal{R}$ shares with aspect-candidates having class-candidate c, is more than that for any other class-candidate. If the class-candidate of any surrounding (valid) aspect-candidate is the same as that with the maximum histogram entry, then we consider two more cases: One, when the region $\mathcal{R}$ is assigned the same class-candidate and next, when the *ASSOC_TABLE* information is also taken into account. The algorithm creates the configuration with the least error, of the above three.

Case 2: $|\mathcal{R}| \geq \Omega_{min}$

Here, we aim at removing Type IV and Type V errors. As for the 1-DOF case, we use the first applicable rule out of the following three heuristic rules:

1. **First Rule:**

   This is applied if the following conditions are satisfied:

   (a) The percentage of the common boundary with aspect-candidates of a particular class-candidate (say, c) is more than a threshold

   (b) The maximum histogram value exceeds a threshold

   (c) c has the maximum histogram value.

   We integrate the region $\mathcal{R}$ with the aspect-candidates with class-candidate c, and suitably update the aspect data, class-candidate list and *ASSOC_TABLE*.

2. **Second Rule:**

   We try the second rule for all histogram entries above another threshold, T_4. If any surrounding valid aspect-candidate has the same class-candidate as the one considered, we integrate the region $\mathcal{R}$ with it. If we exhaust all such possibilities, we try a region-growing heuristic.

3. **Third Rule (Region Growing):**

   The region growing procedure is similar to that in the 1-DOF case. The criterion here is the distance between two aspect-candidates $DIST(\alpha_1, \alpha_2)$ (Section 3.5.1), and Θ_{min} is replaced by Ω_{min}.

If the region growing heuristic also fails, we may be left with unaccounted-for regions $\mathcal{R}$ of various sizes. We handle them in the same manner as we do for Case 1, above.

3.6 Suitability of a Feature Detector for Aspect Graphs

In their work on sensor modeling ([109], [110]), Ikeuchi and Kanade define sensor reliability as a measure of uncertainty in detected features. They model sensor reliability in terms of physical characteristics of the sensor. They consider uncertainty in sensor measurement due to variance in brightness values, light source directions and digitization mechanisms. A disadvantage of such an approach is that it may not be possible to model every type of sensor accurately. Further, existing approaches do not model the characteristics of the entire setup, or consider the behaviour of the image processing algorithms used – such as the effect of non-adaptive thresholds. For example, a setup may use ambient lighting with fluorescent tubelights. There may be flickering effects, and the background intensity may not be the same for two successive snapshots. We propose an experiment-oriented evaluation function for a feature detector which assesses the suitability of the detector for AAG construction. We evaluate the detector on the basis of a large number of observations. The advantage of such an approach is that it automatically takes into account all factors such as the entire setup characteristics, characteristics of the models in the model base, as well as the performance of the image processing algorithms used. Further, such an approach is independent of the type of sensor and feature set used.

We define the Unsuitability Factor for Feature F_k, with respect to the AAG of model number i as follows: (We use the 1-DOF case notation only for clarity, the

ideas presented here apply to the 3-DOF case as well)

$$u(F_k, \mathcal{M}, i, \tau') \triangleq \mu' \sum_j (1 - \rho'_{ij})\ d(f_{ij}, f_{ij+1}) + \nu' \sum_j d(f_{ij}, \mathcal{F}_{ij}) + \sigma' \sum_j \rho'_{ij}$$

The Unsuitability Factor for feature F_k with respect to the AAG of the entire model base is:

$$u_{model\ base}(F_k, \mathcal{M}, \tau') \triangleq (1/M) \sum_{i=1}^{M} u(F_k, \mathcal{M}, i, \tau')$$

Here, f_{ij} is the feature-class corresponding to feature F_k, for the class-candidate D_{ij} in the raw aspect data. $\mathcal{F}_{ij}$ is the feature-class corresponding to feature F_k, for class-candidate c_{ij} assigned to site j by the AAG construction algorithm. $d(\ ,\ )$ denotes the Euclidean Distance between two suitably normalized features, and μ', ν' and σ' are constants which depend upon the feature data characteristics.

For feature F_k to be considered 'suitable' with respect to model base $\mathcal{M}$, $u(F_k, \mathcal{M}, \tau')$ should be low. The last term in $u(F_k, \mathcal{M}, i, \tau')$ has $\rho' = 1$ if the distance between two adjacent feature-classes is above threshold τ', 0 otherwise. The first term takes smoothness over adjacent sites into account. If the distance between two adjacent feature-classes is low, it contributes to the term. If it is high and above threshold τ', the $(1 - \rho)$ term becomes zero, and the prominent discontinuity is not penalized. Thus, the first term ensures that the output of a 'suitable' feature detector is piecewise continuous over the entire stretch of an AAG plot. The third term ensures that these discontinuities are not too large in number. The second term takes fidelity to the original raw aspect data into account, comparing it with the feature-class of the class-candidate assigned by the algorithm. For our experimentation, we have chosen the constants μ', ν' and σ' such that all the three terms have the same order of magnitude.

This function indicates the suitability of a feature detector for aspect graph-based 3-D object recognition. The function $u(F_k, \mathcal{M}, \tau')$ indicates to what extent a particular feature detector F_k will be effective for recognizing objects of a model base. Further, one can compare the Unsuitability Factors for two detectors of the same feature *e.g.*, a grey-level corner detector, and one based on contour curvature. Given

a large set of feature detectors, one may use the Unsuitability Factor to choose a subset (of size k) for aspect graph-based object recognition. One can also perform an ordering of the feature detectors on basis of their Unsuitability Factors, to judge their suitability for the given model base and setup. In such a case, one can select the feature detectors corresponding to the k lowest Unsuitability Factors.

3.7 Experimental Results and Discussion

This section presents experimental results with two model bases for the 1-DOF case. We also present results of experiments for the 3-DOF case, with synthetic data.

3.7.1 The 1-DOF Case: Experimental Results

Our experimental setup has a camera connected to a MATROX Image Processing Card and a stepper motor-controlled turntable. The turntable moves by 200 steps to complete a 360 degree movement. We have experimented extensively with two object sets as model bases. Some details of the object sets are as follows:

1. *Model Base I: 7 Aircraft Models*

   Features used:

   (a) The number of horizontal lines ($\langle h \rangle$),

   (b) The number of vertical lines ($\langle v \rangle$), and

   (c) The number of circles ($\langle c \rangle$)

   We represent a class-candidate as $\langle hvc \rangle$. We have chosen this relatively feature-rich model base to demonstrate the effectiveness of our system on raw aspect data with very low smoothness in the raw aspect data. Figure 3.14 shows the objects in this model base.

2. *Model Base II: 8 Polyhedral Objects*

   Features used:

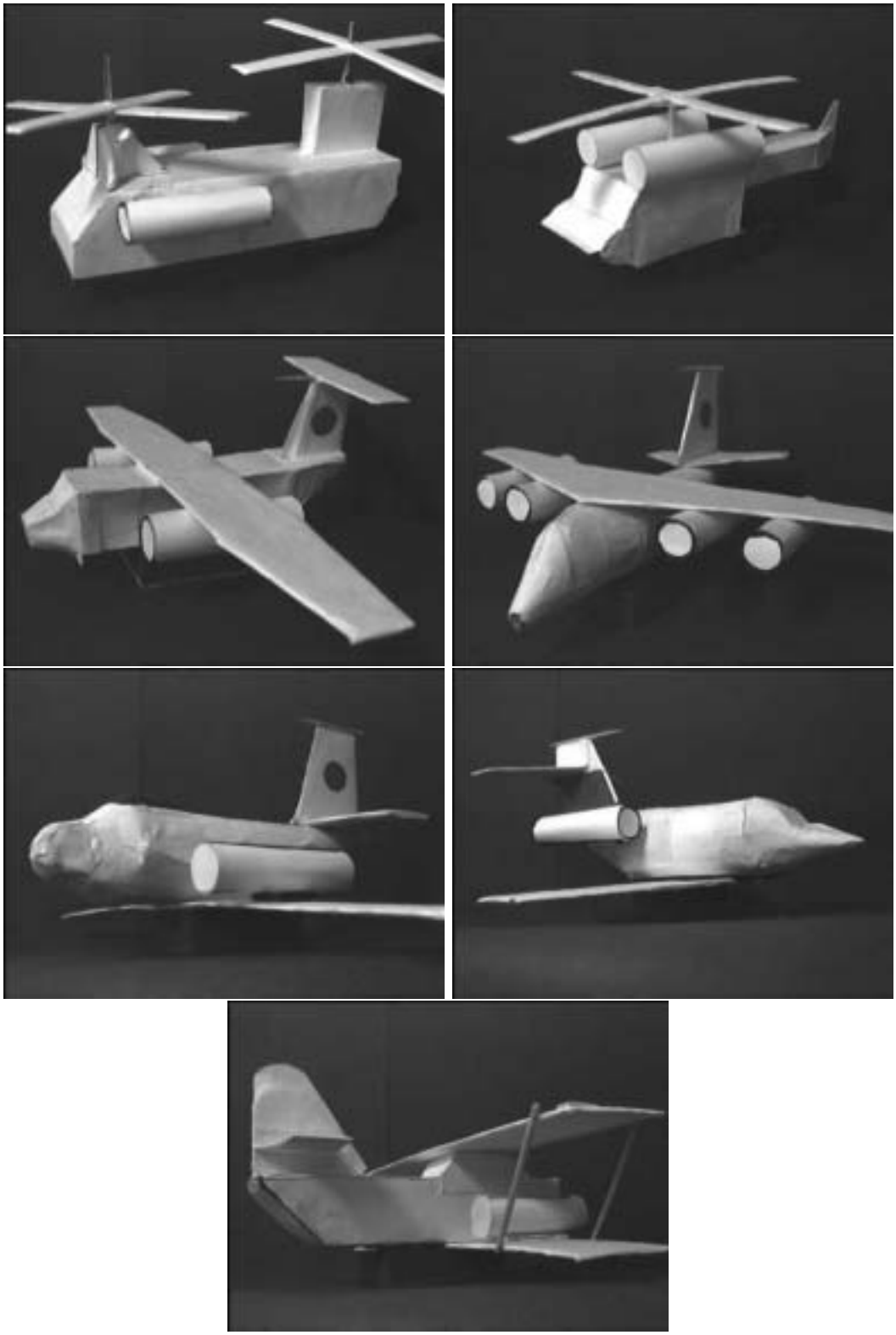


Figure 3.14: Model Base I: The objects (in row major order) are heli_1, heli_2, plane_1, plane_2, plane_3, plane_4, and biplane.

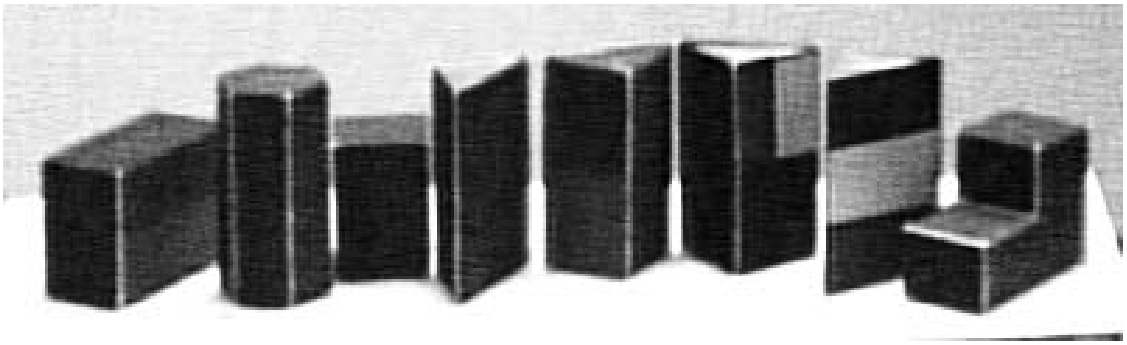


Figure 3.15: Model Base II: The objects (from left) are O_1, O_2, O_3, O_4, O_5, O_6, O_7 and O_8, respectively.

(a) The number of horizontal lines ($\langle h \rangle$),

(b) The number of vertical lines ($\langle v \rangle$), and

(c) The number of non-background segmented regions in an image($\langle r \rangle$)

We represent a class-candidate as $\langle hvr \rangle$. The raw aspect data for this model base has higher smoothness compared to the aircraft models. We have chosen this model base to compare the results of our system with those on the other model base. Figure 3.15 shows the objects in this model base.

The first step in our feature detection is to take a gradient image, using the Sobel Operator [97]. The operator combines speed of edge detection with the implicit smoothing operation to reduce noise. We use a Hough transform-based line and circle detectors [97] on the gradient image. We use the edge direction at a pixel to speed up the Hough transform. For getting the number of regions in the image, we perform Sequential labeling (connected components: pixel labeling) [97], [103] on a thresholded gradient image. This is a two-pass algorithm – the first pass traverses the image in raster-scan order. We require an auxiliary storage of size equal to that of the image, and an equivalence table to store equivalences of labels. The algorithm examines the grey level at a pixel with that of its 3 neighbours (which have been examined earlier). If they match (possibly within some error limit), the label at that

pixel is assigned the label of that neighbour ([97] and [103] discuss the details of various cases that may occur). The second pass resolves equivalent labels.

We recall our definitions of Smoothness (Section 3.5.1). Let the term 'Input Smoothness' ($\mathcal{S}(I)$) refer to the smoothness expression for the raw aspect data. Thus, c_{ij} is D_{ij} here *i.e.*, the raw aspect data item for the ith model at site number j. Similarly, we use the term 'Output Smoothness' ($\mathcal{S}(O)$) to refer to the smoothness expression for the output of the aspect graph construction algorithm. Thus, c_{ij} refers to the class-candidate label assigned to the jth site in the ith model by the algorithm. The aspect data for Model Base I (aircraft models) has a very high value of the Demerit Coefficient $\eta_{model\ base}$ and $\mathcal{S}(I)$ as compared to the aspect data for the other. Hence, we first present results of 100 experiments with the first model base. Each experiment considers a set of raw aspect data from each object in the model base. Then, we compare some figures with those of Model Base II (polyhedral objects).

Output of the Aspect Graph Construction Algorithm

Figure 3.16 shows a comparison of the raw aspect data and the output of our algorithm, for one instance of the aspect data for object plane_2 in Model Base I. A visual inspection of the lower graph shows that the aspects produced are prominent and not too large in number, the graph is piecewise smooth and at the same time, fidelity to the original data is high. Figure 3.17 shows an example for Model Base II.

Input and Output Smoothness

Figure 3.18 shows a comparison of the input and output smoothness for 100 sets of aspect data for the aircraft model base. The mean smoothness values for the input and output data are $\mathcal{S}(I) = 138.85$ and $\mathcal{S}(O) = 20.99$, while the variances are 4.51 and 0.74 respectively. This clearly shows that $\mathcal{S}$ values have greatly decreased. Even though the raw aspect data has a large variation in $\mathcal{S}$ values, the variation in $\mathcal{S}$ values for the output data is very small.

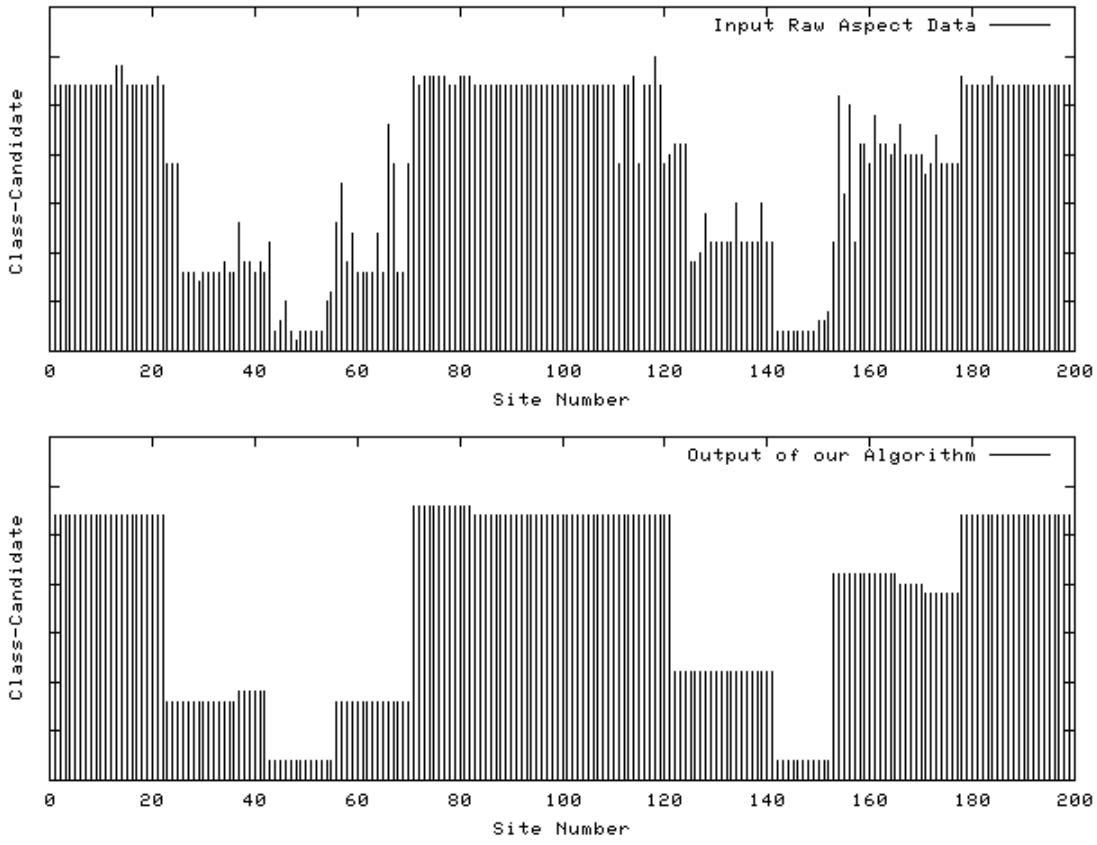


Figure 3.16: Raw aspect data and the output of our algorithm:plane_2, Model Base I. On the **Y**-axis, each class-candidate is represented by an index. Different heights represent different class-candidates. The tessellated viewing space has 200 sites.

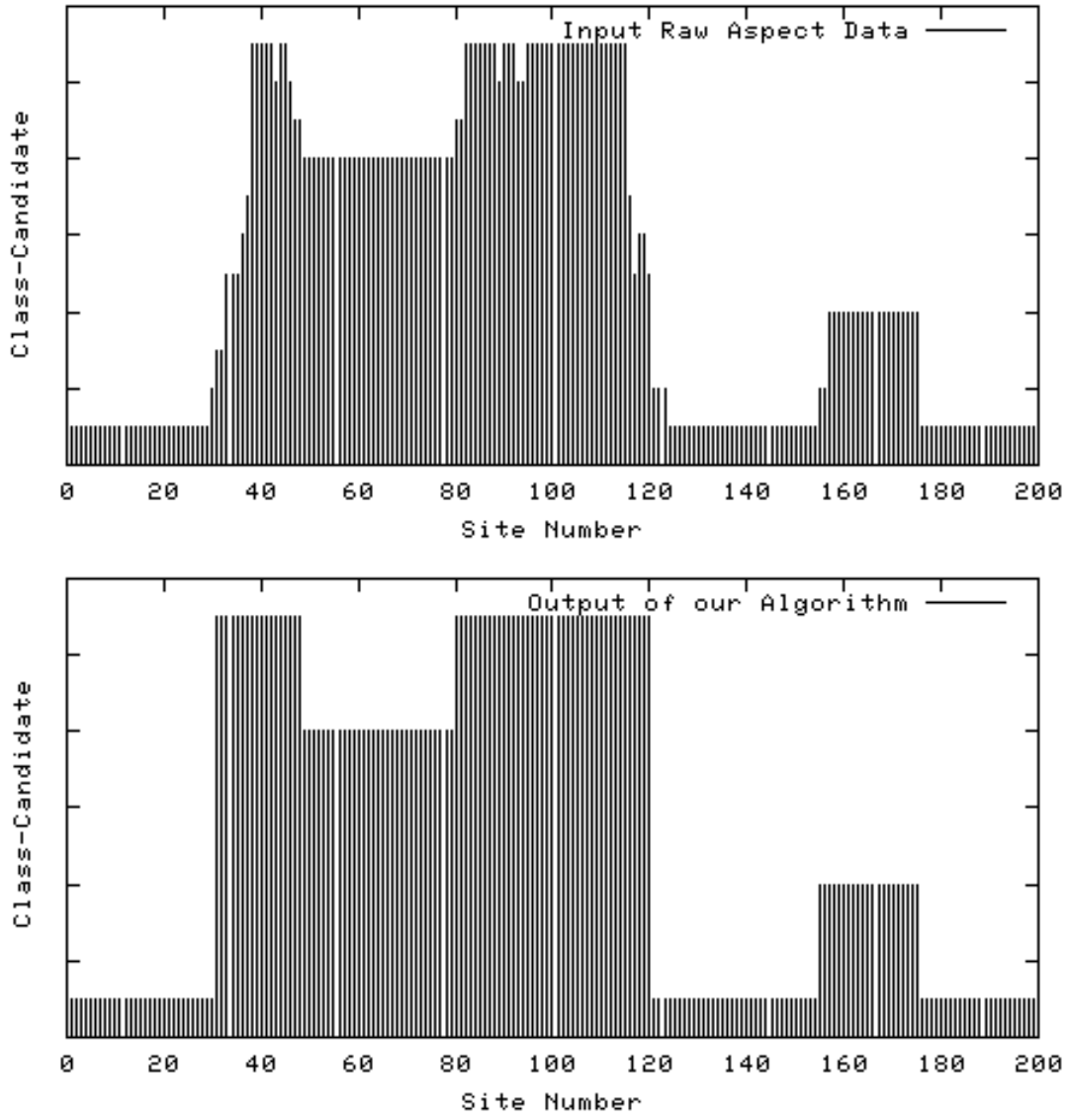


Figure 3.17: Raw aspect data and the output of our algorithm:O_6, Model Base II On the **Y**-axis, each class-candidate is represented by an index. Different heights represent different class-candidates.

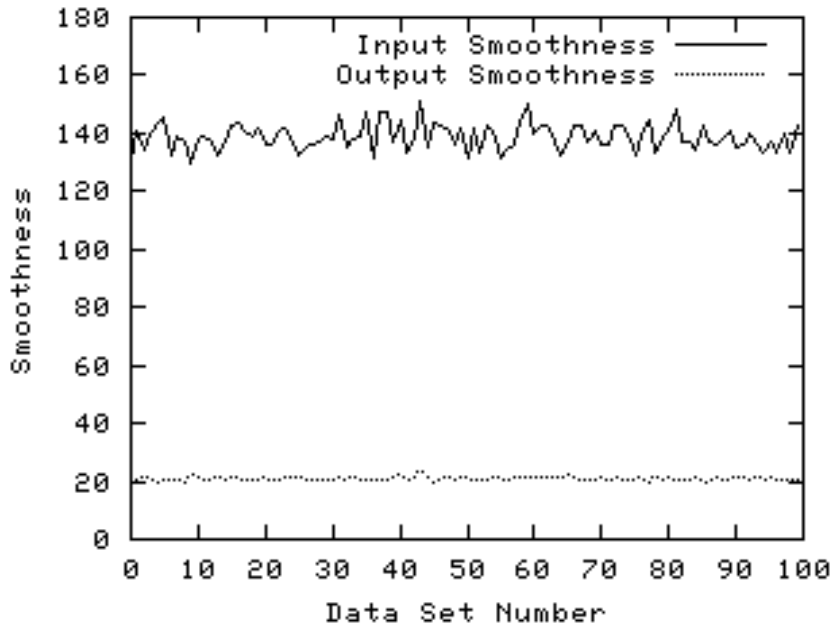


Figure 3.18: Variation in input 'smoothness' with the output 'smoothness'

Total Model Base Error

$\mathcal{E}$ is a measure of fidelity of the output data to the input raw aspect data (Section 3.5.1). Figure 3.19 shows the variation in $\mathcal{E}$ with the input smoothness for the 100 data sets.

Number of Aspects

Figure 3.20 shows the number of aspects obtained as a result of applying the algorithm on 100 instances of raw data for the model base. The figure shows the variation in the number of aspects with the input 'smoothness'.

Demerit Coefficients for Input and Output Aspect Data

Figure 3.21 shows the variation of the Demerit Coefficient for the input aspect data, with the Demerit Coefficient for the output of the AAG construction algorithm for the 100 data sets. Our AAG construction algorithm greatly reduces the Demerit

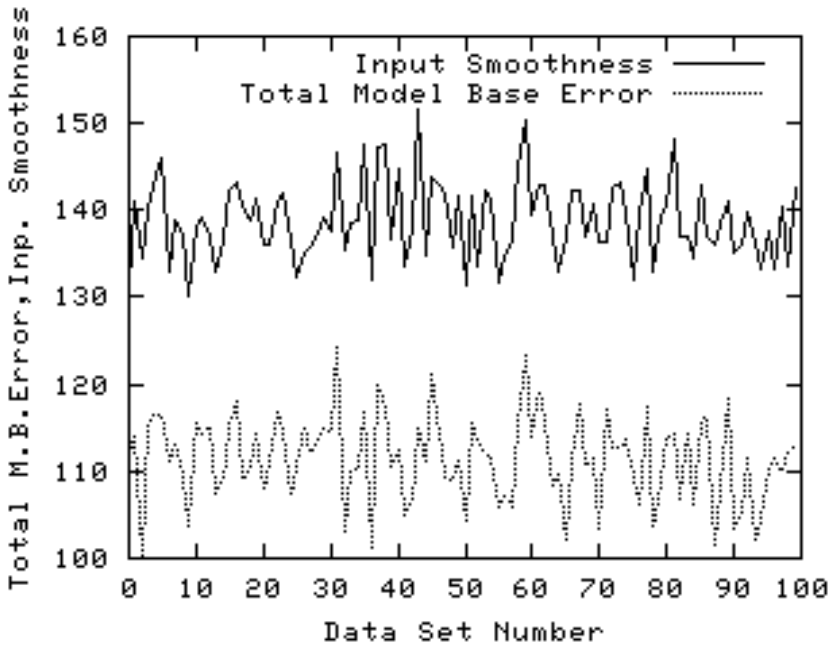


Figure 3.19: Variation in the total model base error with the input 'smoothness' for 100 data sets

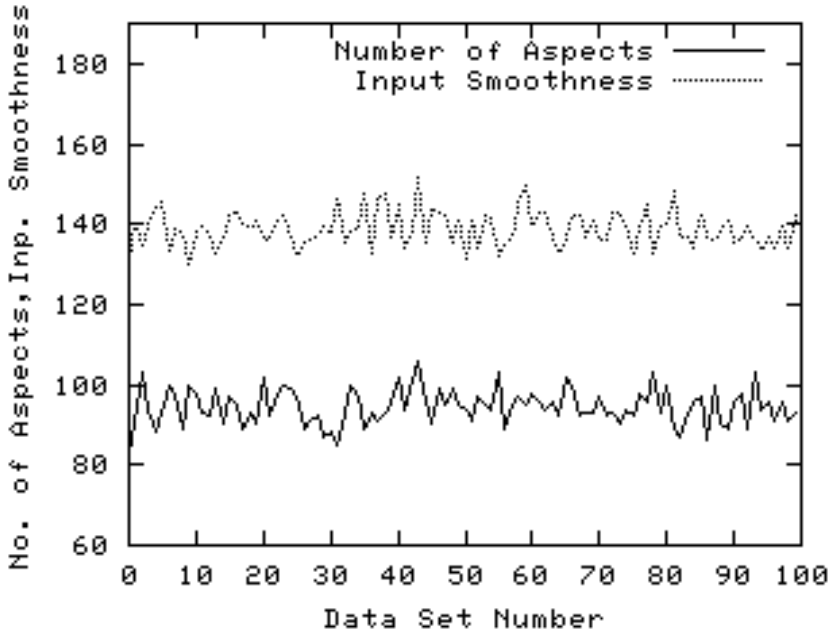


Figure 3.20: Variation in the number of aspects with the input 'smoothness', for 100 data sets

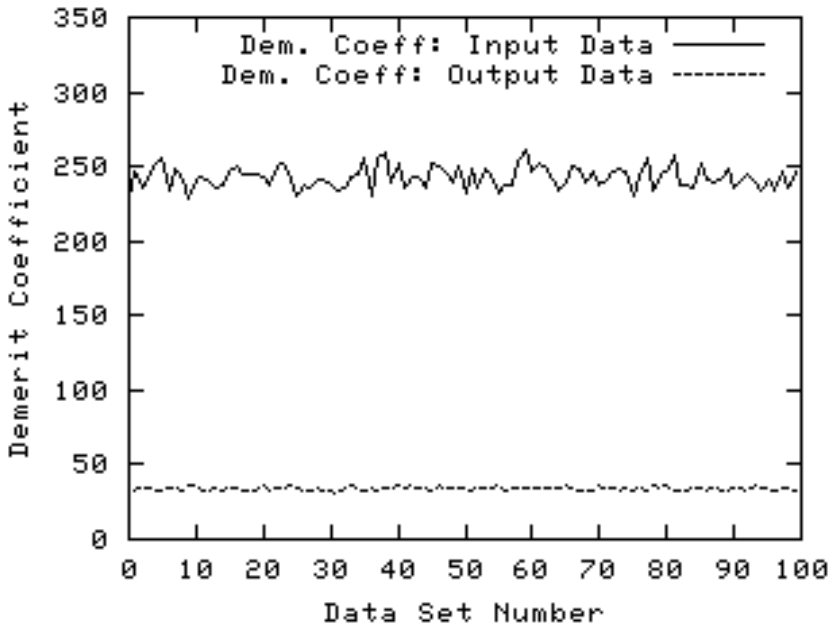


Figure 3.21: Variation in the Demerit Coefficient for the input raw aspect data, with the Demerit Coefficient for the output of the input raw aspect data: for 100 data sets

Coefficient. Further, the variation in the Demerit Coefficient for the output data is quite less compared to that for the input raw aspect data.

Percentage of sites where a single decision boundary had to be taken

The only part of our algorithm which has quadratic time complexity is where a single decision boundary has to be taken over a set of adjacent sites. The rest of it runs in linear time. Figure 3.22 shows the percentage of sites where a single decision boundary had to be taken, for 100 sets of aspect data. This is quite low (mean=29.47%, variance=1.85), even for aspect data with high $\mathcal{S}(I)$ values. This shows the efficiency of our algorithm, since one of the desirable characteristics of an AAG construction algorithm is that it should be fast *i.e.*, have low-order polynomial time complexity.

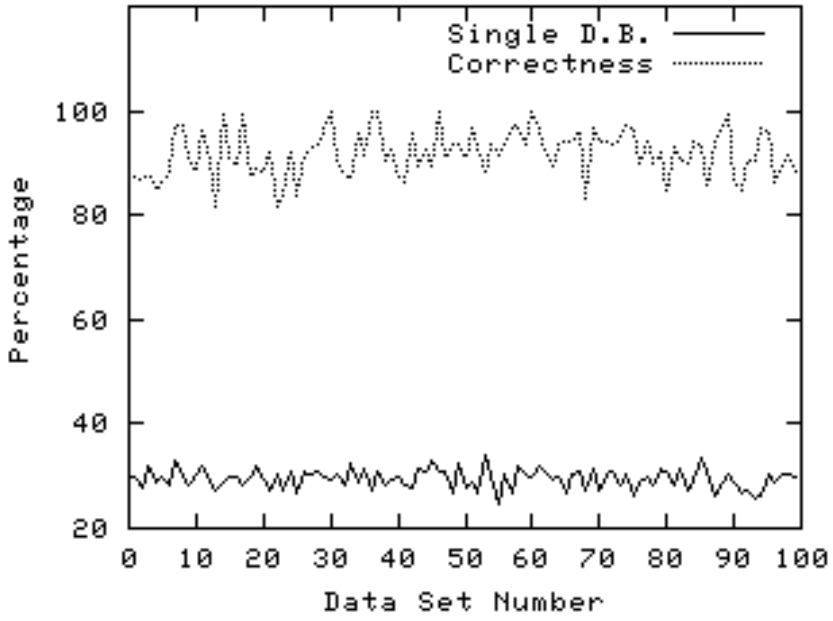


Figure 3.22: Percentage of sites where a single decision boundary had to be taken, and correctness in Phase II estimates in the *ASSOC_TABLE*

Correctness of Phase II *ASSOC_TABLE* estimates

The *ASSOC_TABLE* maintains estimates of the probability with which one class-candidate is observed as another (Section 3.5.3). If $ASSOC\_TABLE[i][j]$ remains above a threshold after Phase II as well as after Phase III (or equivalently, below it after both phases) – we refer to this as 'correctness'. Figure 3.22 shows the variation in percentage correctness of Phase II estimates for 100 data sets.

Percentage reduction in model base error with *ASSOC_TABLE* estimates

If $P(c_{actual} \mid c'_{observed})$ (Section 3.4) is above a particular threshold, we use this fact in order to obtain a minimum-error decision boundary (Section 3.5.3), Our experiments show that this reduces the total model base error, $\mathcal{E}$. The model base error $\mathcal{E}(A)$ is reduced if one used the association data from the *ASSOC_TABLE*. Figure 3.23 shows the percentage reduction in error for 100 instances of model base data.

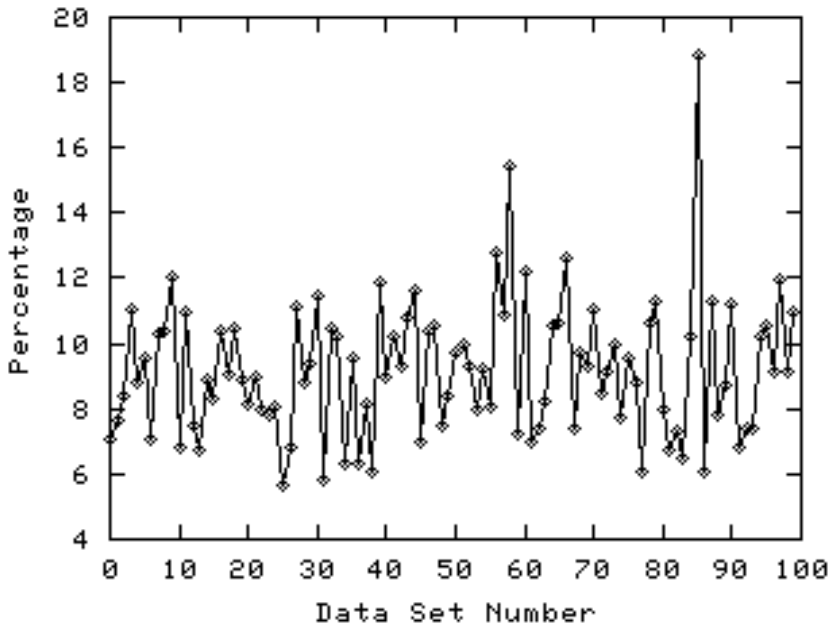


Figure 3.23: Percentage reduction in total model base error with *ASSOC_TABLE* data

Comparison of performance factors of our AAG construction algorithm on the two model bases

Table 3.1 shows the comparison between the two model bases. The figures for Model Base I are for 100 experiments, whereas those for Model Base II are for 4. Though the feature detectors used for the two model bases are different, the range of values taken by the feature-classes for the two model bases are comparable.

The raw aspect data from objects in the first model base has more errors than that from the second. In spite of this, the output smoothness obtained in both cases is comparable, and small, especially for the first case. Since the $\mathcal{S}(I)$ for the first model base is large, its model base error $\mathcal{E}$ is large compared to the corresponding data for the second model base. The first model base is such that the number of aspects is larger than that obtained for the second model base. Due to the large $\mathcal{S}(I)$ values for the first model base, the variance in the number of aspects is larger in the first case. The Demerit Coefficient for the output data is much less than that for the

	Model Base I		Model Base II	
PARAMETERS	*Mean*	*Variance*	*Mean*	*Variance*
Input Smoothness	138.85	4.51	43.99	0.47
Output Smoothness	20.99	0.74	18.93	0.34
Total Model Base Error	111.49	5.01	27.97	1.35
No. of Aspects	94.51	4.55	80.5	1.12
Demerit Coefficient (input data)	242.84	7.32	67.69	2.59
Demerit Coefficient (output data)	34.13	1.17	28.99	0.48
Quadratic Complexity Regions	29.47%	1.85	7.47%	0.41
Correctness of Phase II Estimates	91.91%	4.46	94.74%	3.72
Error reduction with *ASSOC_TABLE*	9.18%	2.08	4.85%	0.37

Table 3.1: Summary of AAG construction algorithm performance parameters for the two model bases: the 1-DOF case

input data. Since the second model base data has less feature detection errors, the percentage of sites in the AAG where a single decision boundary had to be taken is less than the corresponding value for the first model base. The correctness of Phase II *ASSOC_TABLE* estimates is more for the second model base, since the first one has more errors. Hence, the variance in the correctness values is more for the first model base compared to the second. Both cases show a reduction in model base error if the *ASSOC_TABLE* estimates are used to determine the correct class-candidate at a site.

A Comparison of Different Types of Errors

Table 3.2 presents a relative comparison of the different categories of errors (Section 3.4) for the two model bases. We list the mean and variance for each type of error for the 100 data sets in the first model base, and the 4 data sets for the second. *We emphasize here that the relative importance of different types of errors may vary across different model bases, and feature sets.* For a particular setup and a set of models, a particular error type may be more prominent compared to others. For example, while a Type V error is more common for the first model base, a Type I error is more common for the second. *However, from the point of view of robust object recognition, it is equally important to account for all types of errors.*

Suitability of a Feature Detector

Table 3.3 shows the Unsuitability Factors computed for the feature detectors in the two model bases. For our experimental set up and the objects in Model Base I, the number of horizontal lines is the most 'suitable' feature, followed by the number of circles, and the number of vertical lines. The corresponding features for Model Base II are the number of horizontal lines, the number of vertical lines, and the number of segmented regions.

Model Base I				
	Size		*Relative Size*	
	Mean	*Variance*	*Mean*	*Variance*
Type I	1.03%	0.19%	2.95%	1.57%
Type II	1.19%	0.27%	3.34%	2.03%
Type III	1.87%	1.26%	5.27%	9.44%
Type IV	11.60%	1.45%	32.89%	12.44%
Type V	19.38%	0.62%	55.55%	1.71%
Model Base II				
	Size		*Relative Size*	
	Mean	*Variance*	*Mean*	*Variance*
Type I	3.97%	0.03%	55.09%	4.30%
Type II	0.19%	0.00%	2.60%	0.00%
Type III	0.50%	0.00%	6.94%	0.00%
Type IV	0.46%	0.02%	6.31%	3.00%
Type V	2.09%	0.02%	29.06%	3.88%

Table 3.2: A comparison of the relative extents of different types of errors in AAGs in terms of the percentage of the total number of sites ('Size'), and the relative percentages of the errors ('Relative Size') (Section 3.4)

Model Base I		Model Base II	
Feature Detector	*Unsuitability Factor*	*Feature Detector*	*Unsuitability Factor*
Horizontal Lines	71.62	Horizontal Lines	3.31
Vertical Lines	116.44	Vertical Lines	17.47
Circles	85.87	Segmented Regions	25.41

Table 3.3: Unsuitability factors for feature detectors used in the two setups for the two model bases

3.7.2 The 3-DOF Case: Experimental Results

For our experiments with the 3-DOF case, we have used CAD data for a polyhedral object (Figure 3.24) to construct its AAG. We use as features:

1. The number of corners with the number of incident edges as 2, 3, 4 and 5, respectively, and

2. The number of faces of the object visible in the given view

Thus, a 5-dimensional vector of the above features represents a class. For the tessellation of the viewing space around the object, we choose Q=14 as the frequency of geodesic division. Thus, there are 1962 sites in the viewpoint space.

In order to generate the experimental data sets, we have perturbed the aspect data with different amounts of random noise. We randomly chose $k\%$ of the total 1962 sites for changing their class-candidate data. Each feature at such a site was replaced by a random number (in the range of the observed values of the feature in the aspect data obtained from the CAD model), We obtained 8 different data sets by taking k = 0, 1, 5, 10, 15, 20, 25, and 30, respectively.

Figure 3.25 shows a comparison of the raw aspect data with 30% noise, and the output of our algorithm, in the spherical array representation.

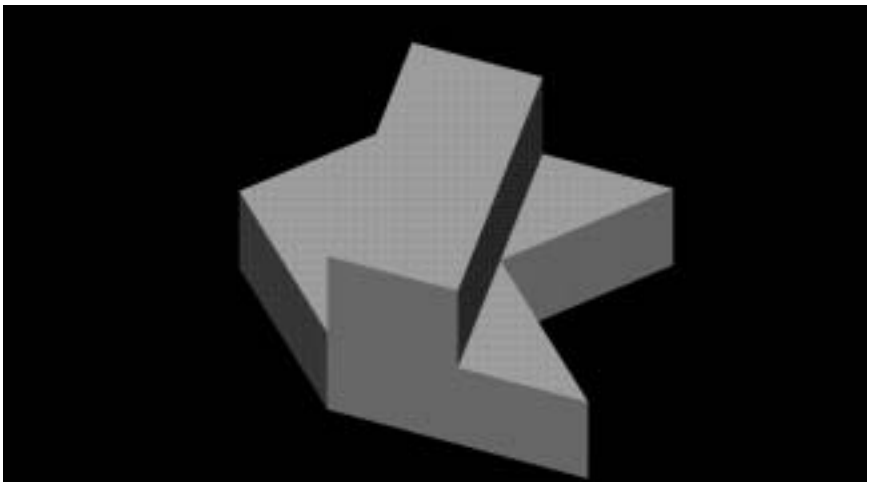


Figure 3.24: The CAD model of a polyhedral object, for experimentation with the 3-DOF case

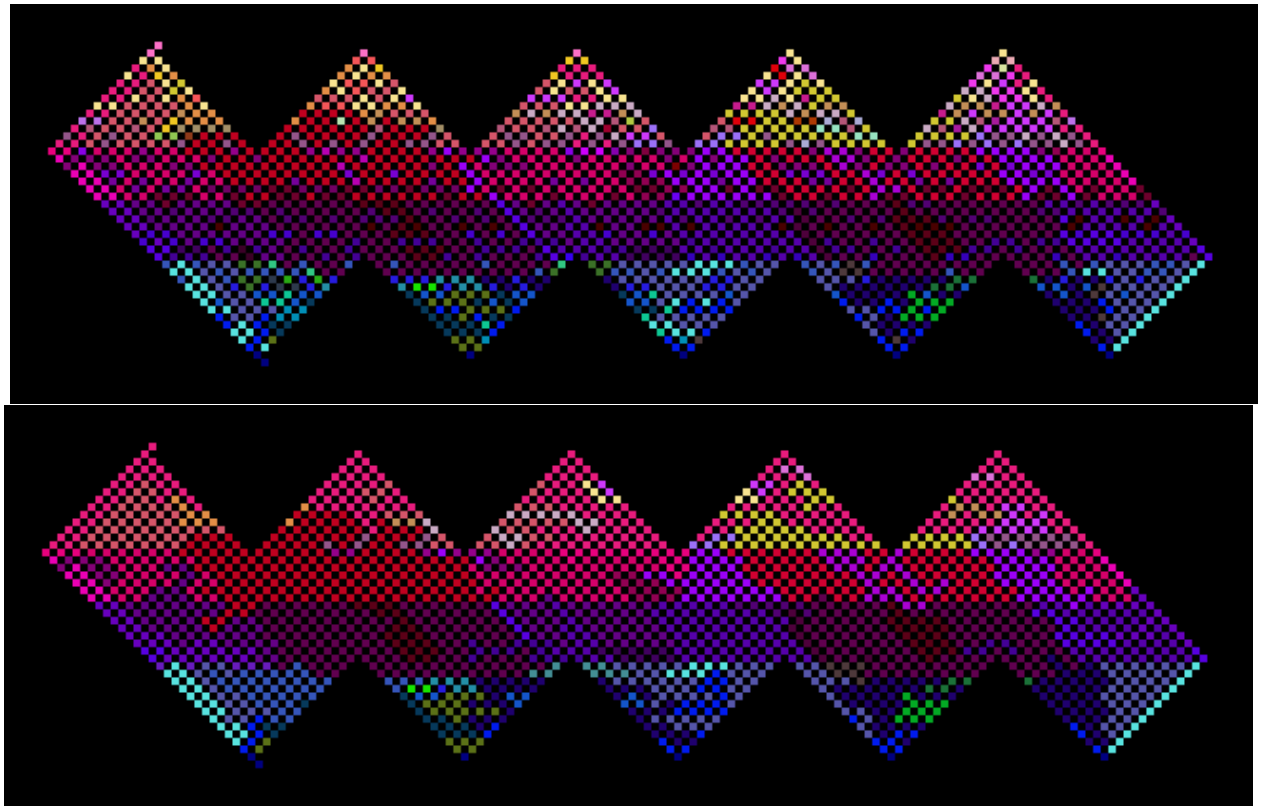


Figure 3.25: Raw aspect data with 30% noise, and the output of our algorithm for the 3-DOF case (Spherical array representation). Different colours represent different class-candidates.

	NOISE							
	0%	1%	5%	10%	15%	20%	25%	30%
$\mathcal{S}(I)$	1405.71	1578.89	2281.07	2945.86	3338.91	3629.05	4347.85	4344.85
$\mathcal{S}(O)$	1183.51	1203.18	1208.42	1306.09	1238.04	1605.85	1619.39	1817.89
$\mathcal{E}(\mathcal{A})$	329.31	411.99	851.48	1370.09	1661.98	1893.88	2385.50	2204.72
# aspects	59	59	57	56	57	61	63	60
$\eta(I)$	2762.44	2979.42	3857.72	4621.23	5090.32	5419.53	6212.21	6239.06
$\eta(O)$	2427.16	2469.35	2457.10	2553.13	2504.53	2846.96	2962.20	3214.79
O.D.B.	4.08 %	5.05 %	4.18 %	5.86 %	5.40 %	3.36 %	5.81 %	5.35 %
Correctness	97.78 %	88.68 %	90.79 %	88.31 %	87.84 %	86.25 %	95.00 %	86.42 %
Err. redn.	0.00 %	0.18 %	0.00 %	0.01 %	0.70 %	0.25 %	0.00 %	0.60 %

Table 3.4: Summary of AAG construction algorithm performance parameters for aspect data perturbed by different amounts of noise: the 3-DOF case. 'O.D.B.' denotes the percentage of the total AAG size where an optimal decision boundary needed to be taken (Details in Section 3.5.4). The term 'Correctness' denotes the correctness of Phase II *ASSOC_TABLE* estimates (Section 3.5.4) 'Err. redn.' denotes the percentage reduction in the model base error if *ASSOC_TABLE* estimates are used.

The lower figure shows prominent aspects created by our algorithm. Table 3.4 shows the results of running our algorithm on these data sets. The first column describes the results obtained by processing the aspect data directly obtained from the CAD model. The $\eta(I)$ value corresponding to this 0% error column represents the quality of the original aspect data with respect to the desirable criteria mentioned in Section 3.5.2. Examining the data in the table, we make the following observations. As we increase the amount of added noise to the aspect data, $\mathcal{S}(I)$ and $\eta(I)$ obviously increase. However, the increase in the $\mathcal{S}(O)$ and $\eta(O)$ values is considerably less. In fact, the values of $\eta(O)$ show that there has been considerable improvement in the

quality of the output data in spite of the noise added (*e.g.*, this is evident in the results for 20%, 25% and 25% noise.) In spite of varying the amount of noise added to the raw aspect data from 1 to 30%, the number of aspects does not vary greatly. The variation in the number of aspects obtained can be attributed to the fact that the sites for the injection of noise have been randomly chosen. (As mentioned before, we randomly choose k% of the total 1962 sites – we have chosen k = 0, 1, 5, 10, 15, 25 and 30.) The percentage of sites where an optimal decision boundary would need to be taken is only a small percentage (under 6%) of the total number of sites. To avoid the exponential time complexity of taking an optimal decision boundary, we perform the following compromise – we consider the errors induced when the region $\mathcal{R}$ is assigned the class-candidates corresponding to each adjacent aspect-candidate. We also consider the error taking into account the *ASSOC_TABLE* estimates, and take the lower of the two. The correctness of Phase II estimates of association pairs is high – above 86%. The table also shows figures for the reduction in the total model base error due to the use of *ASSOC_TABLE* estimates.

3.8 Conclusions

This chapter presents an integrated approach for AAG construction using noisy feature detectors. We handle two important cases of the number of degrees of freedom between the object and the sensor – the 1-DOF case, and the 3-DOF case. We summarize the main features of our scheme as follows:

- We present a classification of errors in raw aspect data
- We propose a new function to evaluate the output of different AAG construction algorithms
- We present a new algorithm for AAG construction with noisy feature detectors. The algorithm transforms noisy raw aspect data into an AAG suitable for use

in an object recognition task. Further, we account for feature detection errors, and store estimates of these for use in robust object recognition (Chapter 4.

- Our method is independent of the object and the feature set

- We characterize the suitability of a feature detector for aspect graphs – in terms of the entire setup, and the given model base.

We present the results of extensive experimentation on a reasonably complex experimental set, in support of our strategy. Since it is not possible to prevent feature detection errors, one may use our strategy for efficient and robust 3-D object recognition. We have reported part of this work in [68]. Papers [70], [64] describe different stages of our work in aspect graph construction.

We conclude this chapter with a small discussion on the effect of errors in aspect data, on a recognition algorithm. An AAG consisting of raw aspect data has a very large number of aspects associated with it. Many of these correspond to feature detection errors, and are very small in size. Hence, the degree of ambiguity associated with a view will be very large, as compared with the case when an error-free AAG is used. The presence of errors in aspect data does not just increase the complexity of the recognition process, it can also cause errors. Errors due to noise often do not occur at a fixed position in an AAG. Hence in a recognition experiment, what may be actually observed at a viewpoint may not correspond to what it was when the raw aspect data was collected. Our algorithms not only aim at reconstructing the corresponding error-free AAG, they also store estimates of feature detection errors. We use these estimates to recover from cases of feature detection errors in our object recognition algorithm (Chapter 4). (In Section 4.4.3, we discuss this point in detail.) We also present the results of experiments comparing the recognition performance with raw aspect data, and the output of our 1-DOF AAG construction algorithm.

Appendix for Chapter 3

To construct a minimum error AAG, we need to enumerate all possible partitionings of the set of sites. We consider all partitionings where the size of a partition is greater than, or equal to Θ_{min}. To each partition in a partitioning, we assign a class-candidate as a label. We consider all such arrangements, with the restriction that no two adjacent partitions have the same class-candidate label. Among all these aspect graphs, we consider the one with the minimum total error. However, this process has exponential time complexity in the size of the AAG, as shown below. Hence, this approach is not feasible.

Proposition 1 *Computation of the minimum error AAG has a time complexity which is exponential in the size of the AAG*

Proof:
The problem of finding the number of possible AAGs for a given set of raw aspect data can be decomposed into two subproblems:

1. Number of ways of partitioning a set of G sites
2. Number of ways of assigning class-candidates from the set of $N_{\mathcal{C}}$ class-candidates to k partitions, with no two adjacent partitions having the same class-candidate label.

Let $P(G, \Theta_{min}, k)$ denote the number of ways of partitioning a set of G sites into k partitions, with the minimum allowable size of a partition being Θ_{min}. Let $Q(k, N_{\mathcal{C}})$ denote the number of ways of assigning class-candidates to k partitions from a set of $N_{\mathcal{C}}$ class-candidates, with the restriction that no two adjacent partitions have the same class-candidate as a label.

The total number of partitions possible with the minimum size restriction is $\kappa = \lfloor \frac{G}{\Theta_{min}} \rfloor$. We first find two bounds on the value of $P(G, \Theta_{min}, k)$. Without any restriction on the size of the resulting partitions, the number is $\binom{G}{k}$. This serves as an

upper bound. We can find a lower bound for $P(G, \Theta_{min}, k)$ by considering aspect sizes to be integral multiples of Θ_{min}. (In case G is not an integral multiple of Θ_{min}, we consider the extra region of size $G - \kappa \cdot \Theta_{min}$ integrated with any one of the partitions of G.) Thus,

$$\binom{\kappa}{k} \leq P(G, \Theta_{min}, k) \leq \binom{G}{k}$$

We can get an upper bound on the value of $Q(k, N_{\mathcal{C}})$ by doing away with the restriction of having different adjacent class-candidate labels. Thus,

$$Q(k, N_{\mathcal{C}}) \leq \sum_{i=2}^{N_{\mathcal{C}}} k^i$$

The number we require is

$$\sum_{k=2}^{\kappa} P(G, \Theta_{min}, k) \cdot Q(k, N_{\mathcal{C}})$$

As shown above, this number is exponential in the number of sites, G. $\square$

Chapter 4

Isolated 3-D Object Recognition using Aspect Graphs: Next View Planning

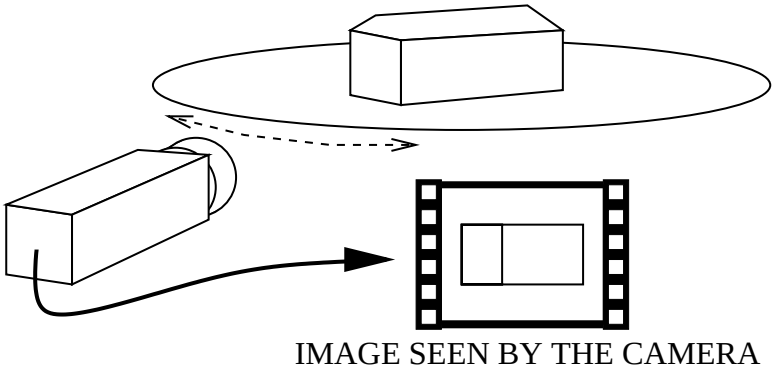
A single view of an object may not contain sufficient features to recognize it unambiguously. This chapter presents a new on-line aspect graph-based recognition scheme based on next view planning for the identification of an isolated 3-D object. We assume a single degree of freedom (rotational) between the object and the (orthographic) camera. The scheme uses a probabilistic reasoning framework for recognition and planning. Our knowledge representation scheme encodes feature-based information about objects as well as the uncertainty in the recognition process. This is used in the probability computations as well as in planning the next view. Our strategy is not dependent on any specific set of features. Numerous experiments with our strategy show the utility of using simple features and suitable planned multiple views, in recognizing fairly complex 3-D objects.

4.1 Aspect Graph-based Object Recognition through Next View Planning

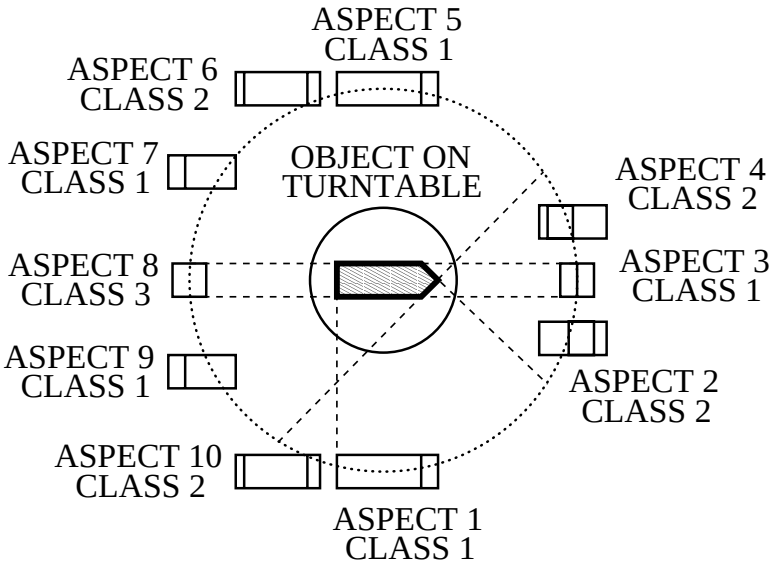
The previous chapter (Chapter 3) presents a strategy for constructing the aspect graph of a 3-D object. The input to the algorithm is raw aspect data – the noisy data collected from all viewpoints in the tessellated viewing space around the object. The output is an aspect graph of the object, along with estimates of feature detection errors. This chapter proposes a new reactive and on-line 3-D object recognition system. The recognition system uses the same noisy feature detectors. The input to the system is the output of the aspect graph construction algorithm, and the feature detection error estimates.

As illustrated in Figure 1.1, a single view of a 3-D object may not contain sufficient features to recognize it unambiguously. We often need to recognize 3-D objects which because of their inherent asymmetry, cannot be completely characterized by an invariant computed from a single view. Further, in recognizing 3D objects from a single view, recognition systems often use complex feature sets [45]. In many cases, it may be possible to achieve the same, incurring less error and smaller processing cost using a simpler feature set and suitably planning multiple observations. A simple feature set is applicable for a larger class of objects than a model base-specific complex feature set. Model base-specific complex features such as 3-D projective invariants have been proposed only for special cases so far (*e.g.*, [213]). The purpose of this chapter is to investigate the use of suitably planned multiple views and simple 2-D invariants for 3-D object recognition. Our recognition strategy is independent of any specific feature set. We assume that there is one rotational degree of freedom between the object and the camera: the viewpoint space is a circle around the object. We also assume an orthographic camera. We illustrate these points in Figure 4.1

In Section 4.2, we propose a new knowledge representation scheme encoding domain knowledge about the object, relations between different aspects, and the cor-



(a)



(b)

Figure 4.1: (a) An example of the 1-DOF case (a single rotational degree of freedom between the object and the camera), and (b) the object with its aspects and classes

respondence of these aspects with feature detectors. This hierarchical knowledge representation scheme not only ensures a low-order polynomial-time complexity of the hypothesis generation process (for the determination of the class corresponding to a view), but also plays an important role in planning the next view. The input to the recognition algorithm is an arbitrary view of an object in our model base. The system probabilistically maps the set of features extracted from this view to the aspect-class corresponding to this view. Our probabilistic reasoning scheme uses the class information to generate hypotheses about the likely aspect corresponding to the view. This is used to determine the corresponding object probabilities. The information from the current view may not be sufficient to identify the object uniquely. Based on the probabilities of the hypotheses generated, our planning algorithm plans the best move to obtain the next view, which would uniquely identify the object. Section 4.3 presents the details of our hypothesis generation mechanism, as well as the next view planning scheme.

4.2 The Knowledge Representation Scheme

In order to use multiple views for an object recognition task, one needs to maintain a relationship between different views of an object, and their relationships with the outputs of different feature detectors. In this section, we propose a knowledge representation scheme that encodes domain knowledge about the objects $O_i \in \mathbf{O}$, relations between different aspects a_{ij}, their classes C_k, and the correspondence of the aspects with the output of the feature detectors. We use our aspect graph construction algorithm of Chapter 3 to get information about the aspects and classes for the objects in the model base. Figure 4.2 illustrates an example of this scheme.

Section 3.1.1 defines the terms *Aspect* and *Class*. Here, we introduce a new term:

Feature-Class A Feature-Class is a set of equivalent aspects defined for one particular feature.

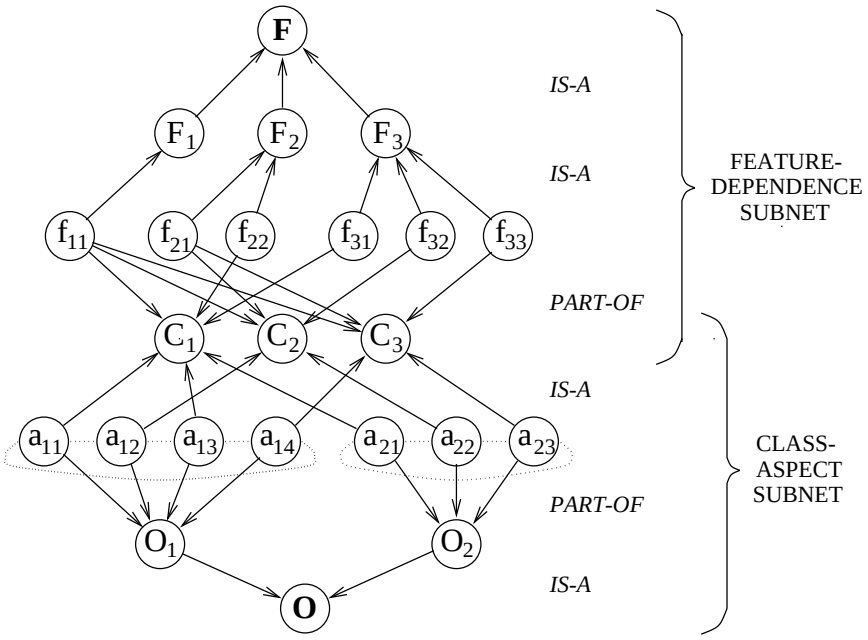


Figure 4.2: The knowledge representation scheme for our aspect graph-based recognition strategy : An example

In Figure 4.2, feature-classes f_{lm} correspond to feature $F_l \in \mathbf{F}$. For example, if F_j detects different types of junctions in images, f_{j1} could correspond to junctions of two line segments, f_{j2} could correspond to junctions of three line segments, and so on. Factors such as noise and non-adaptive thresholds can introduce errors in the feature detection process.

We use this knowledge representation scheme in belief updating as well as in next view planning (Section 4.3).

We emphasize that our system is not based on any specific feature set, unlike that of Dickinson *et al.* [56], [57]. They use volumetric primitives, which are associated with a high feature extraction cost.

The representation scheme consists of two parts:

1. **The Feature-Dependence Subnet**

   In the feature-dependence subnet (Figure 4.2)

   - **F** represents the complete set of features $\{F_j\}$ used for characterizing views

- A Feature node F_j is associated with feature-classes f_{jk}.

  Let p_{jlk} represent the probability that the feature-class present is f_{jl}, given that the detector for feature F_j detects it to be f_{jk}. We define p_{jlk} as the ratio of the number of times the detector for feature F_j interprets feature-class f_{jl} as f_{jk}, and the number of times the feature detector reports the feature-class as f_{jk}. This is another point where we use our aspect graph construction strategy of Chapter 3 – The $N_{\mathcal{C}} \times N_{\mathcal{C}}$ *ASSOC_TABLE* stores information about the number of times one class-candidate is observed as another. We use the *ASSOC_TABLE* to compute p_{jlk} values for each feature-class. The F_j node stores a table of these values for its corresponding feature detector.

- A class node C_i stores its *a priori* probability, $P(C_i)$. A link between class C_i and feature-class f_{jk} indicates that f_{jk} forms a subset of features observed in C_i. This accounts for a *PART-OF* relation between the two. Thus, a class represents an n-tuple $\langle f_{1j_1},\ f_{2j_2},\ \ldots\ f_{nj_n} \rangle$ Since a class cannot be independent of any feature, each class has n input edges corresponding to the n features.

2. **The Class-Aspect Subnet**

   The class-aspect subnet encodes the relationships between classes, aspects and objects (Figure 4.2).

   - **O** represents the set of all objects $\{O_i\}$
   - An object node O_i stores its probability of occurrence, $P(O_i)$
   - An aspect node a_{ij} stores

     its angular extent θ_{ij} (in degrees),

     its probability $P(a_{ij})$ (Section 4.3.2),

     its parent class C_j, and

     its neighbouring aspects

- Aspect a_{ij} has a *PART-OF* relationship with its parent object O_i. Thus, 3-tuple $\langle\ O_i,\ C_j,\ \theta_{ik}\ \rangle$ represents an aspect. Aspect node a_{ij} has exactly one link to any object (O_i) and exactly one link to its parent class C_j.

4.3 The Object Recognition Scheme

We are given an arbitrary view of an object in our model base. We use the aspect graph construction scheme of Chapter 3 and the *ASSOC_TABLE* estimates of feature detection errors, to build the knowledge representation scheme for the given model base. The task at hand is to recognize the given object.

The first step involves recognition of the class corresponding to the given view. This requires the use of different feature detectors to obtain feature-classes corresponding to each feature. (Section 4.2 describes the relation between objects and their aspects, classes, and feature-classes in our knowledge representation scheme.) We now probabilistically map the feature-classes onto the aspect-class corresponding to this view. (Figure 1.1 in Chapter 1 shows such an example). In other words, the class thus obtained could correspond to more than one aspect of different objects in the model base. Our probabilistic reasoning scheme uses the class information to generate hypotheses about the likely identity of the object. Based on the hypothesis probabilities, our planning algorithm plans the best move to obtain the next view, which would uniquely identify the object. The planning process is subject to memory and processing constraints, if any. The camera is moved accordingly. If this does not resolve the ambiguity in the object's identity, the planning process is invoked again – the hypotheses are refined at each stage. The system repeats this process till the object is identified uniquely. Figure 4.3 depicts the interaction of various components of the object recognition system.

The following sections present the three important components of our system, namely

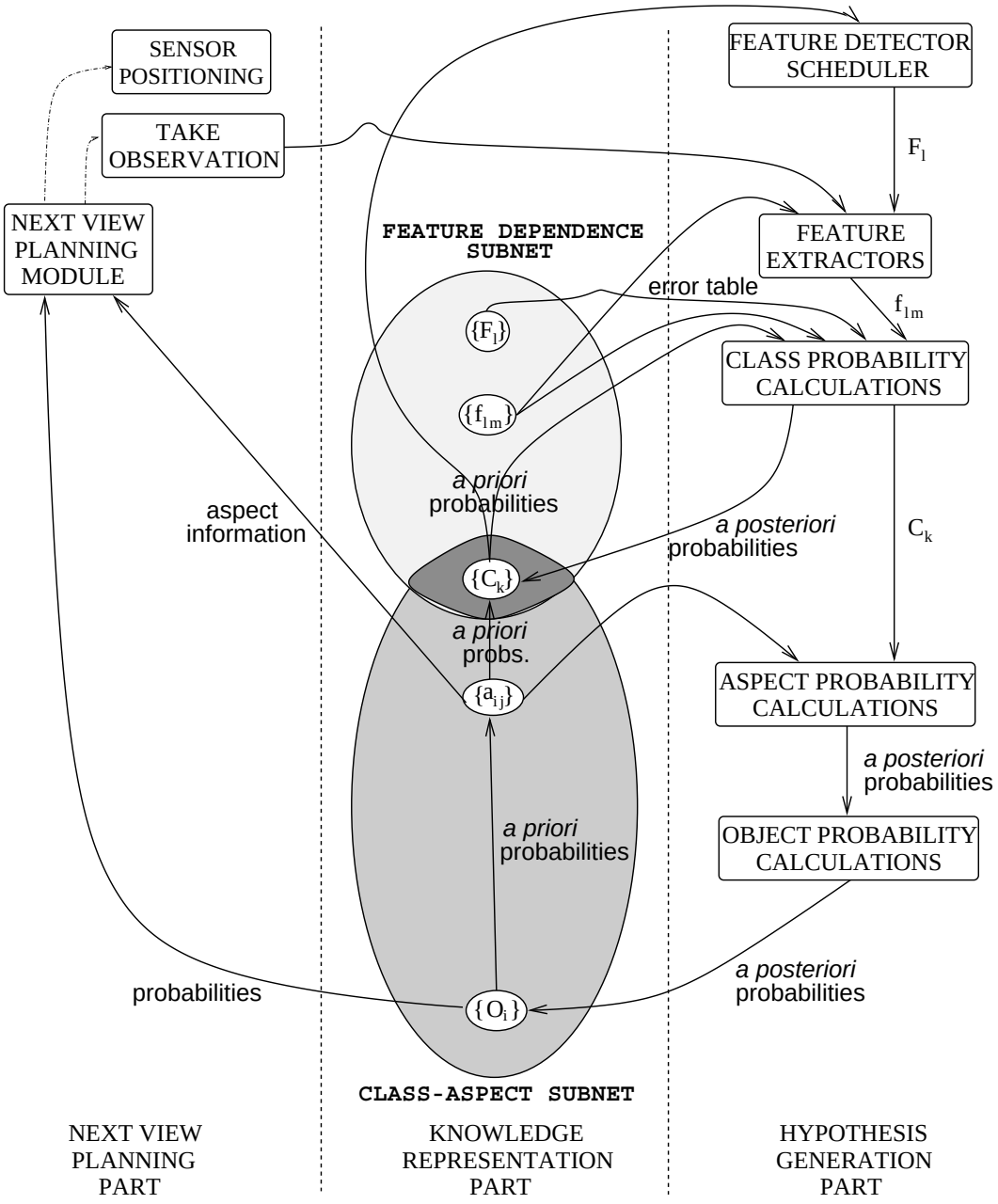


Figure 4.3: Flow diagram depicting the flow of information and control in our system

ALGORITHM identify_class

```
1. compute_a_priori_class_probabilities();
        (* Eq. 4.1; Section 4.3.1  *)
2. fd := identify_feature_detector_to_use();
        (* Section 4.3.1, Ordering of Feature Detectors *)
3. fcl := get_feature_class(image,fd);
        (* Use fd on the image, identify feature class *)
4. compute_a_posteriori_class_probabilities(fcl);
        (* Eqs. 4.2,4.3; Section 4.3.1 Part 2 *)
5. IF the probability of some class is above a
   pre-determined threshold THEN
        pass this class as evidence to the
        object recognition phase, EXIT
6. IF all feature detectors have been used
   AND the probability of no class is above
   the threshold THEN EXIT
7. GO TO Step 2
```

Figure 4.4: The Class Recognition Algorithm

1. Class recognition from a given view of the object, and

2. Object recognition from the identified class

3. Next View Planning

4.3.1 Class Recognition from a Given View of the Object

The input to this phase is an arbitrary view of any object in the model base. The recognition system uses the set of feature-classes from the view to generate hypotheses about the likely identity of the class. This step uses our knowledge representation scheme (Section 4.2), and probabilistic reasoning. Figure 4.4 outlines the class recognition algorithm, which we describe next in detail.

a priori **Class Probability Computations**

Prior to identification of the class corresponding to a given view, we calculate the *a priori* probability of each class. The probability of each class depends upon the probability of the aspects corresponding to it. Let aspects a_{pq} belong to class C_i. $P(a_{pq}|O_p)$ is $\theta_{pq}/360$. Initially, we assume each object to be equally likely to be present *i.e.*, the *a priori* probability of object O_p is $1/n$, where N is the number of objects in the model base. We obtain the *a priori* probability of class C_i as:

$$P(C_i) = \sum_p [P(O_p) \cdot \sum_q P(a_{pq}|O_p)] \tag{4.1}$$

We now describe the computational scheme for the above probability computation, using our knowledge representation scheme. Let N_{F_j}, N_C and N_a denote the number of feature-classes associated with feature detector F_j, the number of classes, and the number of aspects, respectively. We can compute the *a priori* probabilities of all classes in $O(N_C + N_a)$ time using the following steps:

```
(* initialization *)
FOR EACH class C_i DO
    P(C_i) := 0;
(* actual computation *)
FOR EACH object O_p DO
    FOR EACH aspect a_pq under object O_p DO
        BEGIN
        let C_i denote the class node linked to aspect a_pq.
        P(C_i) := P(C_i) + P(a_pq|O_p);
        END
```

The initialization takes time $O(N_C)$. We perform the actual computation for each aspect in each object. Since an aspect belongs to a unique class, this computation can be performed in $O(N_a)$ time.

Ordering of Feature Detectors

A proper ordering of feature detectors speeds up the class recognition process. At any stage, we choose the hitherto unused feature detector for which the feature-class corresponding to the most probable class has the least number of outgoing arcs *i.e.*, the least outdegree. This is done in order to obtain that feature-class which has the largest discriminatory power in terms of the number of classes it could correspond to. For example, in Figure 4.2 if all feature detectors are unused and C_2 has the highest *a priori* probability, F_3 will be tried first, followed by F_2 and F_1, if required.

a posteriori Class Probability Computations

Let the detector for feature F_j report the feature-class obtained to be f_{jk}. Given the *a priori* class probabilities computed in the first step and the feature class f_{jk}, the algorithm now computes *a posteriori* probabilities of all classes C_i.

$$P(C_i|f_{jk}) = \frac{P(C_i) \cdot P(f_{jk}|C_i)}{\sum_m [P(C_m) \cdot P(f_{jk}|C_m)]} \tag{4.2}$$

$P(f_{jk}|C_i)$ is 1 for those classes which have a link from feature-class f_{jk}. It is 0 for the rest. Thus, our class identification and feature detector scheduling scheme facilitates class recognition using a minimum number of feature detectors.

The computation of each $P(C_i|f_{jk})$ for feature detector F_j used takes at most $O(N_C^2)$ time using the following steps:

```
FOR EACH feature-class f_jk under feature detector F_j DO
    BEGIN
    (* denominator calculation *)
    denominator := 0;
    FOR EACH class C_i linked to feature-class f_jk DO
        denominator := denominator + P(C_i);
    (* class probability calculation *)
    FOR EACH class C_i DO
```

```
IF C_i has a link from f_jk
    THEN P(C_i|f_jk) := P(C_i) / denominator;
    ELSE P(C_i|f_jk) := 0;
END
```

The denominator calculation takes $O(N_C)$ time in the worst case, while the class probability calculations take take $O(N_C)$ time. Since the outer loop is executed $O(N_{F_j})$ times, the overall time complexity is $O(N_{F_j} \cdot N_C)$. This can go up to $O(N_C^2)$ in the worst case. Thus, each step in the class probability calculations can be performed in low-order polynomial time.

Class Recognition in the Presence of Feature Detection Errors

For a noise-free case, Equation 4.2 enables us to compute $P(C_i|f_{jk})$: the *a posteriori* probability of class C_i, having (correctly) observed feature-class f_{jk}. However, a feature detector may erroneously report a feature-class f_{jl} as f_{jk}. To handle such cases of feature detection errors, we use p_{jlk} estimates (Section 4.2, page 132). The system computes the *a posteriori* probability of class C_i as follows:

$$P'(C_i) = \sum_l P(C_i|f_{jl}) \cdot p_{jlk} \tag{4.3}$$

where f_{jl}s are feature-classes associated with feature F_j. This summation reduces to one term, $P(C_i|f_{jr}) \cdot p_{jrk}$, since there is only one feature-class under feature F_j. The output of the aspect graph construction algorithm for each object in the model base gives us *ASSOC_TABLE* values (Chapter 3), from which we calculate p_{jlk} values (Section 4.2).

We use the following steps to compute the *a posteriori* class probabilities in $O(N_{F_j})$ time:

```
(* the observed feature-class is f_jk *)
FOR EACH feature-class f_jr under feature node F_j DO
    FOR EACH class C_i with a link from f_jr DO
        P'(C_i) := P(C_i|f_jr) · p_jrk
```

In the case where there are no spurious feature-classes (ones not associated with any class), the time complexity is $O(N_C)$, since the above computations are performed for each class.

Thus, our algorithm can handle cases of feature detection errors, and recover from them. If the probability of any class is above a pre-determined threshold, the system passes this information as evidence to the object recognition phase. Otherwise, it schedules the next feature detector as above, and repeats the process.

4.3.2 Object Identification Given the Identified Class

The input to the object identification phase is an arbitrary view of an object in the model base. The object identification phase uses the class recognition algorithm (Section 4.3.1) as a module. Figure 4.5 presents our overall object identification algorithm. In what follows, we describe the various steps of the algorithm in detail.

Object Probability Calculations in the First Phase

Before the system takes an image of the given view of the object, we calculate the *a priori* probabilities of each object in the model base.

$$P(a_{j_pk_p}) = P(O_{j_p}) \cdot P(a_{j_pk_p}|O_{j_p}) \tag{4.4}$$

The probability of each of the N objects is initialized to $1/N$ before the first observation. $P(a_{j_pk_p}|Oj_p)$ is $\theta_{j_pk_p}/360$. *a priori* aspect probability calculations take $O(N_a)$ time.

The system now takes an image of the object. The class recognition algorithm identifies the class corresponding to a given view of the object. The class observed could have come from more than one aspect, each with its own range of positions within the aspect. The system now computes *a posteriori* probabilities as follows.

Let the class recognition phase report the observed class to be C_i. Let us assume that C_i could have come from aspects $a_{j_1k_1}$, $a_{j_2k_2}$, $\ldots a_{j_mk_m}$, where all j_1, j_2, $\ldots j_m$

ALGORITHM identify_object

```
            (* ------ FIRST PHASE ------ *)
1.  initialize_object_probabilities();
            (* Initialize to 1/N *)
2.  image:=get_image_of_object();
3.  class:=identify_class(image);
            (* Section 4.3.1 *)
    IF class=UNKNOWN THEN exit;
4.  search_tree_root:=construct_search_tree_node(class,0);
5.  compute_object_probabilities(search_tree_root);
            (* Eqs. 4.5,4.6 *)
6.  IF the probability of some object is above a
    pre-determined thresh.  THEN exit & declare success;
7.  expand_search_tree_node(search_tree_root,0,class);
            (* Section 4.3.3 *)
    best_leaf:=get_best_leaf_node(search_tree_root);
            (* ------ SECOND PHASE ------ *)
        previous:=search_tree_root;
        expected:=best_leaf;
8.  angle:=compute_angle_to_move_by(expected,previous);
    make_movement(angle);
    image:=get_image_of_object();
9.  class:=identify_class(image);
    IF class=UNKNOWN THEN exit;
10. new_node:=construct_search_tree_node(class,angle);
11. compute_object_probabilities(new_node);
12. IF the probability of some object is above a
    pre-determined thresh.  THEN exit & declare success;
13. expand_search_tree_node(new_node);
    best_leaf:=get_best_leaf_node(new_node);
        previous:=new_node;
        expected:=best_leaf;
14. GO TO step 8
```

Figure 4.5: The Object Recognition Algorithm

are not necessarily different. We obtain the *a posteriori* probability of aspect $a_{j_i k_l}$ given this evidence using the Bayes rule:

$$P(a_{j_l k_l}|C_i) = \frac{P(a_{j_l k_l}) \cdot P(C_i|a_{j_l k_l})}{\sum_{p=1}^{m}[P(a_{j_p k_p}) \cdot P(C_i|a_{j_p k_p})]} \tag{4.5}$$

$P(C_i|a_{j_l k_l})$ is 1 for aspects with a link to class C_i, 0 otherwise. Finally, we obtain the *a posteriori* probability

$$P(O_{j_p}) = \sum_{l} P(a_{j_p k_l}|C_i) \tag{4.6}$$

where aspects $a_{j_p k_l}$ belong to class C_i.

We use the following steps for object probability calculations.

1. *a priori* aspect probability calculations take $O(N_a)$ time, since Equation 4.4 has to be calculated for each aspect associated with each object.

2. Let the class observed be C_i. Then, *a posteriori* aspect probability computations take $O(N_a)$ time as can be seen in the following pseudo-code:

```
(* calculate denominator *)
denominator := 0;
FOR EACH aspect a_jk linked to class C_i DO
    denominator := denominator + P(a_jk);
(* actual aspect probability calculations *)
FOR EACH object O_jl DO
        FOR EACH aspect a_jlkl belonging to object O_jl DO
            IF a_jlkl has a link to class C_i
                THEN P(a_jlkl|C_i) := P(a_jlkl) / denominator;
                ELSE P(a_jlkl|C_i) := 0;
```

The denominator calculations take $O(N_a)$ time in the worst case. The other computations take $O(N_a)$ time due to the outer loops being executed for all aspects in the model base.

3. The final object probability calculations (Equation 4.6) takes $O(N_a)$ time, because the probability of an object is the sum of the probabilities of all its constituent aspects. Thus, each step in the object probability calculations can be performed in linear time.

An important feature of our system is the role played by the hierarchical knowledge scheme in hypothesis generation. In Hutchinson and Kak's work [106], feature matches are directly used for generating object hypotheses. Although their formulation ensures polynomial-time implementation of Dempster's combination rule, their system still incurs the overhead of intersection calculations in refining frames of discernment, and incorporating consistency checks.

In our hierarchical scheme, the link conditional probabilities (representing relations between nodes) themselves enforce consistency checks at each level of evidence. The feature evidence is progressively refined as it passes through different levels in the hierarchy, leading to simpler evidence propagation and less computational cost.

If the probability of some object is above a threshold the algorithm reports a success, and stops. In our experiments with the model base consisting of polyhedral objects (Section 4.4.1), we take this to be 0.87. *However, if the probability of no object is above the threshold, this implies that the information from the given view of the object is not sufficient to recognize it unambiguously. We have to take another view of the object, which will best disambiguate between the competing objects.* The next section describes our next view planning scheme in detail.

4.3.3 Next View Planning

Next View Planning is required when the probability of no object is above a particular threshold. This means that the class observed could have come from more than one aspect, each with its own range of positions within an aspect. Due to this ambiguity, one has to search for the best move to disambiguate between these competing aspects subject to memory and processing limitations, if any. The planning process aims to

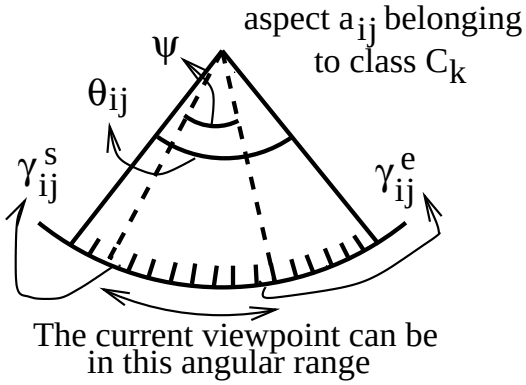


Figure 4.6: The parameters characterizing the state of the recognition system (Section 4.3.3)

determine a move from the current step, which would uniquely identify the given object. We pose the planning problem as that of a forward search in the state space which takes us to a state in which the aspect list corresponding to the class observed has exactly one node. We use a search tree to find a state in the search space corresponding to a unique aspect. The parameters describing the state (a search tree node) are as follows (Figure 4.6):

- the unique class observed for the angular movement made so far,
- the aspects possible for this angle-class pair, and
- for each aspect, the range of positions possible within it ($\gamma_{ij}^s - \gamma_{ij}^e$). γ_{ij}^s and γ_{ij}^e denote the two positions within aspect a_{ij} where the current viewpoint can be, as a result of the movement made thus far.

Here, $\gamma_{ij}^s \leq \gamma_{ij}^e$; and γ_{ij}^s, and $0 \leq \gamma_{ij}^e \leq \theta_{ij}$, where θ_{ij} is the angular extent of aspect a_{ij}.

A leaf node is one which has a unique aspect associated with it, or corresponds to a total movement of 360 degrees from the root node, or more. The aim is to find the move that will best disambiguate between the competing objects, subject to memory and processing limitations, if any.

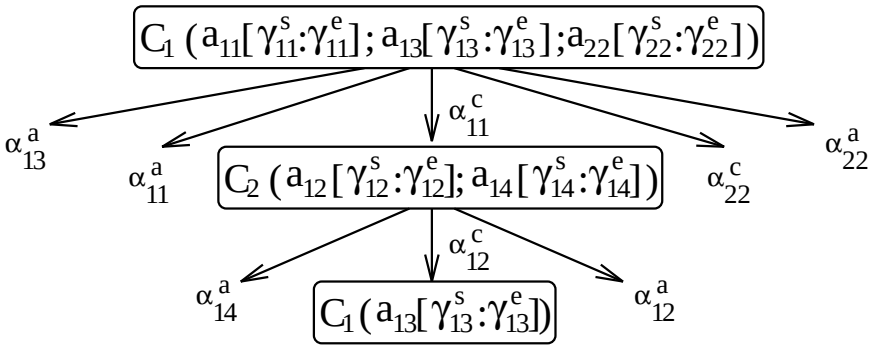


Figure 4.7: A Partially Constructed Search Tree

Choosing the right step size of movement

For the search tree node corresponding to an observed view, we do not know the aspect corresponding to the observed class. Further, even the position within the aspect is not known. An infinite number of moves is possible from a particular state. Even for discrete movement steps, the number of moves from the observed node is very large, equal to the number of steps needed to move around an object (the number of sites in the tessellated viewpoint space).

Hence, an important parameter is the *step size of movement*. If it is too small, then we may remain in the same aspect - incurring image processing cost. A large step size, on the other hand may cause us to miss a unique aspect corresponding to an object. In view of these facts, we categorize moves from a particular viewpoint, as follows:

Primary Move A primary move represents a move from an aspect by α, the minimum angle needed to move out of it.

Auxiliary Move An auxiliary move represents a move from an aspect by an angle corresponding to the primary move of another competing aspect.

Figure 4.7 shows an example of a partially constructed search tree. Let α_{ij}^c and α_{ij}^a represent the minimum angles necessary to move out of the current assumed aspect in the clockwise and anti-clockwise directions, respectively. Three cases are possible:

1. **Type I move**: α_{ij}^c and α_{ij}^a both take us out of the current aspect to a single aspect in each of the two directions – a_{ip} and a_{iq}, respectively. We construct search tree nodes corresponding to both moves.

2. **Type II move**: Exactly one out of α_{ij}^c and α_{ij}^a takes us to a single aspect a_{ip}. For the other direction, the aspect we would reach depends upon the initial position (which may lie between γ_{ij}^s and γ_{ij}^e, both points inclusive) in the current aspect. We construct a search tree node corresponding to the former move.

3. **Type III move**: Whether we move in the clockwise or the anti-clockwise direction, the aspect reached depends on the initial position in the current aspect. We choose the move which leads us to the side with the largest angular range possible in any reachable aspect.

We expand a non-leaf node by generating child nodes corresponding to primary moves for all competing aspects in its aspect list. We can also generate additional child nodes by considering auxiliary moves (logical moves out of the assumed aspect by angles corresponding to primary moves for other aspects). We assign a code 0 to Type I and II primary moves and 1 to Type II auxiliary moves. Type III primary moves moves get a code of 2, and Type III auxiliary moves, 3. The weight associated with a node is $4^i \cdot Code$, where i is the depth of the node in the search tree. We use three levels of filtering to determine the best leaf node. First, we consider those on a path from the most probable aspect(s) corresponding to the previously observed node. Among these, we consider those having paths of least weight. From these, we finally select one with the minimum total movement.

Since we are constructing a node for each observation taking into account all aspects possible for the angular move made, it may still be possible that the observed node created is present as an ancestor node. This is due to the fact that we are taking a move for the most probable aspect expected at each stage, rather than deterministically constructing the entire search tree and performing an exhaustive search. We can have two cases here. If the constructed node is the same as an ancestor

which is not the root node, we consider moves out of this node corresponding to the moves out of the ancestor node's parent to avoid this cycle. In the other case, we take a random move within the 360 degree restriction, and continue.

The Second Phase of the Object Recognition Algorithm

The search process finds the best move to resolve the ambiguity associated with this view. The system takes the move corresponding to the best leaf node in the search tree, and the class identification module is called again. We construct a new search tree node corresponding to this move.

We again need to calculate the *a priori* probability of each class C_i. This process is similar to Equation 4.1 of Section 4.3.1, except for the following two differences. The *a posteriori* object probabilities $P(O_{j_p})$ of Equation 4.6 of Section 4.3.2 serve as the *a priori* probabilities for the class identification part here. Additionally, we have to *account for the movement in the probability calculations.* For example, a particular movement may preclude the occurrence of some aspects for a given class observed. We compute the value of $P(a_{j_pk_p}|O_{j_p})$ as follows:

$$P(a_{j_pk_p}|O_{j_p}) = \phi_{j_pk_p}/360 \tag{4.7}$$

where $\phi_{j_pk_p}$ $(0 \leq \phi_{j_pk_p} \leq \theta_{j_pk_p})$ represents the angular range possible within aspect $a_{j_pk_p}$ for the move(s) taken to reach this position. Due to the movement made, we could have observed only m $(0 \leq m \leq r)$ aspects out of a total of r aspects belonging to class C_i. In our hierarchical scheme, the link conditional probabilities (representing relations between nodes) themselves enforce consistency checks at each level of evidence. The feature evidence is progressively refined as it passes through different levels in the hierarchy, leading to simpler evidence propagation and less computational cost. This is an advantage of our scheme over that proposed in [106].

Depending on the observed class C_i, we calculate the *a posteriori* probability of aspect $a_{j_lk_l}$, and finally the *a posteriori* probability of each object O_{j_p} using Equation 4.5 and Equation 4.6 of Section 4.3.2. If the probability of some object is above

the threshold, then the algorithm terminates. Otherwise, the search process is repeated. This illustrates the reactive nature of our strategy. We discuss this further, along with issues relating to the finiteness of the search procedure and scalability, in the next section.

4.3.4 The Object Identification Algorithm: A Discussion

Search tree node expansion is always finite due to the following reasons:

1. Even if all the aspects in the competing objects are the same, the search tree construction stops when the total movement along a path is 360^o.

2. The number of aspects is finite. No aspect is repeated along a path in the process of node creation *i.e.*, there are no cycles. Thus, there can be no search tree node expansion indefinitely oscillating between a set of aspects.

Given the combinatorial nature of the problem, it is often not feasible to construct the entire search tree off-line. This is especially true if the number of competing aspects for a view is quite large. In contrast, our planning process is on-line and reactive. Taking primary moves helps us to prune the search space by selecting an appropriate step size of movement. This eliminates redundant moves (which would have incurred large image processing cost). Auxiliary moves additionally provide better aspect resolution. Further, our planning strategy is scalable – one can plan with primary moves alone (greater pruning of the search space), or with a combination of primary and auxiliary moves (which have greater discriminatory power), depending on memory and processing limitations. Our robust class recognition algorithm can recover from many feature detection errors at the class recognition phase itself (Section 4.3.1). Our planning scheme is global – its reactive nature incorporates all previous movements and observations in the probability calculations (Section 4.3.2), as well as in the planning process. An off-line system does not have the capability of recovering from feature detection errors. This is so because all the planning is done off-line, and

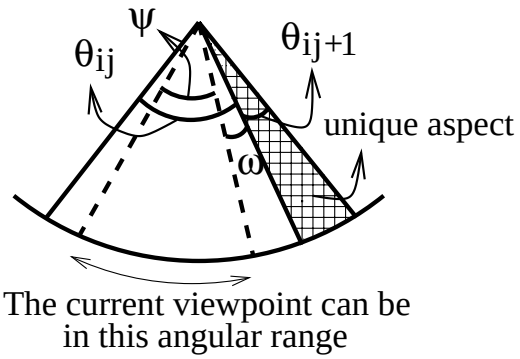


Figure 4.8: A case when our algorithm is not guaranteed to succeed (Section 4.3.4)

the system does not have any means of dealing with an unplanned situation. The reactive nature of our strategy incorporates all previous movements and observations in the probability calculations (Sections 4.3.1 and 4.3.2), as well as in the planning process. If the obtained view indeed corresponds to the most probable aspect at a particular stage, then our search process is guaranteed to perform aspect resolution and uniquely identify the object in the following step, assuming no feature detection errors. Even if the view does not correspond to the most probable aspect, the list of possible aspects a given view could correspond to, is refined at each observation stage (avoiding the exponential time complexity of having to expand out the entire search tree at one go). The planning process is initiated with the new aspect list. This illustrates the reactive nature of our planning strategy.

Assuming no feature detection errors, our algorithm is guaranteed to succeed except in three cases. The first is for objects with the same aspect structure (*i.e.*, the layout of classes in the aspect graph) but different aspect angles. Further, our strategy does not handle the case when the aspect angles are greater than or equal to 180 degrees. Figure 4.8 shows an example of the third case. Let us suppose that we have to move anti-clockwise. Let ψ denote the angular extent of the smallest aspect observed so far. The current viewpoint lies in this angular range. Let a_{ij+1} be a unique aspect for the assumed object. The anti-clockwise movement will be by an angle $\psi + \omega$. If $\psi + \omega > \theta_{ij+1}$, we may miss this unique aspect altogether.

Bounds on the Number of Observations

It is instructive to consider bounds on the number of observations required to disambiguate between a set of n aspects. Let us evaluate these for a simple deterministic case, to serve as a benchmark.

Let us assume that the class initially observed could correspond to n aspects belonging to the objects in the model base. We also assume the space of viewing positions around the object to be quantized to N_V. Each of the N_V possible moves from the starting position partitions the aspect list associated with a class node into equivalence classes, each of which is an aspect-class. Due to the uncertainty in position within an aspect, any movement from an aspect could lead us to more than one aspect. Let the upper bound on the number of aspects reachable from any given aspect be ν. For the sake of simplicity, however let us assume ν to be 1. We also assume no errors in either movement or image processing. Given these assumptions, we state the following proposition:

Proposition 2 *If the assumptions made above hold, the average number of observations required to uniquely identify the given object is $O(log_e n)$, where n is the number of aspects the initially observed class could correspond to.*

Proof:

Let $T(n)$ denote the number of observations required to disambiguate between n aspects. These n aspects are obtained as a consequence of the first observation and the consequent processing. Each of the N_V moves partitions this set of aspects into equivalence classes, each of which is an aspect-class. We choose a move that partitions the initial aspect set into more than one equivalence class. (A distinguishing move has to exist for the object set, else all objects in the model base would be indistinguishable from one another.) If the size of the aspect list in one such equivalence class is j, the expected additional number of observations is $T_{avg}(j)$, where $1 \leq j < n$. We assume j to take on any of the values 1 to $n-1$ with equal probability. We have

$T_{avg}(n) = 1 + \frac{\sum_{j=1}^{n-1} T_{avg}(j)}{n-1}$

Further, $T_{avg}(1) = 1$.

We now use Mathematical Induction to prove that $T(n) = O(log_e n)$.

Induction Hypothesis: $T_{avg}(n) = O(log_e n)$, where $1 \leq n < m,\ m > 2$.

Induction Step: $T_{avg}(m) = 1 + \frac{\sum_{j=1}^{m-1} T_{avg}(j)}{m-1}$

$= 1 + \frac{T_{avg}(1)+T_{avg}(2)+ \cdots +T_{avg}(m-1)}{m-1}$

Since $T_{avg}(n) = O(log_e n)$, we may write $T_{avg}(n) = u_n \cdot log_e n + v_n$, where u_n and v_n are constants.

We now invoke the Induction hypothesis on the summation:

$T_{avg}(m) = (1 + (\frac{1}{m-1}) \cdot (\sum_{j=2}^{m-1} v_j) + \frac{1}{m-1}) + (\frac{1}{m-1}) \cdot (\sum_{j=2}^{m-1} (u_j \cdot \log_e j))$

We replace the term $(\frac{1}{m-1}) \cdot (\sum_{j=2}^{m-1} v_j)$ by a constant q_1. Further, we replace each u_j by the largest of all u_j's, $2 \leq j < m$. Let us call this constant q_0. This results in the following inequality:

$T_{avg}(m) \leq (1 + q_1 + \frac{1}{m-1}) + (\frac{q_0}{m-1}) \cdot (\sum_{j=2}^{m-1} \log_e j)$

$\leq (1 + q_1 + \frac{1}{m-1}) + (\frac{q_0}{m-1}) \cdot (\int_2^m \log_e x\ dx)$

$\leq (1 + q_1 + \frac{1-2q_0 log_e 2}{m-1}) + q_0 \cdot (1 + \frac{1}{m-1}) \cdot (log_e m)$

Since $m > 1$, we can replace the terms with $(m-1)$ in the denominator by the corresponding terms multiplied by $(m-1)$ without affecting the inequality. Thus, $T_{avg}(m) \leq q_3\, log_e m + q_4$, where q_2 and q_3 are constants. Hence, $T_{avg}(m) = O(log_e m)$ $\Box$.

4.4 Experimental Results and Discussion

Our experimental set is the same as that described in Section 3.7. We have a camera connected to a MATROX Image Processing Card. We also have a stepper motor-controlled turntable on which the object to be recognized, is placed. The turntable moves by 200 steps to complete a 360 degree movement.

We have experimented with two sets of objects:

1. *Model Base I: 7 Aircraft Models* (Figure 4.9)

   We use as features,

   (a) the number of horizontal lines ($\langle h \rangle$),

   (b) the number of vertical lines ($\langle v \rangle$), and

   (c) the number of circles ($\langle c \rangle$).

   We represent a class as $\langle hvc \rangle$. We have chosen this relatively feature-rich model base for two reasons. First, we wish to demonstrate the effectiveness of our AAG construction algorithm on raw aspect data with very low smoothness (Chapter 3). The second reason is to show the effectiveness of using simple and robust features with multiple views for recognizing complex 3-D objects.

2. *Model Base II: 8 Polyhedral Objects* (Figure 4.10)

   We use as features,

   (a) the number of horizontal lines ($\langle h \rangle$),

   (b) the number of vertical lines ($\langle v \rangle$), and

   (c) the number of non-background segmented regions in an image ($\langle r \rangle$).

   We represent a class as $\langle hvr \rangle$. We have chosen this model base so that most objects have more than one view in common. The list of possible aspects associated with one initial view is quite large here (18, compared with 10 for the first set).

As described in Section 3.7, we use Hough transform-based line and circle detectors. For getting the number of regions in the object, we use sequential labeling on a thresholded gradient image.

Here, we reiterate that our scheme is independent of any particular set of features. While we use simple features for the purpose of illustration, one may also use other features such as those based on

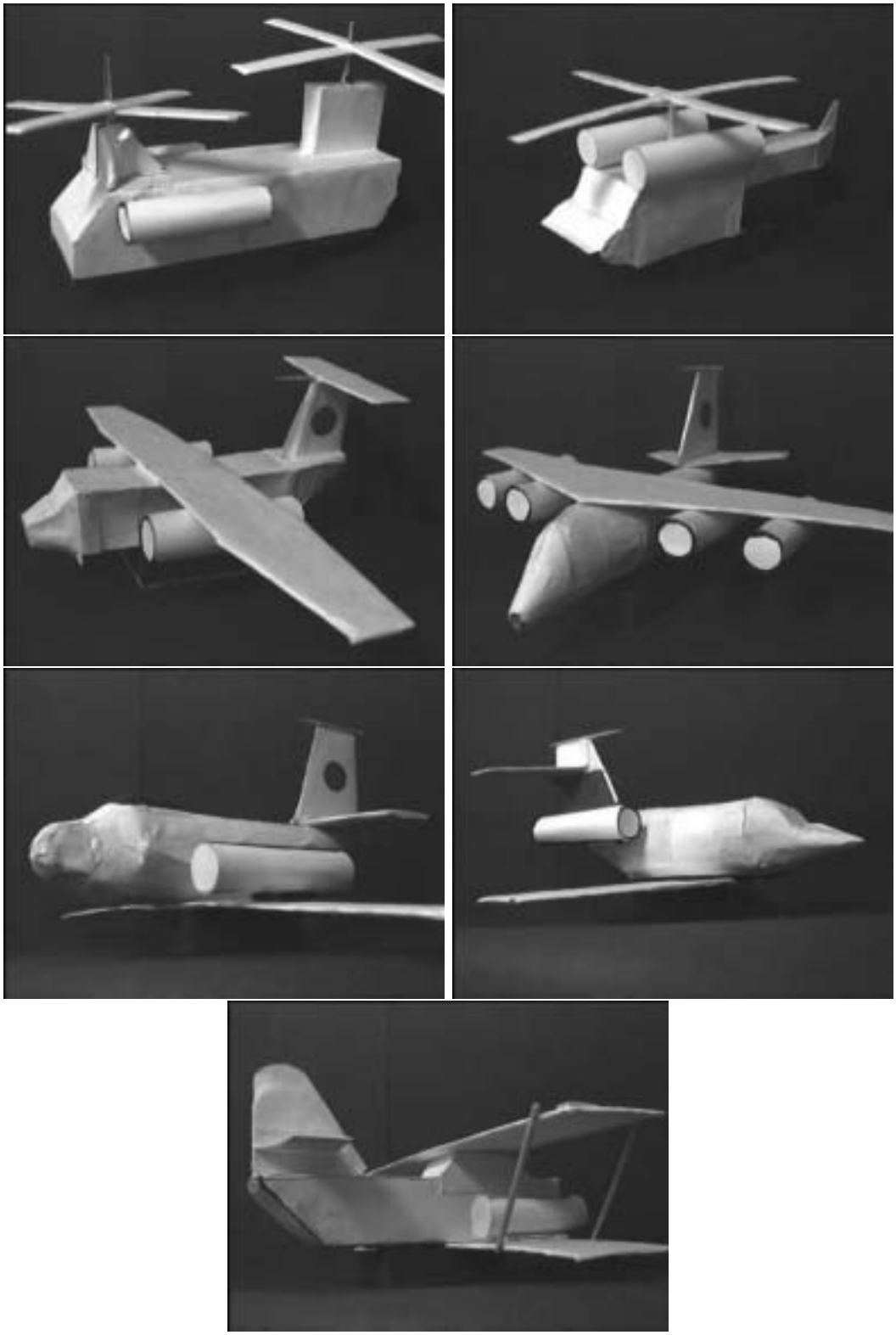


Figure 4.9: Model Base I: The objects (in row major order) are heli_1, heli_2, plane_1, plane_2, plane_3, plane_4, and biplane.

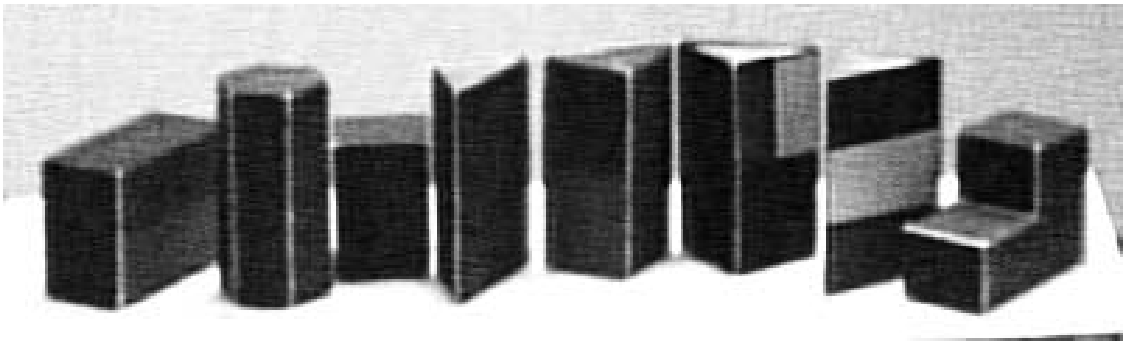


Figure 4.10: Model Base II: The objects (from left) are O_1, O_2, O_3, O_4, O_5, O_6, O_7 and O_8, respectively.

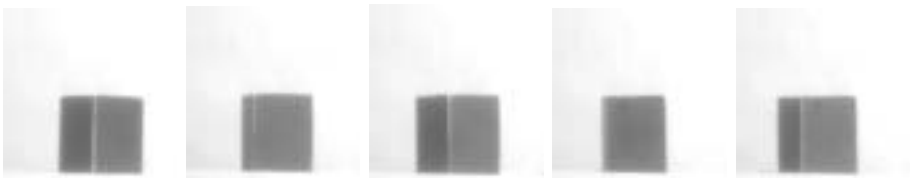
- 2-D projective invariants (*e.g.*, [175]),
- texture (*e.g.*, [9]),
- colour and multidimensional receptive field histograms (*e.g.*, [188], [99], [76], [179]), or
- reflectance ratios ([149])

In this section, we describe two classes of experiments:

1. Sections 4.4.1 and 4.4.2: Recognition experiments related to ideas presented in Chapter 4, and
2. Section 4.4.3: Experiments showing a comparison of the action of the recognition algorithm on raw aspect data, and on the AAGs constructed using the algorithm of Chapter 3

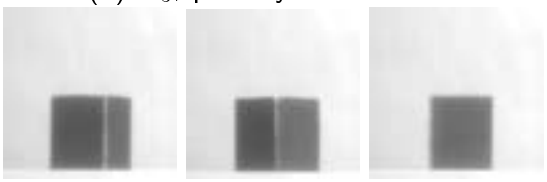
4.4.1 Object Recognition Experiments with Model Base II: Polyhedral Objects

Figures 4.11 and 4.12 show some experiments with the objects in the polyhedral objects model base. For Figure 4.11, the initial class observed in each case is $\langle 232 \rangle$,



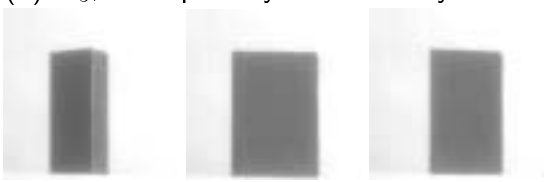
$\langle 232\rangle \xrightarrow{-40} \langle 231(221)\rangle \xrightarrow{-12} \langle 232\rangle \xrightarrow{-34} \langle 221\rangle \xrightarrow{-11} \langle 232\rangle$

(a) O_3, primary moves alone



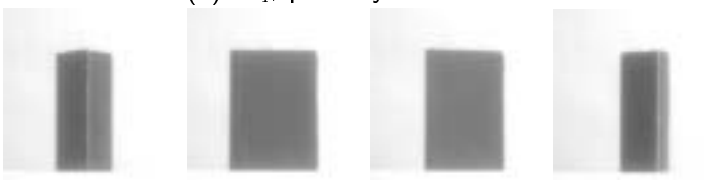
$\langle 232\rangle \xrightarrow{-40} \langle 232\rangle \xrightarrow{-77} \langle 221\rangle$

(b) O_3, both primary and auxiliary moves



$\langle 232\rangle \xrightarrow{-40} \langle 221\rangle \xrightarrow{-12} \langle 221\rangle \xrightarrow{-2} \langle 221\rangle$

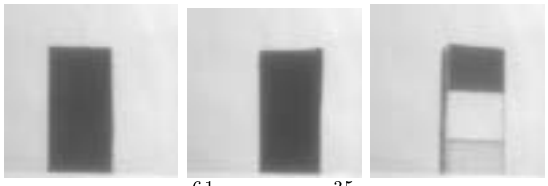
(c) O_4, primary moves alone



$\langle 232\rangle \xrightarrow{-40} \langle 221\rangle \xrightarrow{-12} \langle 221\rangle \xrightarrow{-33} \langle 221\rangle$

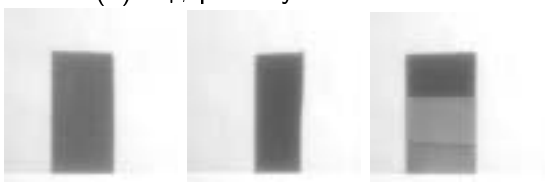
(d) O_4, both primary and auxiliary moves

Figure 4.11: Some experiments with Model Base I: initial class $\langle 232\rangle$. The numbers above the arrows denote the number of turntable steps. A negative sign indicates a clockwise movement. (The figure in parenthesis shows an example of recovery from feature detection errors)



$\langle 221 \rangle \xrightarrow{61} \langle 221 \rangle \xrightarrow{35} \langle 423 \rangle$

(a) O_7, primary moves alone

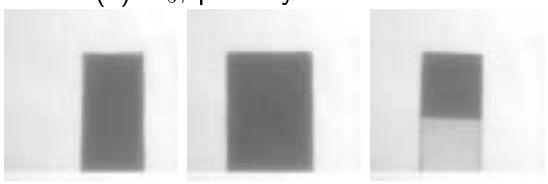


$\langle 221 \rangle \xrightarrow{77} \langle 221 \rangle \xrightarrow{72} \langle 423 \rangle$

(b) O_7, both primary and auxiliary moves

$\langle 221 \rangle \xrightarrow{61} \langle 221 \rangle \xrightarrow{35} \langle 221 \rangle \xrightarrow{-8} \langle 221 \rangle \xrightarrow{-30} \langle 221 \rangle$

(c) O_5, primary moves alone

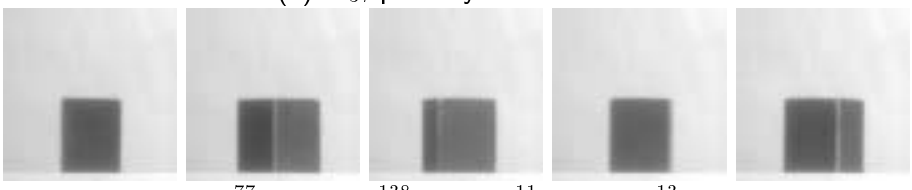


$\langle 221 \rangle \xrightarrow{77} \langle 221 \rangle \xrightarrow{72} \langle 322 \rangle$

(d) O_5, both primary and auxiliary moves

$\langle 221 \rangle \xrightarrow{61} \langle 232 \rangle \xrightarrow{16} \langle 232 \rangle$

(e) O_3, primary moves alone



$\langle 221 \rangle \xrightarrow{77} \langle 232 \rangle \xrightarrow{-138} \langle 232 \rangle \xrightarrow{11} \langle 221 \rangle \xrightarrow{13} \langle 232 \rangle$

(f) O_3, both primary and auxiliary moves

Figure 4.12: Some experiments with Model Base I: initial class $\langle 221 \rangle$. The numbers above the arrows denote the number of turntable steps. A negative sign indicates a clockwise movement

while it is $\langle 221 \rangle$ in Figure 4.12. We have considered cases: taking only primary moves, as well as planning with both primary and auxiliary moves. We make the following observations:

Primary and Auxiliary Moves

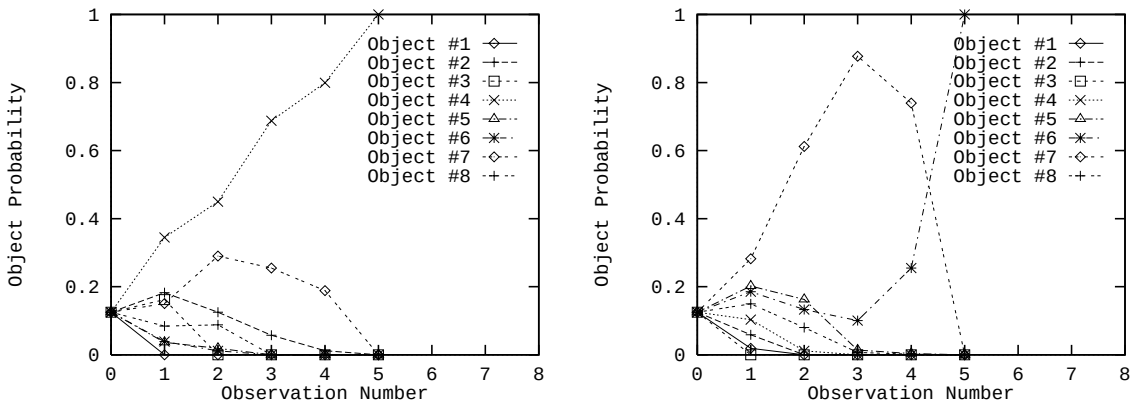
In most cases, the number of image processing steps required is less in the latter case as compared to the former. Planning with both primary and auxiliary moves thus in general, reduces image processing overheads at the expense of additional search time and memory requirements. When memory and search time are limited, the planning process may use primary moves alone. An interesting case is observed in Figures 4.12(e) and (f) - an opportunistic case when the number of steps with primary moves is less than the one with both primary and auxiliary moves. At step 2, the move planned was not for the aspect eventually observed in step 3. However, this particular sequence of moves turns out to be unique for object O_3.

Ordering of Feature Detectors

The third image in Figure 4.11(a) shows an the advantage of our scheduling of feature detectors. The line detector reports the feature-class present to be $\langle 23 \rangle$. For the objects in our model base, this could correspond to classes $\langle 232 \rangle$ and $\langle 233 \rangle$. Our probability calculations account for the movement taken around the object. The probability of class $\langle 232 \rangle$ for the movement made so far exceeds the class probability threshold(0.87). Hence, the system does not need to use the other feature detector.

Recovery from Feature Detection Errors

The second image in Figure 4.11(a) shows a situation where the system recovers from an error in the feature detection process. Due to the thresholds we use, the correct class is $\langle 221 \rangle$. The line detector, however reports the probabilities of classes $\langle 221 \rangle$ and $\langle 231 \rangle$ as 0.004 and 0.856, respectively. The probability of no class is above the



(a) For Figure 4.11(a) (b) For Figure 4.12(c)

Figure 4.13: Variation of object probabilities: two examples

threshold. The other feature detector is now scheduled, which reports the number of regions to be 1. The probability calculations of Equation 4.3 result in the probabilities of the two as 0.997 and 0.002, respectively.

Variation of Object Probabilities

Figure 4.13 shows the variation in object probabilities with each observation. The two cases shown here are for the moves in Figure 4.11(a) and Figure 4.12(c). Figure 4.13(a) shows the probability of object O_4 steadily rising till it gets to 1, in the 5th move. The probabilities of all objects change as each view accounts for new evidence. Figure 4.13(b) shows an interesting case of an object O_7, whose probability is the highest till observation number 3. The successive evidence lessens the probability of this object, and increases that of another object O_6. This evidence results in the probabilities of all objects which had non-zero probabilities thus far, dropping to zero. The next move results in a view which is unique to object O_6.

Some Sample Search Tree Details

We now consider some cases in detail. For each row in Figure 4.11 , the initial view could have come from 18 aspects belonging to objects in our model base and for

Figure 4.12, the corresponding number is 17. For the strategy involving primary moves alone, the total number of search tree nodes generated for Figures 4.11(a), 4.11(c), 4.12(a), 4.12(c) and 4.12(e) are 53, 48, 34, 48 and 39, respectively. For the strategy involving both primary and auxiliary moves (Figures 4.11(b), 4.11(d), 4.12(b), 4.12(d) and 4.12(f)), the corresponding numbers are 324, 279, 127, 127 and 151, respectively. Let us consider Figure 4.12(d). The algorithm plans a move of 77 steps. The second observation reports the number of aspects possible as 6. The next move by 72 steps corresponds to a unique aspect.

Average Number of Observations for a Given Number of Competing Aspects

The upper part of Table 4.1 gives an idea of the average number of observations for a given number of competing aspects for the experiments with the first model base. The average is computed over 46 experiments.

4.4.2 Object Recognition Experiments with Model Base I: Aircraft Models

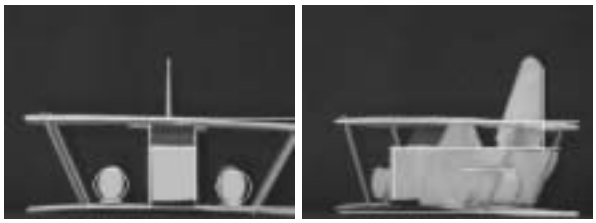
Figures 4.14, 4.15 and 4.16 show some results of experimentation with the objects in the aircraft model base. The initial classes observed in these figures are $\langle 332 \rangle$, $\langle 411 \rangle$ and $\langle 410 \rangle$, respectively.

Primary and Auxiliary Moves

For the experiments with the second model base, in most cases the number of observations required with primary observations alone, and with both primary and auxiliary moves are the same. This can be attributed to the fact that the degree of uncertainty associated with a view for an object in this model base is less than one for the first model base (The maximum number of competing aspects here is 10, compared with 18 in the first case).

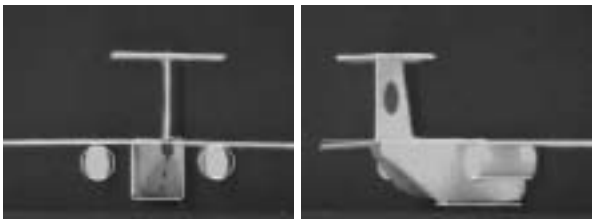
Model Base II: Polyhedral Objects		
Number of	*Average number of observations*	
Competing Aspects	*Primary Moves Alone*	*Primary and Auxiliary Moves*
5	2.00	2.50
17	3.09	3.07
18	4.00	3.38
Model Base I: Aircraft Models		
Number of	*Average number of observations*	
Competing Aspects	*Primary Moves Alone*	*Primary and Auxiliary Moves*
4	2.00	2.00
5	2.00	2.09
7	2.00	2.00
9	2.00	2.00
10	2.67	2.67

Table 4.1: The average number of moves for a given number of competing aspects



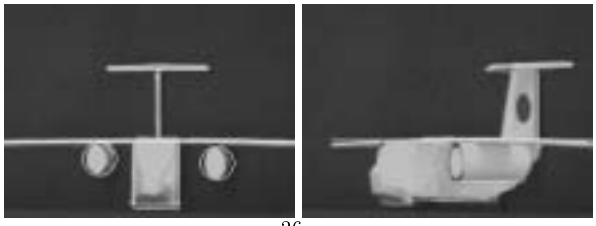
$\langle 332 \rangle \xrightarrow{26} \langle 420 \rangle$

(a)



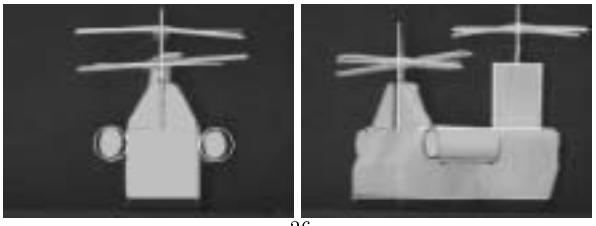
$\langle 342(332) \rangle \xrightarrow{26} \langle 410 \rangle$

(b)



$\langle 332 \rangle \xrightarrow{26} \langle 410 \rangle$

(c)



$\langle 332 \rangle \xrightarrow{26} \langle 540 \rangle$

(d)

$\langle 332 \rangle \xrightarrow{26} \langle 510 \rangle \xrightarrow{12} \langle 510 \rangle$

(e)

Figure 4.14: Experiments with the initial class as $\langle 332 \rangle$. (The figure in parentheses shows an example of recovery from feature detection errors). In each of these cases, the results for planning with primary moves alone, and those for both primary and auxiliary moves are identical

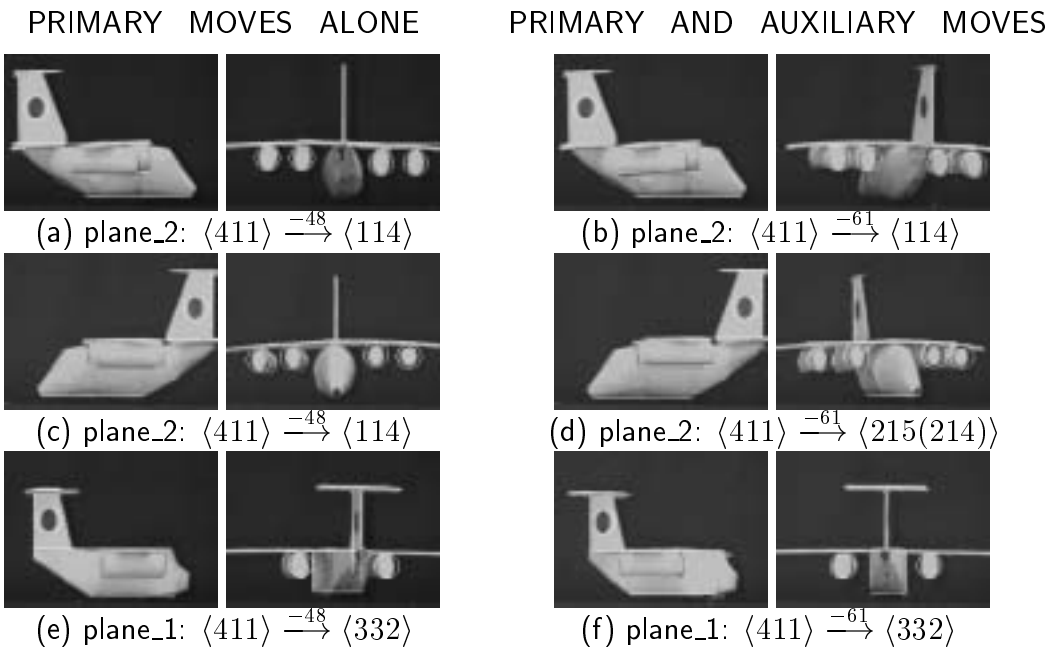


Figure 4.15: Experiments with the initial class as $\langle 411\rangle$. (The figure in parentheses shows an example of recovery from feature detection errors).

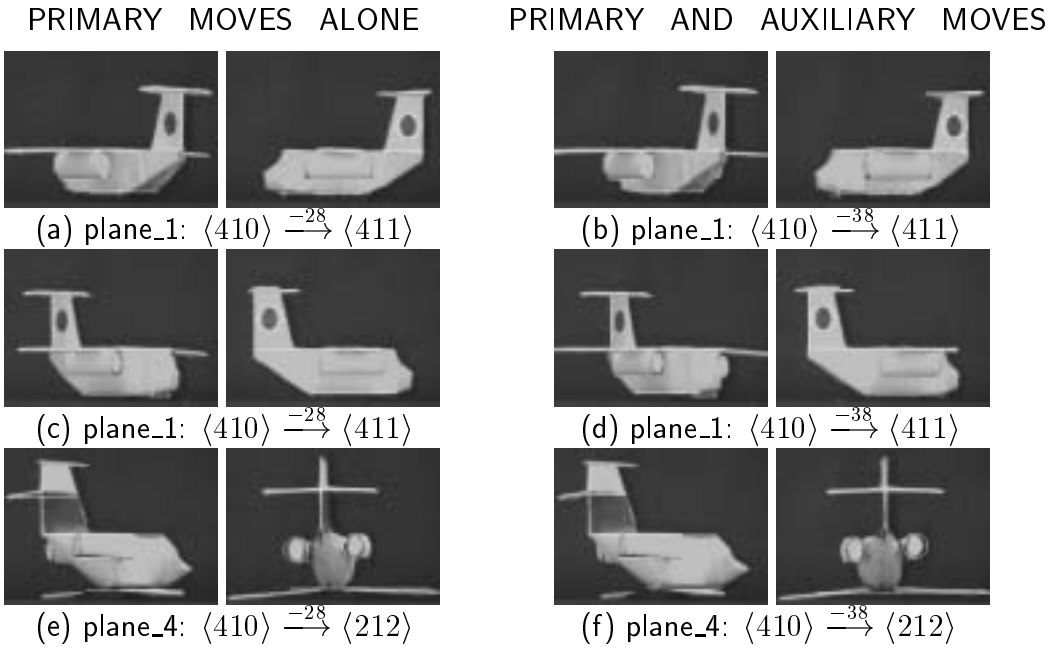


Figure 4.16: Experiments with the initial class as $\langle 410\rangle$

Ordering of Feature Detectors

The second images in Figures 4.15(a), (b) and (c) show cases where the system does not need to use the second feature detector. In each case, the line detector reports the feature-class present to be $\langle 11 \rangle$. For the objects in this model base, this could correspond to classes $\langle 112 \rangle$ and $\langle 114 \rangle$. Due to the movements made, the probability of class $\langle 114 \rangle$ becomes 0.95 and exceeds the class probability threshold. No further feature detection is required.

Recovery from Feature Detection Errors

In the first image in Figure 4.14(b), due to the shadow of the wing on the fuselage of the aircraft, the feature detector detects 4 vertical lines instead of 3, the correct number. Our recovery mechanism (Section 4.3.1) corrects this error. The second image in Figure 4.15 shows another example of an error recovery. The number of circles is detected to be 5. The $p_{circles\ 5\ 4}$ value (Section 4.2) is 1.0, hence the system corrects the error.

Some Sample Search Tree Details

For the experiments shown in Figure 4.15, the number of search tree nodes constructed for primary moves alone is 14, whereas the corresponding number for both primary and auxiliary moves is 125. The corresponding numbers for the experiments in Figure 4.16 are 14 and 41, respectively.

Average Number of Observations for a Given Number of Competing Aspects

For the results concerning the second model base, the average is computed over 58 experiments. The lower part of Table 4.1 shows the average number of observations for a given number of competing aspects.

4.4.3 Experiments Comparing Recognition Performance on Raw Aspect Data and AAGs Constructed as in Chapter 3

In this section, we compare the performance of our recognition algorithm on two different types of input data – raw aspect data, and the AAG constructed using the strategy of Chapter 3. Table 4.2 summarizes the results of 83 experiments with the polyhedral objects, and 96 experiments with the aircraft models. For the recognition experiments of Sections 4.4.1 and 4.4.2, the focus was on experiments with the initial view having a large degree of ambiguity associated with it – one which corresponds to a large number of aspects. Here, we take random views around the given object as starting points, and use our recognition algorithm.

The raw aspect data considers clusters of adjacent points in having the same feature data, as an aspect. Since we use noisy feature detectors for obtaining the raw aspect data, it contains instances of feature detection errors. Given an instance of raw aspect data, our AAG construction algorithm aims at reconstructing the corresponding error-free AAG as far as possible. A small region whose feature data is different from its neighbouring regions is more likely to be an error, as against an aspect of an error-free AAG (Section 3.4). Further, the positions in the AAG where such errors occur, are also not fixed. When we use a particular set of raw aspect data for a recognition experiment, the observed feature data at a particular position may not correspond to what it was in the raw aspect data. The recognition algorithm works on the basis of the aspect data – in this case, it is the raw aspect data itself. Any such case of the above discrepancy between the expected feature value and the observed one, amounts to a feature detection error. Hence, the recognition algorithm is likely to fail in a larger number of cases with the raw aspect data, as compared to experiments with the output of our AAG construction algorithm.

Our AAG construction algorithm stores estimates of feature detection errors as *ASSOC_TABLE* estimates (Section 3.5.3). We use these *ASSOC_TABLE* estimates

Model Base II: Polyhedral Objects		
	Raw Aspect Data	*AAG of Chapter 3*
Cases of feature detection errors	24	26
Recovery from feature detection errors	–	6
Failure due to feature detection errors	24	20
Av. no. of Image Processing Operations	2.322	2.242
Av. no. of search tree nodes	450.373	136.468
Model Base I: Aircraft Models		
	Raw Aspect Data	*AAG of Chapter 3*
Cases of feature detection errors	77	40
Recovery from feature detection errors	–	16
Failure due to feature detection errors	77	24
Av. no. of Image Processing Operations	2.368	1.784
Av. no. of search tree nodes	264.684	24.431

Table 4.2: A comparison of the object recognition algorithm on raw aspect data, and the output of the AAG construction algorithm of Chapter 3

to compute p_{jlk} values for each feature detector (Section 4.2). The use of p_{jlk} values makes the recognition system robust to many feature detection errors (Section 4.3.1). The recognition system using the raw aspect data cannot recover from any feature detection error.

A consequence of using raw aspect data in the recognition algorithm is the presence of very large number of aspects. Many of these correspond to feature detection errors, and are very small in size. Hence, the degree of ambiguity associated with a search tree node is expected to be quite high. Thus, the average number of search tree nodes is also much higher than the case of using an AAG constructed using the algorithm of Chapter 3. The search process of Section 4.3.2 may be subject to memory and processing constraints. Hence at any stage, one expects the aspect resolution process to be less complete than the latter situation. In such a case, the number of image processing operations is expected to be far higher than the system using output of our AAG construction algorithm.

The upper part of Table 4.2 shows the comparison for the polyhedral objects. The raw aspect data for the the polyhedral objects model base has a lower value of the Demerit Coefficient $\eta_{model\ base}$ and $\mathcal{S}(I)$ as compared to the aircraft model base (Section 3.7). Thus the raw aspect data is closer to the output of the AAG construction algorithm for the polyhedral objects model base, as compared with the aircraft models.

For the 83 experiments with the polyhedral objects, the number of cases of feature detection errors is nearly comparable (24 and 26, respectively for the raw aspect data, and the AAG constructed using the algorithm of Chapter 3). However, there are 6 cases of recovery from feature detection errors in the latter, which is not so for the raw aspect data. The average number of image processing operations is more for the raw aspect data (2.322), as compared with 2.242 for the latter. The average number of search tree nodes for the raw aspect data is about 3 times the corresponding value for the output of our AAG construction algorithm.

The aircraft model base clearly highlights the difference between the results with

raw aspect data, and the output of our AAG construction algorithm. This model base has a high value of Demerit Coefficient $\eta_{model\ base}$ and $\mathcal{S}(I)$ (Section 3.7). The number of feature detection errors is significantly higher for the raw aspect data – 77, as compared with 40 for the latter. Further, there are 16 instances of recovery from feature detection errors with the latter, using the $ASSOC\_TABLE$ estimates to advantage. The average number of image processing operations for the two types of inputs are 2.368 and 1.784, respectively. The average number of search tree nodes is significantly higher for the raw aspect data – nearly 11 times the corresponding value for the output of our AAG construction algorithm.

4.5 Conclusions

This chapter presents an integrated approach for the recognition of an isolated 3D object through on-line next view planning using probabilistic reasoning. We assume a single rotational degree of freedom between the object and an orthographic camera. We summarize the main features of our recognition system as follows:

- Our strategy is independent of any specific feature set.
- We account for feature detection errors not only in the AAG construction process (Chapter 3), but also in the object recognition stage, which uses the same noisy feature detectors.
- The probabilistic hypothesis generation mechanism can also handles cases of feature detection errors.
- Our hierarchical knowledge representation scheme enables fast and efficient hypothesis generation. It also facilitates planning by exploiting the relationships between features, aspects and object models.
- The planning process is reactive. It utilizes information from the current observation, as well as the past history, to plan a sequence of moves to disambiguate

between similar objects.

The recognition scheme has the ability to correctly identify objects even when they have a large number of similar views. If a feature set is not rich enough to identify an object from a single view, this strategy may be used to identify it from multiple views. We demonstrate that the proposed recognition strategy works correctly even under processing and memory constraints due to the incremental reactive planning strategy. No related work has addressed this problem.

While we use simple features for the purpose of illustration, one may use other features such as texture, colour, specularities and reflectance ratios. Over 100 experiments demonstrate the effectiveness of using simple features and multiple views even on a relatively complex class of objects with a high degree of ambiguity associated with a view of the object. Our experiments show that one may use simple features to recognize objects with complex 3-D shapes (as in Figure 3.14).

We have described various stages of our work on aspect graph-based object recognition in [65], [63], [62]. We have also reported a part of this work in [68].

Chapter 5

Inner Camera Invariants: A Tool for Next View Planning

In a multi-view 3-D object recognition system, pose information corresponding to a view is necessary to generate different hypotheses corresponding to the information extracted from the image of that view. Most pose estimation methods assume camera internal parameters to be known, or at least, fixed. In this chapter, we address the cases where the camera internals may be changed either accidentally, or on purpose. We use the basic projection model of a pin-hole camera to derive new constraints which are invariant to the internal parameters of the camera. For our formulation, we consider the most general 6-DOF case between the object and the uncalibrated camera. We show the application of these Inner Camera Invariants for pose estimation.

5.1 Planning with an Uncalibrated Camera

The object recognition scheme of Chapter 4 considers the problem of recognizing a 3-D object given that the object always fits in the camera's field of view. The system assumes an orthographic camera, and a 1-DOF case – a single degree of freedom

(rotational) between the object and the camera. We now proceed to relax these requirements. First, we remove the requirement for the entire object to fit in the camera's field of view. What may be visible to the camera at a point in time is only a portion of the complete object. Now, we consider a projective camera. Further, we consider the 6-DOF (3 degrees for rotation, 3 for translation) case between the object and the camera.

A multi-view 3-D object recognition system needs pose information for a given view, to generate different hypotheses corresponding to the information extracted from the view. The next view planning module uses this information to propose a move from the current position to disambiguate between the competing hypotheses. Most approaches to pose determination assume internal parameters of a camera to be fixed and known. We address the cases under which the internal parameters of the camera can be changed, either accidentally or voluntarily. (Other related papers that deal with the case of varying internal camera parameters include those of Polleyfeys *et al.* [161] and Crowley, Bobet and Schmid [54], [53]) Some examples of intentional variation of internal parameters are the use of an auto-focus and auto-aperture camera, using lenses of different focal lengths, and the use of lens range extenders. We use the basic projection model of a pin-hole camera [78] to derive new constraints which are invariant to the internal parameters of the camera. We show that these new constraints can be used for pose estimation – without going through the often cumbersome step of camera calibration.

Conventional camera calibration methods assume that the correspondences between the world points and the image points are given via the imaging of a regularly patterned calibration object. They estimate external and internal parameters of the camera either linearly or non-linearly, by considering a sufficient number of points (*e.g.*, [197], [183], [77], [78]). Many recent methods of carrying out Euclidean measurements using computer vision have attempted to get rid of the regularly patterned calibration object required for obtaining the camera internals using camera self-calibration (*e.g.*, [142], [52], [98], [4], [161], [101]). The constraint obtained by

elimination is independent of what has been eliminated. Therefore if we eliminate the internal parameters of the camera from the basic constraint, the resulting constraint is independent of the camera internals. In such a case the change of the internal parameters of the camera does not affect the constraint. This is the key point of the proposed method in this chapter. Most approaches attempt to explicitly find out the internals of the camera, and use them for various vision tasks. *In this chapter, we attempt to investigate the process of working with functions which are invariant to the internal parameters of a camera, and using them for various applications.*

5.2 Inner Camera Invariants

The classical pin-hole camera model is often assumed when the camera is used as a measuring device in vision tasks. The following equation describes the imaging process [78]:

$$\lambda \mathbf{m} = \mathbf{P}\mathbf{M} = \mathbf{A}\,[\mathbf{R} \mid \mathbf{t}]\,\mathbf{M} \tag{5.1}$$

Here, $\mathbf{M} = (X, Y, Z, W)^T$ is a 3-D world point, and $\mathbf{m} = (x, y, 1)^T$ is the corresponding image point. $\mathbf{R}$ (3×3) and $\mathbf{t}$ (3×1) are the rotation and translation aligning the world coordinate system with the camera coordinate system (the *External Parameters*), and $\mathbf{A}$ is the matrix of the *Internal Parameters* of the camera. $\mathbf{A}$ may be written as [78]:

$$\mathbf{A} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{5.2}$$

where f_x and f_y are the effective focal lengths in the x and y directions and (u_0, v_0) is the principal point. (A more general model assumes the term A_{12} in $\mathbf{A}$ to represent a *camera skew* term. However, such a term may often be considered negligible [78],

[161].) $[\mathbf{R} \mid \mathbf{t}]$ is the matrix of camera externals given by

$$[\mathbf{R} \mid \mathbf{t}] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} = \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix} \tag{5.3}$$

Here, r_{ij}, $i, j \in \{1, 2, 3\}$ are functions of R_x, R_y and R_z – the rotations about the **X**-, **Y**- and **Z**- axes, respectively. The external parameters of a camera have six degrees of freedom, while the internal parameters have four [78].

Suppose we know three 3-D points, $\mathbf{M}_p = (X_p, Y_p, Z_p, 1)^T$, $p \in \{i, j, k\}$, and their images on the image plane, $\mathbf{m}_p = (u_p, v_p, 1)^T$, $p \in \{i, j, k\}$. By eliminating the internals of the camera, we obtain

$$\begin{cases} J_{ijk} = \frac{u_i - u_j}{u_i - u_k} = \frac{\frac{\mathbf{r}_1\mathbf{M}_i}{\mathbf{r}_3\mathbf{M}_i} - \frac{\mathbf{r}_1\mathbf{M}_j}{\mathbf{r}_3\mathbf{M}_j}}{\frac{\mathbf{r}_1\mathbf{M}_i}{\mathbf{r}_3\mathbf{M}_i} - \frac{\mathbf{r}_1\mathbf{M}_k}{\mathbf{r}_3\mathbf{M}_k}} \\ K_{ijk} = \frac{v_i - v_j}{v_i - v_k} = \frac{\frac{\mathbf{r}_2\mathbf{M}_i}{\mathbf{r}_3\mathbf{M}_i} - \frac{\mathbf{r}_2\mathbf{M}_j}{\mathbf{r}_3\mathbf{M}_j}}{\frac{\mathbf{r}_2\mathbf{M}_i}{\mathbf{r}_3\mathbf{M}_i} - \frac{\mathbf{r}_2\mathbf{M}_k}{\mathbf{r}_3\mathbf{M}_k}} \end{cases}, \tag{5.4}$$

in which J_{ijk} and K_{ijk} are image measurements that are functions of $\mathbf{R}$, $\mathbf{t}$ and $\mathbf{M}_p$, $p \in \{i, j, k\}$, and are independent of the internals of the camera. Thus the above equations can be re-written as the following constraints:

$$\begin{cases} J_{ijk} = f_{ijk}(\mathbf{R}, \mathbf{t}, \mathbf{M}_i, \mathbf{M}_j, \mathbf{M}_k) \\ K_{ijk} = g_{ijk}(\mathbf{R}, \mathbf{t}, \mathbf{M}_i, \mathbf{M}_j, \mathbf{M}_k) \end{cases} \tag{5.5}$$

J_{ijk} and K_{ijk} are image measurements that are independent of the internals of the camera. The left hand sides of **constraint** Equation 5.5 represent image measurements based on three points which are invariant to the camera internals. The right hand sides are non-linear trigonometric expressions which are functions of only the camera externals and the structure (Euclidean coordinates) of the three points. We refer to J_{ijk} and K_{ijk} as **Inner Camera Invariants**. These parameters are indeed the invariants of the homography **A** which represents the change of the projective coordinate systems between the camera and image.

5.3 Pose Estimation using Inner Camera Invariants

In this section we address the problem of pose estimation from known landmarks using the invariants described above. Given $n \geq 3$ control points in the 3-D world, we can get $2(n-2)$ *independent* constraints from one view of the 3-D scene. Suppose the number of views is N, then the total number of the *independent* constraints is $2N(n-2)$. We show the use of inner camera invariants to estimate the pose of an object, or a part of the object, whose Euclidean coordinates are known.

5.3.1 Pose estimation using Euclidean landmarks: general case

Suppose that we know the Euclidean coordinates $(X_i, Y_i, Z_i, 1)^T$ of 5 points in the world coordinate system. Six independent invariant measurements give us six equations in terms of the six unknowns in $(\mathbf{R}, \mathbf{t})$. The six equations can be solved numerically (using nonlinear constrained optimization routines for systems of nonlinear equations, for example) for a complete pose estimation using an uncalibrated camera and known landmarks.

In the case of constrained planar motion, $\mathbf{R}$ has only one degree of freedom and $\mathbf{t}$ has two degrees of freedom. The total number of unknowns in such a case is three. Then four control points are sufficient for pose estimation.

For a 4-DOF case (*e.g.*, as in Figure 1.5: the camera can move along the $\mathbf{Y}$-axis in addition to the constrained planar motion setup mentioned above), four control points result in four equations in four unknowns. Hence, we can perform pose estimation using four control points.

It turns out that in some special cases it is possible to obtain closed-form or linear solutions to the pose estimation problem using the image invariants.

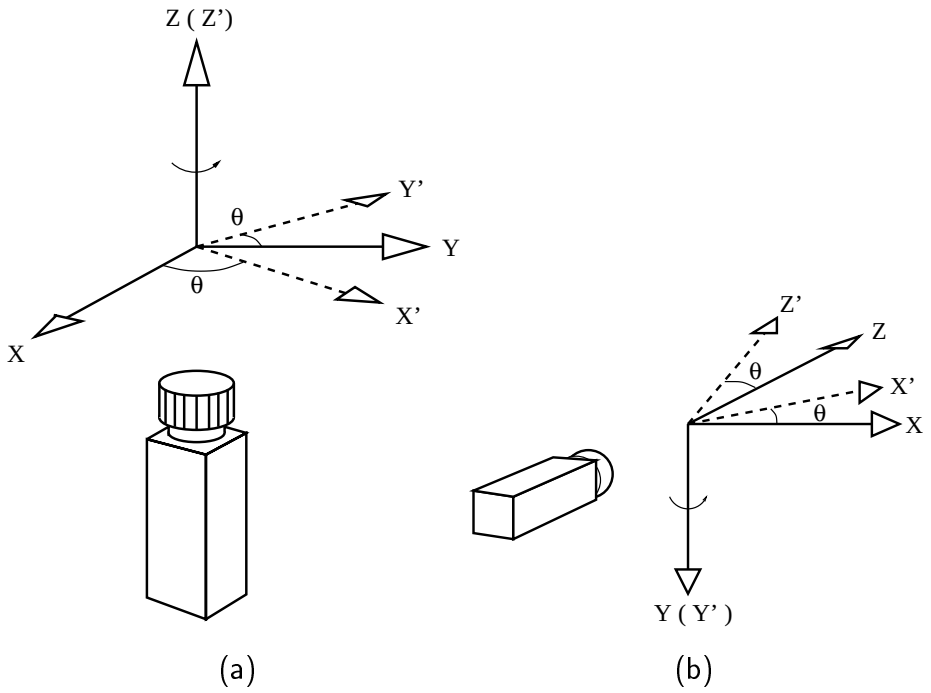


Figure 5.1: Illustrations of special cases: (a) constrained camera rotation only about the **Z**-axis, and (b) constrained camera rotation only about the **Y**-axis.

5.3.2 Special Case: Rotation only about Z-axis

Let $\mathbf{X}_w\mathbf{Y}_w\mathbf{Z}_w$ be the world coordinate system. Consider restricted motion with rotation θ only about the $\mathbf{Z}_w$-axis, shown in Figure 5.1(a). This would be the case of a robot moving on the ground with a special camera looking vertically up at the landmarks on the ceiling to facilitate localization.

Suppose there are three 3-D control points $\mathbf{M}_p$, $p \in \{i, j, k\}$ lying on the $\mathbf{X}_w\mathbf{Y}_w$-plane (ceiling) of the 3-D world coordinate system, where $\mathbf{M}_p = (X_p, Y_p, 0, 1)^T$, and the corresponding image coordinates are $\mathbf{m}_p = (u_p, v_p, 1)^T$. The image projection is

$$\begin{aligned} u_p &= f_x \frac{X_p cos\theta - Y_p sin\theta + t_x}{t_z} + u_0 \\ v_p &= f_y \frac{X_p sin\theta + Y_p cos\theta + t_y}{t_z} + v_0 \end{aligned} \quad , p \in \{i, j, k\}. \tag{5.6}$$

By eliminating the internals of the camera from Equation 5.6, we obtain

$$\begin{array}{l} \frac{u_i-u_j}{u_i-u_k} = \frac{(X_i-X_j)cos\theta-(Y_i-Y_j)sin\theta}{(X_i-X_k)cos\theta-(Y_i-Y_k)sin\theta} \\ \frac{v_i-v_j}{v_i-v_k} = \frac{(X_i-X_j)sin\theta+(Y_i-Y_j)cos\theta}{(X_i-X_k)sin\theta+(Y_i-Y_k)cos\theta} \end{array} . \tag{5.7}$$

Let $J_{ijk} = (u_i - u_j)/(u_i - u_k)$ and $K_{ijk} = (v_i - v_j)/(v_i - v_k)$. Let $\mathcal{P}_{pq}$ stand for $\mathcal{P}_p - \mathcal{P}_q$. We obtain

$$tan\theta = \frac{X_{ij} - J_{ijk}X_{ik}}{Y_{ij} - J_{ijk}Y_{ik}} = \frac{Y_{ij} - K_{ijk}Y_{ik}}{K_{ijk}X_{ik} - X_{ij}}. \tag{5.8}$$

We can easily compute θ (the rotation about the **Z**-axis) from Equation 5.8. Once θ is obtained, the translation vector $\mathbf{t}$ can be computed as follows. Suppose we get another 3-D control point $\mathbf{M}_l = (X_l, Y_l, Z_l, 1)^T$ which is not on the $\mathbf{X}_w\mathbf{Y}_w$-plane (let us not stop at just painting the ceiling, but hang a few sticks as well). We compute J_{ijl} and K_{ijl} in the same way as in Equation 5.7. Then we have

$$\begin{array}{l} \frac{t_z}{Z_l} = \frac{J_{ijl}(a_i+t_x)-(a_i-a_j)}{(a_i-a_j)-J_{ijl}(a_i-a_l)} \\ \frac{t_z}{Z_l} = \frac{K_{ijl}(b_i+t_y)-(b_i-b_j)}{(b_i-b_j)-K_{ijl}(b_i-b_l)} \end{array} \tag{5.9}$$

where

$a_p = X_pcos\theta - Y_psin\theta$, and

$b_p = X_psin\theta + Y_pcos\theta$, for $p \in \{i, j, l\}$

By equating the two equations in Equation5.9, and rearranging the terms,

$$A_{ijl}\ t_x + B_{ijl}\ t_y = C_{ijl} - D_{ijl} \tag{5.10}$$

where

$A_{ijl} = \frac{J_{ijl}}{(a_i-a_j)-J_{ijl}(a_i-a_l)}$;

$B_{ijl} = \frac{-K_{ijl}}{(b_i-b_j)-K_{ijl}(b_i-b_l)}$;

$C_{ijl} = \frac{b_iK_{ijl}-(b_i-b_j)}{(b_i-b_j)-K_{ijl}(b_i-b_l)}$;

$D_{ijl} = \frac{a_iJ_{ijl}-(a_i-a_j)}{(a_i-a_j)-J_{ijl}(a_i-a_l)}$.

If we know more 3-D control points which are not on the $\mathbf{X}_w\mathbf{Y}_w$-plane, we can get more equations in the form of Equation 5.10. In such a case, a linear least squares

technique can be used to solve for t_x and t_y. These values can be substituted in Equations 5.10 to compute t_z:

$$t_z = Z_l\ (D_{ijl} + A_{ijl}\ t_x) = Z_l\ (C_{ijl} - B_{ijl}\ t_y) \tag{5.11}$$

Thus, in this special case, we obtain a closed-form solution to the localization problem, given three Euclidean landmarks on the $\mathbf{X}_w\mathbf{Y}_w$-plane and at least two landmarks off the $\mathbf{X}_w\mathbf{Y}_w$-plane. If we have more control points, we have a linear method to do the same, as shown above.

5.3.3 Special Case: Planar motion and rotation about Y-axis

Let $\mathbf{X}_w\mathbf{Y}_w\mathbf{Z}_w$ be the world coordinate system. Consider a restricted motion with rotation θ only about the $\mathbf{Y}_w$-axis, shown in Figure 5.1(b). This would be the case of a robot moving on the ground with a camera mounted horizontally (we look normally, at last). Consider three points in the world coordinate system $\mathbf{M}_p$, $p \in \{1, j, k\}$, where $\mathbf{M}_p = (X_p, Y_p, Z_p, 1)^T$ and the corresponding image coordinates are $\mathbf{m}_p = (u_p, v_p, 1)^T$.

By projecting the 3-D control points on to the image plane, we have

$$\begin{aligned} u_p &= f_x \frac{X_p cos\theta + Z_p sin\theta + t_x}{-X_p sin\theta + Z_p cos\theta + t_z} + u_0 \\ v_p &= f_y \frac{Y_p + t_y}{-X_p sin\theta + Z_p cos\theta + t_z} + v_0 \end{aligned} \quad , p \in \{i, j, k\}. \tag{5.12}$$

Now, let the first two points i and j be on the $\mathbf{Y}_w$-axis and the third point k be on the $\mathbf{X}_w\mathbf{Y}_w$-plane. By eliminating the internal parameters of the camera from the above equation we obtain

$$\begin{aligned} J_{ijk} &= \frac{u_i - u_j}{u_i - u_k} = 0 \\ K_{ijk} &= \frac{v_i - v_j}{v_i - v_k} = \frac{Y_i - Y_j}{(Y_i + t_y) - \frac{Y_k + t_y}{1 - X_k \frac{sin\theta}{t_z}}} \end{aligned} \quad . \tag{5.13}$$

If we consider another point l on the $\mathbf{X}_w\mathbf{Y}_w$-plane, we obtain, similarly,

$$\begin{aligned} J_{ijl} &= \frac{u_i - u_j}{u_i - u_l} = 0 \\ K_{ijl} &= \frac{v_i - v_j}{v_i - v_l} = \frac{Y_i - Y_j}{(Y_i + t_y) - \frac{Y_l + t_y}{1 - X_l \frac{sin\theta}{t_z}}} \end{aligned} \quad . \tag{5.14}$$

Because J_{ijk} and J_{ijl} vanish, we can separate the terms of $sin\theta$ and t_z to obtain

$$\begin{aligned}\frac{t_z}{sin\theta} &= \frac{X_k((1-K_{ijk})Y_i - Y_j - K_{ijk}t_y)}{Y_i - Y_j + K_{ijk}(Y_k - Y_i)} \\ &= \frac{X_l((1-K_{ijl})Y_i - Y_j - K_{ijl}t_y)}{Y_i - Y_j + K_{ijl}(Y_l - Y_i)} = T_s. \end{aligned} \tag{5.15}$$

Therefore t_y can be easily found from Equation 5.15. In order to find θ, we have to know another control point lying on the $\mathbf{Y}_w\mathbf{Z}_w$-plane, say m. Following the derivation process of Equation 5.15, we can obtain a similar relationship as follows

$$\frac{t_z}{cos\theta} = -\frac{Z_m((1-K_{ijm})Y_i - Y_j - K_{ijm}t_y)}{Y_i - Y_j + K_{ijm}(Y_m - Y_i)} = T_c. \tag{5.16}$$

Since we know t_y, we can compute T_s and T_c. Thus, we can compute θ and t_z using the relations $tan\theta = T_c/T_s$ and $t_z = T_s sin\theta = T_c cos\theta$.

In Equation 5.12, J_{ijk} vanishes because both 3-D points i and j are on the $\mathbf{Y}_w$-axis. J_{ijk} would not be zero if we choose 3 points in general position. Let us select a point $\mathbf{M}_n = (X_n, Y_n, Z_n)^T$ in general position. Now that we know θ, t_y and t_z, solving for t_x is trivial. We use the two points $\mathbf{M}_l$ and $\mathbf{M}_k$ on the $\mathbf{X}_w\mathbf{Y}_w$-plane we had selected above.

$$t_x = \frac{b_n\ N_{kl} - J_{kln}\ a_l\ D_{kn}}{J_{kln}\ a_l\ (b_n - a_k) - b_n(a_l - a_k)}, \tag{5.17}$$

where

$a_p = -X_p sin\theta + t_z,\ p \in \{k, l\}$;

$b_n = -X_n sin\theta + Z_n cos\theta + t_z$;

$N_{kl} = (a_l X_k - a_k X_l) cos\theta$;

$D_{nk} = (b_n X_k - a_k X_n) cos\theta - a_k Z_n sin\theta$.

Thus, in this special case, we obtain a closed form solution to the localization problems with specially located landmarks - two on the $\mathbf{Y}_w$-axis, two on the $\mathbf{X}_w\mathbf{Y}_w$-plane, one on the $\mathbf{Z}_w\mathbf{Y}_w$-plane, and one in general configuration.

5.4 3-D Euclidean reconstruction from known ego-motions

Suppose we have a robot whose ego-motion and odometry can be known exactly from positional encoders and we know the $\mathbf{R}$ and $\mathbf{t}$ at N locations of the camera. Such information can be obtained for example, by solving the pose estimation problem for one station from known landmarks and then rigidly moving the camera with accurate known odometry to obtain arbitrary other landmarks within the field of view. As before, we assume that the internals of the camera are unknown, and may vary. For Euclidean structure estimation of n 3-D control (landmark) points we have $3n$ unknowns. With N views we have $2N(n-2)$ independent constraint equations and the relationship $2N(n-2) \geq 3n$ must be satisfied. Thus with 3 views ($N = 3$) of at least 4 points ($n \geq 4$) we can compute the 3-D Euclidean structure provided the pose of each of the camera stations are known.

5.5 Experimental Results

We have carried out several experiments on real images, and compared the results with that of standard calibration. We present results of experiments with a special calibration object (Figure 5.2), as well as a model whose points of interest have known 3-D coordinates (Figure 5.3). For calibration, we consider the object at two positions, a fixed distance apart ($200mm$). Data from two such positions constitutes one data set. The world coordinates corresponding to both the positions of the calibration object are known and they serve as Euclidean landmarks. We refer to the world points (corners) on our calibration objects with numbers assigned according to a row major order. We obtained the calibration data using Tsai's method [197].

Across our experiments, the calibration and 3-D reconstruction results from classical camera calibration are also listed for comparison, all distances are in mm and all angles are in degrees. For all our experiments we used the non constrained-linear op-

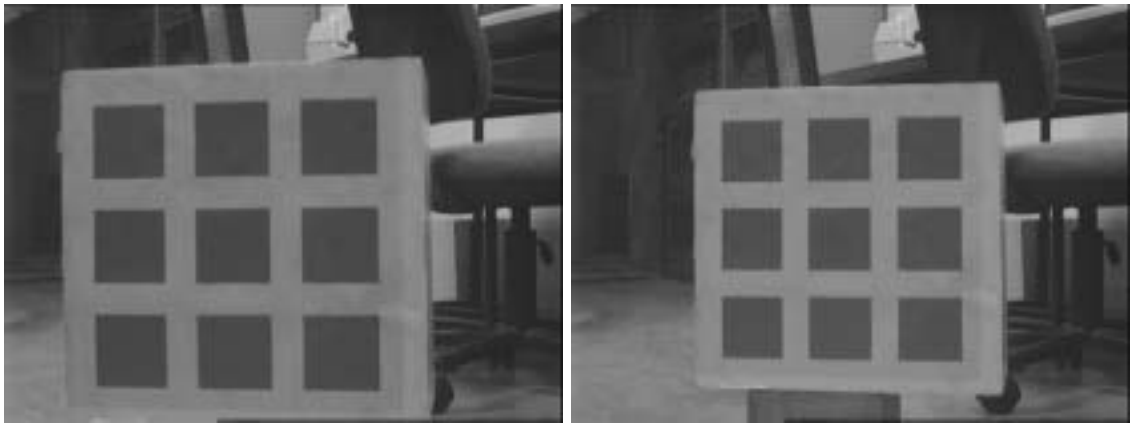


Figure 5.2: Images of the Calibration object at two positions $200mm$ apart

timization routine of MATLAB(`constr/fmincon`). We first set up the inner camera invariant equations (as in Equation 5.5, for example), and consider each optimization equation as the absolute difference of the left and right hand sides of the inner camera invariant equation. These are optimized with respect to suitable constraints *e.g.*, the allowable error, solution search neighborhoods, etc.

5.5.1 General Pose Estimation

We present some results with the calibration object (Figure 5.2), as well as the model house (Figure 5.3). In Table 5.1 we present some sample results for two camera stations of calibration grid using 5 points. In Table 5.2, we show some sample results using 20 points for two camera stations in the house sequence.

5.5.2 Pose Estimation: Rotation only about the Z-axis

For the first special case (Section 5.3.2), we used two camera stations, with two sets of 5 points each for each camera station. Figure 5.4 shows images from three such camera stations (constrained rotation about the **Z**-axis only: We kept our camera horizontal and rotated our calibration object). Table 5.3 summarizes the results.

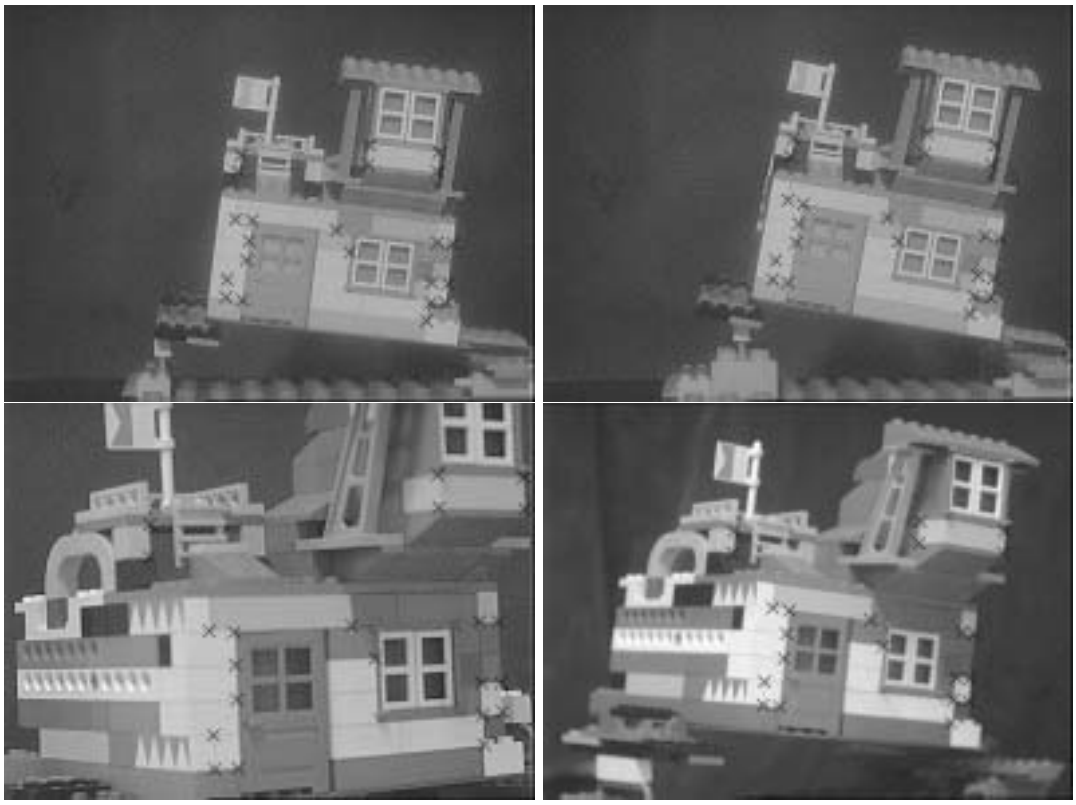


Figure 5.3: Images of a model house used for pose estimation (general case) and structure estimation (general case). Image points used in our experiments are marked with crosses.

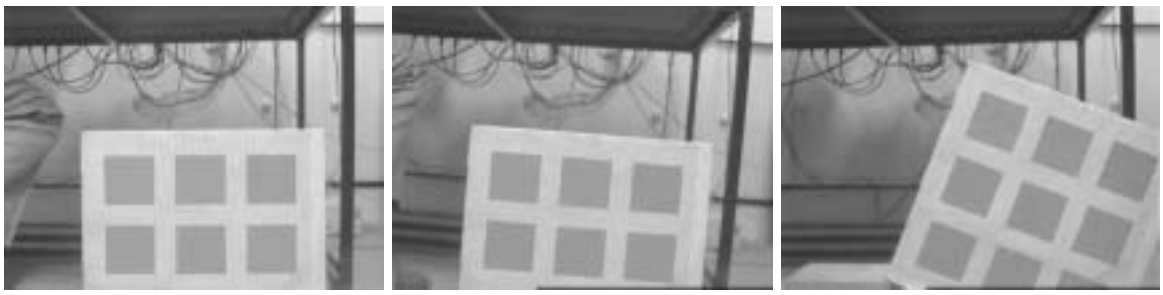


Figure 5.4: Images taken at three camera stations, which we have used for pose estimation: special case 1 (constrained camera rotation about the **Z**-axis only: Section 5.3.2)

	Pose 1		**Pose 2**	
	Calib.	*Estim.*	*Calib.*	*Estim.*
$R_x(degrees)$	0.376	1.906	0.003	-0.252
$R_y(degrees)$	-4.795	-6.518	-28.727	-28.449
$R_z(degrees)$	0.649	0.509	0.316	0.454
$t_x(mm)$	-694.802	-728.196	-434.520	-434.542
$t_y(mm)$	24.189	17.798	18.000	17.971
$t_z(mm)$	1041.215	1087.956	822.988	822.989

Table 5.1: Pose estimation experiments with the calibration object: Some sample results. We compare the poses ($\mathbf{R}$ and $\mathbf{t}$) computed by our method and standard calibration at two camera stations denoted by Pose 1 and Pose 2, respectively. All angles are in degrees, and all distances in mm.

	Pose 1		**Pose 2**	
	Calib.	*Estim.*	*Calib.*	*Estim.*
$R_x(degrees)$	-0.641	-0.052	0.311	-0.882
$R_y(degrees)$	-2.787	-2.830	-32.467	-27.754
$R_z(degrees)$	6.753	6.704	-1.797	-2.681
$t_x(mm)$	-519.353	-516.074	-461.840	-450.918
$t_y(mm)$	-62.001	-63.691	487.470	-491.092
$t_z(mm)$	855.277	833.717	183.907	168.206

Table 5.2: Pose estimation experiments (general case) with the model house: Some sample results at two camera stations(Pose 1 and Pose 2), and comparison with calibration data. All angles are in degrees, and all distances in mm.

Camera Station I: Calibration Data: t_x = -634.112 mm, t_y = 16.029 mm, t_z = 1057.223 mm $R_x = 1.080^o$, $R_y = 0.837^o$, $R_z = -0.317^o$				
Points	$R_z(degrees)$	$t_x(mm)$	$t_y(mm)$	$t_z(mm)$
5,10,15,40,48	-0.578	-654.555	15.526	934.162
8,12,19,39,44	-0.524	-648.851	39.923	1063.521
Camera Station II: Calibration Data: t_x = -633.105 mm, t_y = -35.049 mm, t_z = 1053.015 mm $R_x = 0.831^o$, $R_y = 0.672^o$, $R_z = 4.701^o$				
Points	$R_z(degrees)$	$t_x(mm)$	$t_y(mm)$	$t_z(mm)$
6,13,18,25,37	4.975	-634.835	-29.289	990.971
4,15,20,27,48	2.047	-575.648	-37.7681	1062.035
Camera Station III: Calibration Data: t_x = -455.540 mm, t_y = -263.391 mm, t_z = 1054.937 mm $R_x = 1.146^o$, $R_y = 0.624^o$, $R_z = 21.009^o$				
Points	$R_z(degrees)$	$t_x(mm)$	$t_y(mm)$	$t_z(mm)$
9,16,20,37,43	21.311	-476.379	-245.627	1191.419
12,17,22,26,37	20.531	-450.724	-249.275	1771.772

Table 5.3: Pose estimation results, Special Case 1 (Section 5.3.2: the case of constrained camera rotation about the **Z**-axis alone). We give the estimated values of R_z and $\mathbf{t}$ for two camera stations for different choices of 5 points. We also indicate the results from standard calibration for comparison. All angles are in degrees, and all distances are in mm.

Figure 5.5: The sequence of images used for special case 2 of pose estimation (Section 5.3.3): constrained rotation about the **Y**-axis only

5.5.3 Pose Estimation: Rotation only about the Y-axis

We took two sets of six points each, for two camera stations (shown in Figure 5.5). In Table 5.4, we show results for special case 2 of pose estimation(Section 5.3.3).

5.5.4 3-D Euclidean reconstruction from known ego-motions

Figure 5.6 shows images taken from 4 viewpoints around our calibration object, which are used for experiments in this case. In Table 5.5, we show some results for recovering the 3-D structure of points from known ego-motions(Section 5.4). We consider N=3 and n=4.

In Table 5.6, we show some results for the model house images (Figure 5.3).

We considered two cases here, i) N=3 and n=4, and ii) N=4, n=5. Results for these are shown in the two parts of the table.

5.6 Conclusions

We summarize our results using inner camera invariants as follows:

1. General pose estimation: Knowing the Euclidean coordinates of at least 5 world points enables complete pose recovery

2. Constrained rotation about the **Z**-axis: We can recover the pose of the camera,

Camera Station I: Calibration Data: $t_x = 3.935\ mm$, $t_y = 20.416\ mm$, $t_z = 1063.303\ mm$ $R_x = 0.003^o$, $R_y = -28.727^o$, $R_z = 0.316^o$				
Points	$R_y(degrees)$	$t_x(mm)$	$t_y(mm)$	$t_z(mm)$
7,19,6,21,31,42	-25.002	50.245	18.330	929.101
7,13,22,24,37,44	-35.107	-108.729	185.001	2391.125
Camera Station II: Calibration Data: $t_x = 21.948\ mm$, t_y=31.988 mm, t_z=1034.709 mm $R_x = -0.931^o$, $R_y = -39.155^o$, $R_z = 1.314^o$				
Points	$R_y(degrees)$	$t_x(mm)$	$t_y(mm)$	$t_z(mm)$
7,19,6,21,31,42	-30.843	138.861	14.758	883.234
7,13,4,22,37,47	-33.573	33.137	20.163	957.999
Camera Station III: Calibration Data: t_x = -210.449 mm, t_y = 18.132 mm, t_z = 1195.758 mm $R_x = 0.345^o$, $R_y = 23.262^o$, $R_z = 0.504^o$				
Points	$R_y(degrees)$	$t_x(mm)$	$t_y(mm)$	$t_z(mm)$
7,19,8,22,37,44	28.266	-99.024	47.190	1420.395
13,19,15,20,43,47	27.275	-153.946	45.356	1311.451

Table 5.4: Pose estimation results, Special Case 2 (Section 5.3.3: the case of constrained camera rotation about the **Y**-axis alone). We give the estimated values of R_z and $\mathbf{t}$ for two camera stations for different choices of 5 points. We also indicate the results from standard calibration for comparison. All angles are in degrees, and all distances are in mm.

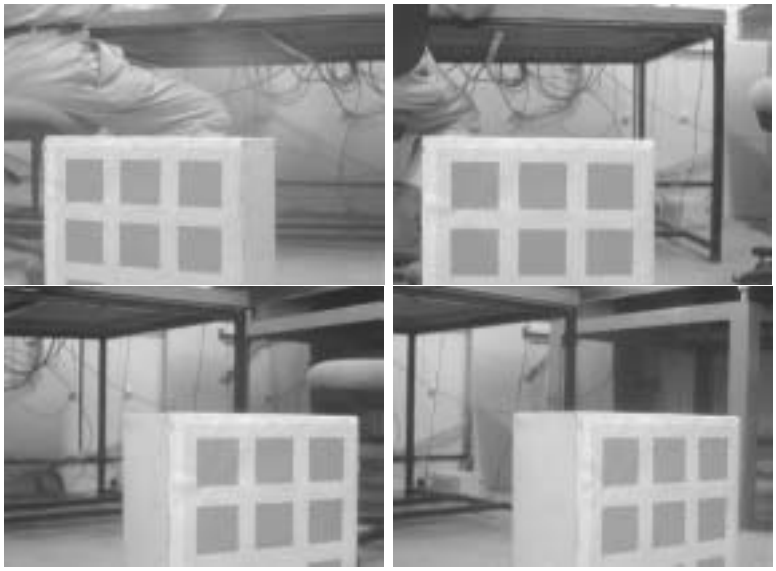


Figure 5.6: The 4 viewpoints used for the 3-D structure estimation experiments (Section 5.4) with the calibration object

Calibration Data			Reconstructed Data		
$X(mm)$	$Y(mm)$	$Z(mm)$	$X(mm)$	$Y(mm)$	$Z(mm)$
500	0	200	495.884	3.501	197.995
700	70	0	704.152	69.4679	-1.348
500	100	200	507.501	98.997	203.334
670	0	200	669.324	0.190	199.888

Table 5.5: Structure estimation from known ego-motions (Section 5.4): Some sample results for the calibration object. All coordinates (X, Y and Z) are in mm.

Calibration Data			Reconstructed Data		
Xmm	$Y(mm)$	$Z(mm)$	$X(mm)$	$Y(mm)$	$Z(mm)$
515.875	19.05	0.0	515.884	19.088	0.0532
515.875	28.575	0.0	515.871	28.584	-0.140
563.5	19.05	0.0	563.504	19.085	-0.005
563.5	-31.75	-31.75	563.489	-31.897	-31.662
515.875	19.05	0.0	515.971	11.889	3.267
515.875	28.575	0.0	515.591	20.120	-3.854
563.5	19.05	0.0	559.544	11.644	-3.282
563.5	-31.75	-31.75	560.507	-33.378	-28.832
507.938	-12.7	31.75	508.842	-17.897	34.472

Table 5.6: Structure estimation from known ego-motions (Section 5.4): Some sample results for the model house. All coordinates (X, Y and Z) are in mm.

knowing the Euclidean coordinates of 3 points on the **XY**-plane, and two or more points in general configuration

3. Constrained rotation about the **Y**-axis: We can recover the camera pose from the Euclidean coordinates of 2 points on the **Y**-axis, 2 points on the **XY**-plane, one on the **YZ**-plane, and one in general configuration.

4. General 3-D Euclidean Structure Estimation: For general 3-D structure estimation, the following relation between the number of views N and the number of points in each view n, should be satisfied: $2N(n-2) \geq 3n$.

While a solution in the first case involves non-linear optimization, we show that in the other two cases, it is possible to obtain closed-form linear solutions to the pose estimation problem using these image invariants. We have reported a part of this work in [204], [205]. In [205], we also show in detail how different methods of self-calibration are all based on elimination of different entities from the basic projection equations. We show that instead of explicitly estimating the possibly varying parameters of a camera, one may use inner camera invariants to advantage in various vision applications.

For our experiments in part-based object recognition (Chapter 6), we use the general pose estimation method (Section 5.3.1). We have not used the special cases of pose estimation (Sections 5.3.2 and 5.3.3) because they impose a slightly strict condition on the structure of the landmarks (the parts of the object). In our implementation of the part-based object recognition system, we did not need to use the structure estimation result (Section 5.4). However, this may be important in cases where we want to have an extra verification step: After having found out the pose of an object or a part, one may take a few more views around it. From these, and using correspondence information between the points of interest in the different views, one can calculate the 3-D Euclidean structure of the points of interest. This may be used to verify the identity of the object or the part.

Chapter 6

Part-based Recognition through Next View Planning using Inner Camera Invariants

A complete view of a 3-D object may not fit in the camera's field of view. Further, the available view of the object may not be sufficiently distinctive to identify it uniquely. In this chapter, we present a new on-line scheme for the recognition and pose estimation of such 3-D objects using reactive next view planning. We consider an uncalibrated, projective camera, and consider the case when the internal parameters of the camera may vary. For our formulation, we consider the most general 6-DOF case between the object and the camera.

6.1 Part-based Object Recognition

We need to modify the recognition and planning scheme of Chapter 4 when the complete view of a 3-D object is not inside the camera's field of view. The system of Chapter 4 assumes an orthographic camera, and a 1-DOF case - a single rotational degree of freedom between the object and the camera. We now do away with these

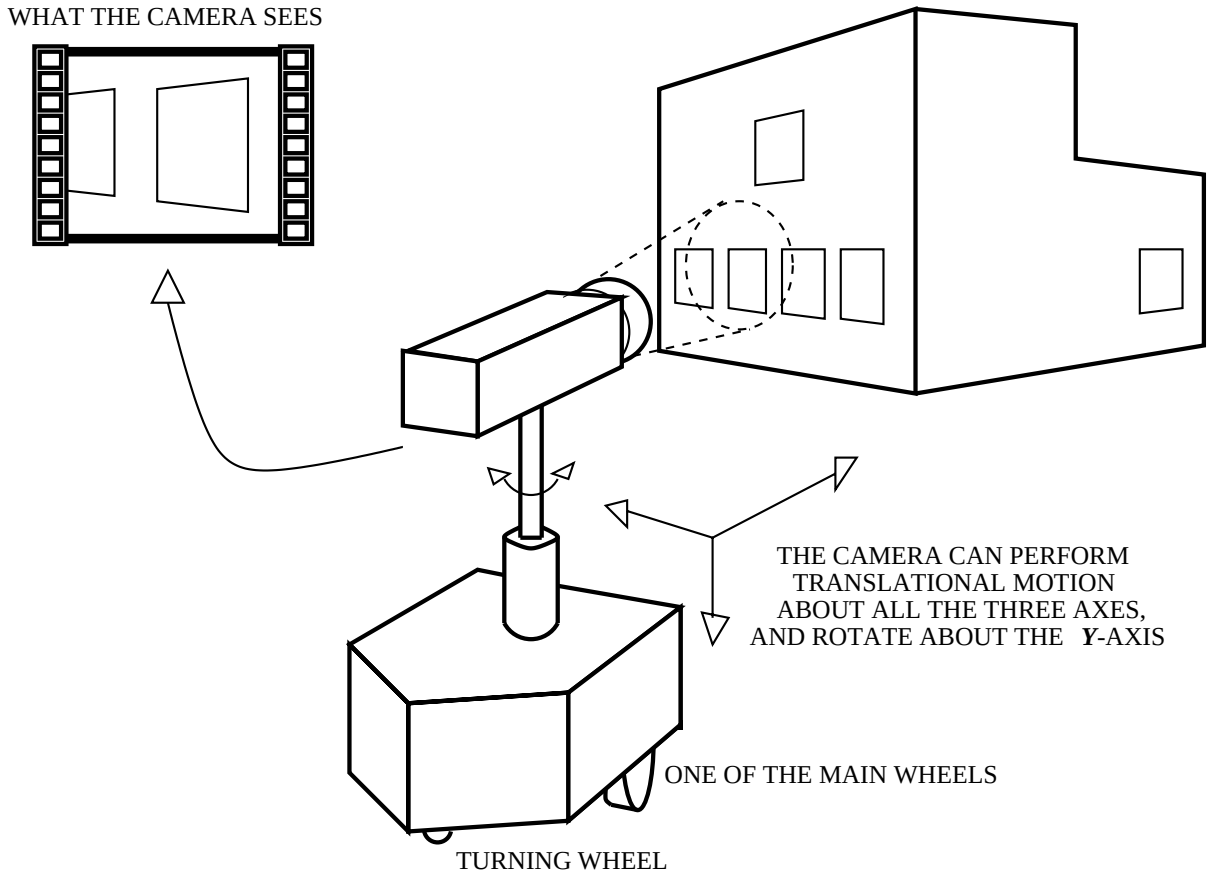


Figure 6.1: A robot with an attached camera, observing a building. The entire object does not fit in the camera's field of view. Not only is the identity of the object unknown, the robot also does not know its pose with respect to the object.

three requirements altogether. We wish to consider the case when first, the object does not fit in the camera's field of view. The camera may only view a portion of the entire object. Second, we consider a more general projective camera model. Further, we consider the 6-DOF case – three rotational and three translational degrees of freedom.

Figure 6.1 shows a robot with a camera fixed on it, observing a building. For this particular example, the number of degrees of freedom is 4 – three translational and one rotational. The identity of the object (the building) is not known. Even if the identity of the object were known, the pose of the camera with respect to the object is

not known uniquely. To resolve this ambiguity, the robot may position itself so as to observe the rightmost window. This sequence of moves may be unique to distinguish the building (from other buildings in the model base), as well as to determine its pose with respect to the building. We use a similar setup for our experimentation. However, our formulation is for the general 6-DOF case.

While our isolated 3-D object recognition system of Chapter 4 does not have the limitations associated with other systems such as [141], [106], [94], [56], [57], like the others, it suffers from two important limitations. First, all these approaches assume that the object completely fits into the camera's field of view. The second is handling the case when internal parameters of the camera are allowed to vary, either unintentionally or on purpose.

In this chapter, we specifically consider situations where a complete view of a 3-D object is not available. We consider a very general definition of the word **'part'**. What may be observed are 2-D or 3-D *parts* of objects (which are detectable using 2-D or 3-D invariants, for example), and other 'blank' or 'featureless' regions which the given set of feature detectors cannot identify. Thus, an object is composed of parts, but is not partitioned into a collection of identifiable parts.

For this problem, we present a new part-based recognition scheme using next view planning. The planning scheme uses an estimate of the current pose of the camera with respect to an observed part of the 3-D object. We use inner camera invariants (Chapter 5) for this purpose. Our hierarchical knowledge representation scheme represents an object in terms of its parts, and $\mathbf{R}$ and $\mathbf{t}$ relations between them. Each part has an coordinate system associated with it. Parts which are equivalent with respect to the feature set are grouped into **part-classes**.

The input to the system is a view of an object – this may have any number of identifiable parts. The feature detection process extracts the part-classes present in the view. At this stage, we do not know the identity of the parts in the view – we only know their part-classes. The system then generates hypotheses about the identity of the parts. We use inner camera invariants (Chapter 5) to calculate the pose of

each hypothesized part. Pose constraints from the knowledge representation scheme refine the hypotheses about the identity of the configuration of parts present in the view. We assign a probability value to each configuration hypothesis. Based on these probabilities, we calculate the probability of each object in our model base. If the probability of some object is above a fixed pre-determined threshold, the algorithm declares it to be identified.

If the probability of no object is above the threshold, or the pose of the camera with respect to the identified object is not uniquely known, we invoke the planning process to decide on a move which will disambiguate between the competing hypotheses. The system makes the required move, and repeats the above process till the pose with respect to an object is uniquely identified. The probabilistic planning process deals with all three types of uncertainties in the system - feature detection uncertainty, movement uncertainty, as well as the uncertainty in the pose value computed using inner camera invariants.

6.2 The Knowledge Representation Scheme

This section proposes a hierarchical knowledge representation scheme that encodes domain knowledge about the objects in the model base. We use the knowledge representation scheme for probability calculations, as well as in planning the next view.

6.2.1 Parts and Part-Classes

Existing part-based object recognition systems such as that of Dickinson and coworkers [58], [60], [59], [56], [57]; and Huang, Camps and Kanungo [105], [35] assume that an object is wholly composed of identifiable parts. (Of these, only [56] and [57] are active recognition systems.) For example in [105], Huang, Camps and Kanungo consider appearance-based parts. They define a part as "polynomial surfaces approximating closed, non-overlapping image regions that optimally partition the image in

a minimum description length (MDL) sense."

We consider a very general definition of the word **'Part'**. We assume that in a single view, what may be observed are 2-D or 3-D *parts* of objects (which are detectable using 2-D or 3-D invariants, for example), and other 'blank' or 'featureless' regions which the given set of feature detectors cannot identify. Thus, an object is composed of parts, but is not partitioned into a collection of identifiable parts. We now define the following term:

Part-Class A Part-Class is a set of parts, equivalent with respect to a feature set. In other words, the set of parts is partitioned into different equivalence classes with respect to a given feature set. These equivalence classes are part-classes.

We assume that an object O_i contains N_i parts. We represent the jth part of object O_i as $\rho_{i,j}$, $1 \leq j \leq N_i$. We primarily assume a geometry-based representation of parts in the model base. We may use other features, such as those based on colour or grey levels, photometric information, or reflectance-based properties, in conjunction with geometric features. Each part $\rho_{i,j}$ has a coordinate system associated with it. We assume that each part has at least n vertices. The value of n depends upon the degrees of freedom between the object and the camera. If the pose estimation problem has 6 parameters to estimate (the 6-DOF case), then n must be greater than or equal to 5 (Section 5.3.1).

6.2.2 A Part-based Hierarchical Knowledge Representation Scheme

In this section, we explain the organization of our hierarchical knowledge representation scheme. In what follows, we enumerate its salient features, and describe its various components. Figure 6.2 illustrates an example of our knowledge representation scheme.

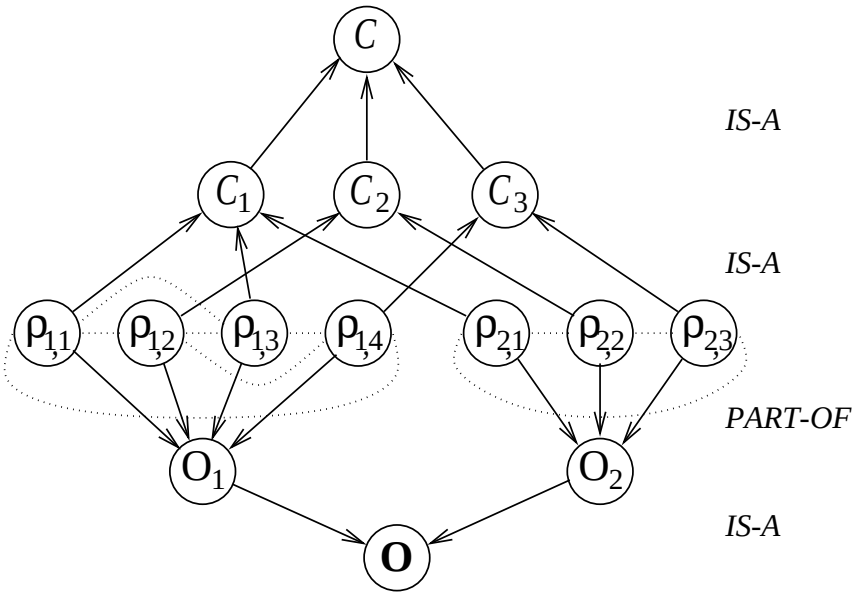


Figure 6.2: The knowledge representation scheme: an example

- **O** represents the set of all objects $\{O_i\}$. An object node O_i stores its probability, $P(O_i)$

- An object O_i is composed of N_i parts. Thus, a part $\rho_{i,j}$ $(1 \leq j \leq N_i)$ has a *PART-OF* relationship with its parent object O_i. A part node stores the 3-D Euclidean structure of its n constituent vertices $[X_i, Y_i, Z_i]^T$, $1 \leq i \leq n$, $n \geq 5$. We require $n \geq 5$ because of the requirements of the general pose estimation procedure for a part for a 6-DOF case - three rotational and three translational degrees of freedom between the object and the camera. For our experiments, we have a 4-DOF setup (as in Figure 6.1) – one rotational, and three translational degree of freedom. For this case, the requirement is $n \geq 4$ (Section 5.3.1).

- A part node has links with its nodes corresponding to its neighbouring parts. Each part has a coordinate system associated with it. Links between part nodes represent 3-D structural information between pairs of part nodes. Each link has a 6-tuple attribute associated with it – the $\mathbf{R}$ and $\mathbf{t}$ values *i.e.*, the rotations and translations needed to go from the origin of the coordinate system of one

part, to that of its neighbour. Figure 6.2 shows an example where the part nodes form a complete graph.

- $\mathcal{C}$ represents the set of all part-classes $\{\mathcal{C}_1, \mathcal{C}_2, \ \ldots \ \mathcal{C}_k\}$ for all parts belonging to the objects in the model base.

- Each part is associated with a particular part-class. We assume a function *PART_CLASS* to map the set of parts to the set of part-classes *i.e.*,

$$PART\_CLASS : \{\rho_{i,j}\} \longrightarrow \mathcal{C}$$

  There is an *IS-A* relationship between a part, and its associated part-class. Thus, a part node $\rho_{i,j}$ has exactly one link with its corresponding part-class node $\mathcal{C}_k$, and the node for the object O_i, to which it belongs.

6.2.3 Pose Estimation of a Part using Inner Camera Invariants

A multi-view 3-D object recognition system needs pose information for a given view, to generate different hypotheses corresponding to the information extracted from the view. In this chapter, we consider the problem of recognizing an object when it does not necessarily fit in the camera's field of view. Thus, a given view may contain either one or more parts of the object, or none at all. In case one or more parts are observed, we need to estimate the camera pose with respect to these parts. The next view planning module uses this information to propose a move from the current position to disambiguate between the competing hypotheses.

In Chapter 5, we use the basic perspective projection model of a pin-hole camera [78] to derive new constraints which are invariant to the internal parameters of the camera. We refer to these constraints as Inner Camera Invariants. We show in Chapter 5 that these new constraints can be used for pose estimation – without going through the often cumbersome step of camera calibration. This section describes the process of estimating the pose of a part.

A part stores with it, 3-D Euclidean coordinates of its n constituent vertices. These 3-D coordinates are with respect to its own coordinate system. *From the image of a part, one can only identify its part-class.* We may use 2-D or 3-D projective invariants, and other non-geometric features to identify the part-class corresponding to the part. The part-class could correspond to a number of parts of the objects in the model base, *i.e.*, the identity of the part is not known. We follow the following procedure to estimate the pose of the camera with respect to an observed part.

Pose Estimation of a Part: The 6-DOF Case

In Section 5.3.1, we show that six independent (inner camera) invariant measurements give us six equations – which can be solved numerically, for complete pose estimation using an uncalibrated camera and known landmarks. From the image of the part, we extract the 2-D coordinates of its n points, $n \geq 5$. For each part that the observed part-class could correspond to, we estimate the pose of the camera with respect to it. For this purpose, we associate the 2-D image coordinates of the observed part with the 3-D coordinates of the assumed part. Using this information, we write out equations in the form of Equation 5.5. Since we have at least 5 points, we get at least 6 such equations. We use constrained optimization to find solutions to these equations. This gives us *the pose of the camera with respect to the observed part.*

Since these equations are non-linear, we use a suitable constrained non-linear optimization method (Section 5.5). An important requirement for such methods is the requirement of an initial value for the solution. For this purpose, we use rough pose estimates. One may either use odometric information from the robot on which the camera is fixed, or supply this information by hand. We use constraints of a maximum allowable error, and suitable search neighbourhoods to arrive at a solution. These constraints are not dependent on any particular observed part, but are general – used for the entire experimental setup. The use of suitable search neighbourhoods for example, is to ensure that the optimization converges to the correct results only

for those parts whose 3-D coordinates could have given rise to the observed image information. We describe this further in the context of hypothesis generation, in the next section.

Pose Estimation of a Part: The 4-DOF Case

For the 6-DOF case as mentioned above, we need at least 5 points for estimating the pose of a part. If a given setup has less degrees of freedom between the object and the camera, we can relax this requirement further. In this section, we specifically consider the case of 4 degrees of freedom. This is motivated by the fact that our experimental setup has one rotational degree of freedom (rotation about the **Y**-axis), and three translational degrees of freedom (translation along the **X**-, **Y**-, and **Z**-axes, respectively).

In this case, we have 4 unknowns - R_y, t_x, t_y, and t_z. *Four* independent inner camera invariant measurements give us four equations. We solve these numerically, as in the general 6-DOF case. Hence, we need at least 4 points to be associated with a part. Our experimental set of models have planar parts – doors and windows of different polygonal shapes. It is important to note that we could not have used the general 6-DOF pose estimation procedure with planar parts, since 4 points determine a plane projectivity. Any further point from the same plane would result in a dependent set of equations.

6.3 Object Recognition and Pose Identification through Next View Planning

We are given an arbitrary view of an object in our model base. Let this view contain m parts. Our aim is to identify the given object, and the viewer pose with respect to it. Our recognition scheme is divided into two parts:

1. Probabilistic hypothesis generation, and

2. Next view planning

In what follows, we discuss these three topics in detail. Figure 6.3 describes the main steps in our part-based object recognition algorithm. The first phase of the object recognition algorithm involves initialization of all object probabilities. The system then takes an image of the given view, and identifies the part-classes corresponding to the parts present in the image. The next step is the formation of hypotheses about the identity of the observed parts. The system then computes the probability of each hypotheses. We describe our probabilistic hypothesis generation scheme in detail in Section 6.3.1. If the probability of some hypothesis is above a pre-determined threshold, then we exit and declare success. Otherwise, we invoke our search process to decide the best move from the current viewpoint, which will disambiguate between the competing hypotheses. Section 6.3.2 describes the search process and the second phase of the object recognition algorithm, in detail.

6.3.1 Probabilistic Hypothesis Generation

The input to the system is a view of the given object. As mentioned in Section 6.2.1, we use the notation $\rho_{i,j}$ to denote the jth part of the ith object. Let this given view contain m parts – ρ_{i,j_1}, ρ_{i,j_2}, ... ρ_{i,j_m}. (These observed parts are all from the same object – we are looking at the problem of recognition of an isolated 3-D object. Hence, the first subscript i is common to all the m observed parts.) From the image information, we can only identify the *part-classes* $\mathcal{C}_{k_1}$, $\mathcal{C}_{k_2}$, ... $\mathcal{C}_{k_m}$ (where $\mathcal{C}_{k_p}$ and $\mathcal{C}_{k_q}$ are not necessarily different) corresponding to each observed part, respectively ($PART\_CLASS(\rho_{i,j_p}) = \mathcal{C}_{k_p}$). The part-classes may be identified by using 2-D or 3-D projective invariants, possibly in conjunction with some other non-geometric features such as grey level or colour information, reflectance ratio values, etc. For our experimentation, we use 2-D projective invariants and grey-level information (Section 6.4). *However, our scheme is independent of the particular technique used to detect a part-class.* This configuration of visible parts could belong to any of the n objects in the

ALGORITHM identify_object_and_pose

```
(* ------ FIRST PHASE ------ *)
1.  initialize_object_probabilities();
        (* Initialize to 1/N *)
2.  image:=get_image_of_object();
3.  part_class_info:=identify_part_classes(image);
    IF NO part_class observed THEN
            make random movement; GOTO step 2;
4.  search_tree_root:=
    construct_search_tree_node(part_class_info,[I|0]);
5.  compute_hypothesis_probabilities(
            search_tree_root);  (* Eq. 6.2 *)
6.  IF the probability of some hypothesis is ≥ a
    pre-determined thresh THEN exit & call success;
7.  expand_search_tree_node(search_tree_root,
        MAX_LEVELS);  (* Section 6.3.2 *)
(* ------ SECOND PHASE ------ *)
      previous:=search_tree_root;
      expected:=get_best_leaf_node(search_tree_root);
8.  {[R|t]}:=compute_movements(expected,previous);
    make_movements({[R|t]});
    image:=get_image_of_object();
9.  part_class_info:=identify_part_classes(image);
    IF NO part_class observed THEN
       (* — backtrack — *)
       undo_movements({[R|t]});
       expected:=get_next_best_leaf_node(previous);
       GOTO step 8;
10. IF obs view does NOT correspond to expected
    THEN
      new_node:=construct_search_tree_node(
                part_class_info,{[R|t]});
    ELSE
      modify_search_tree_node_with_observation(
                 expected,part_class_info);
      new_node:=expected;
11. compute_hypothesis_probabilities(new_node);
12. IF the probability of some hypothesis is ≥ a
    pre-determined thresh THEN exit & call success;
13. expand_search_tree_node(new_node,MAX_LEVELS);
      expected:=get_best_leaf_node(previous);
      previous:=new_node;
14. GOTO step 8
```

Figure 6.3: The Object Recognition and Pose Identification Algorithm

model base. Further, this configuration could have come from many different positions within the same object O_i. Figure 6.4 shows an example. Figure 6.4(d) shows two windows with 4 corners each. From the image, the only information available to us is the presence of two 4-cornered windows (the part-class), one beside the other. This particular configuration could correspond to 204 pairs of parts for the three models *LH*, *DS* and *GH* (shown in Figure 6.4(a), (b) and (c), respectively).

We wish to generate different hypotheses corresponding to the identity of the observed configuration of parts in the image. For the first part, we construct hypotheses corresponding to every part node $\rho_{i,j}$, which has an outward link to part-class node $\mathcal{C}_{k_1}$ *i.e.*, every part which belongs to part-class $\mathcal{C}_{k_1}$. For every such part, we associate the given image coordinates of the part to the 3-D Euclidean structure of the hypothesized part (having the same part-class), and compute its pose (Section 6.2.3). As mentioned in this section, we use non-linear optimization routines with a rough initial value of the pose – either with odometric information from the robot on which the camera is fixed, or with roughly measured values. Also, we look for solutions within fixed upper and lower bounds with respect to this approximate pose estimate. These bounds are the same for any observed part – they are fixed values for the experimental setup. In our experiments for example (Section 6.4), we look for solutions that are within $\pm$ 5 degrees and $\pm$ 20mm of the rough initial value of the pose parameters. Any such part whose pose does not lie within these bounds is considered invalid. In other words, such a part could not have accounted for the observed image information. *Thus, the part pose estimation phase itself helps in a first-level pruning of the list of competing hypotheses for the interpretation of a view.*

We repeat this procedure for each observed part in the image – looking for parts in the model base which could give consistent hypotheses for the part being considered, with respect to the existing hypothesized configurations. At each stage, we use the pose information to prune out invalid hypotheses. This is another point where we use 3-D structural information from the model base. *We compare the estimated pose information for each such part, with the estimated pose information for the parts*

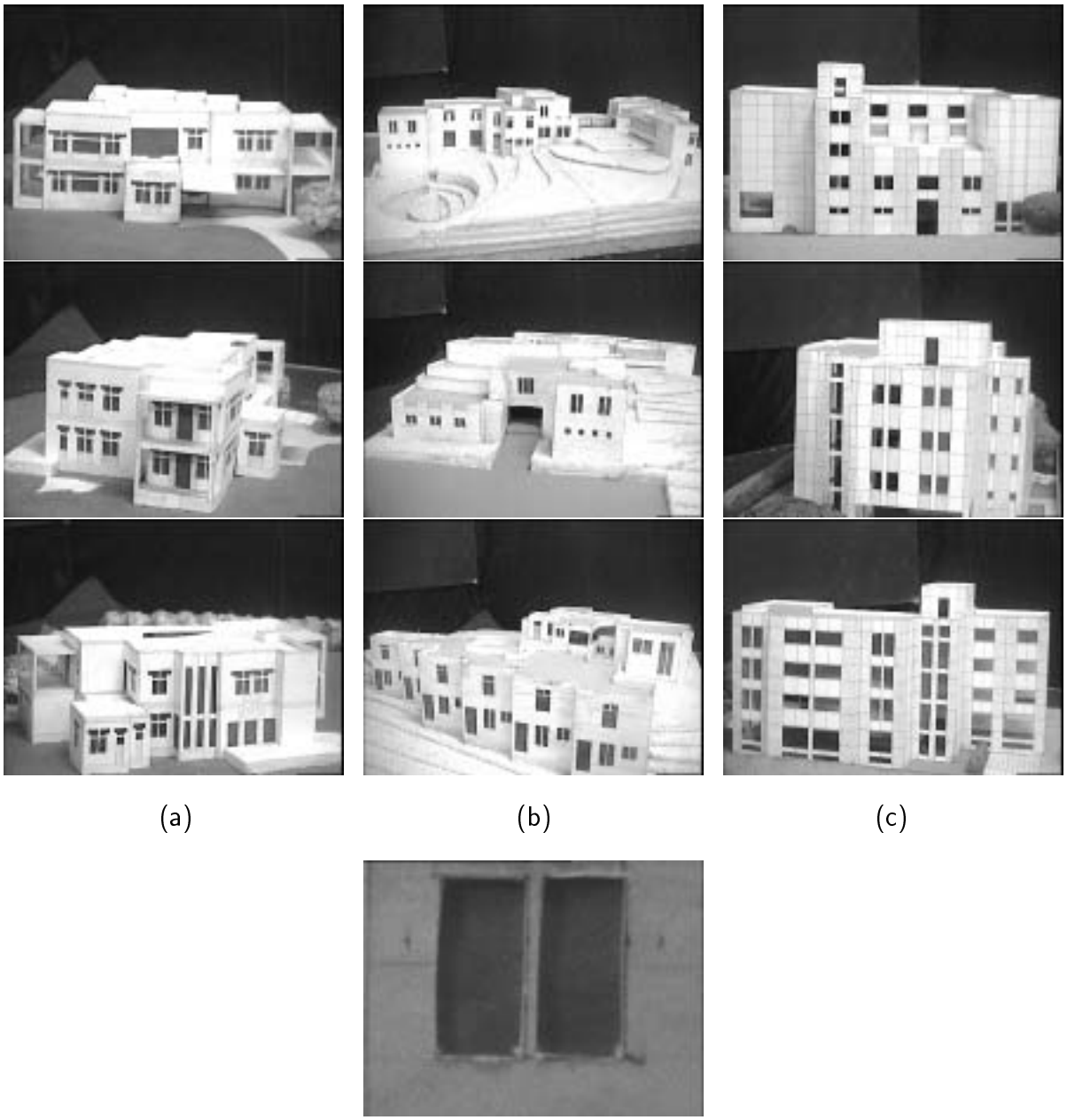


Figure 6.4: The three architectural models used for our experimentation: (a) LH, (b) DS, and (c) GH, respectively. (d) A given view of an object. Only a portion of the object is visible. This could have come from any of the three models.

already considered thus far. If this is not consistent with the $\mathbf{R}$ *and* $\mathbf{t}$ *link information between parts in the model base, we reject this hypothesis, and proceed to the next.* In our experiments for example (Section 6.4), we check if the experimentally obtained angles are within $\pm$ 5 degrees, and distances are within $\pm$ 20mm of the values in the model base. *This also helps in accounting for some inaccuracies in the pose estimation process.* Thus, one does not need to use joint projective invariants between observed parts – our method relies directly on 3-D pose estimates to check consistency relations between a group of parts.

At the end of this phase, we are left with a list of hypotheses of part configurations, which could have given rise to the observed configuration of parts in the given view. The next section describes the procedure for computing the probability of each such hypothesis.

a priori Probability Calculations

The given view consists of m parts $\rho_{i,j_1},\ \rho_{i,j_2},\ \ldots\ \rho_{i,j_m}$. The hypothesis generation stage computes a list of valid hypotheses of part configurations (of size K, say), which could have given rise to the observed view. First, we compute *a priori* probabilities for each such hypothesis. For N objects in the model base, the *a priori* probability of each object before taking the first observation, is $1/N$. We need estimates of the *a priori* probabilities of different configurations of parts that may occur.

$$P(\rho_{i,j_1},\ \rho_{i,j_2},\ \ldots\ \rho_{i,j_m}) = P(O_i)\ \cdot\ P(\rho_{i,j_1},\ \rho_{i,j_2},\ \ldots\ \rho_{i,j_m} \mid O_i) \tag{6.1}$$

We may form estimates of $P(\rho_{i,j_1},\ \rho_{i,j_2},\ \ldots\ \rho_{i,j_m} \mid O_i)$ by taking a very large number of views of the given object from different positions, and different values of the internals of the camera (the focal length, for example on which the field of view of the camera depends) — this is done *off-line*, before taking the first observation. However, a satisfactory estimation of *a priori* probabilities using this method, is difficult. In the following section, we propose a method of approximating these *a priori* probabilities.

An Approximation with Assumptions about the Nature of 3-D Object Models

If we make some assumptions about the nature of the 3-D object models, we can formulate an approximate method to estimate the *a priori* conditional part-configuration probabilities. Let us consider the domain of objects with planar faces.

This method considers relative visible areas of different parts in the model base. We base the *a priori* probability calculations for a part on its area. (The faces and the parts on it – both are planar). Given the degrees of freedom between the object and the camera in the setup, we consider the set of all faces that can be viewed. The *a priori* probability of a part is considered to be the ratio of its area to the total area of all faces that can be viewed. To compute the *a priori* probability of a configuration of adjacent parts, we consider their total relative area. These measurements are done with respect to the 3-D model, not on any image-based features.

The rationale behind this approximation is as follows. Let us consider the situation from the point of view of a single part. Ideally, a proper estimate of the *a priori* probability of the part would involve taking a large number of observations. The pose of the camera for these observations would be drawn randomly from a uniform sampling of the camera parameter space – external, as well as internal. We may compute this by using the relative frequency interpretation of probability – the ratio of the number of times the part is observed, to the total number of observations. For external parameters for example, one would have as many observations with the camera observing the object from the right side, as there would be from its left. Hence, one may have a good estimate of the *a priori* probability by looking at the part simply head-on.

The situation is similar for observing the part from various viewpoints, by varying the field of view of the camera. We may assume that the part is such that it always remains within the field of view of the camera (every camera has an allowable range of values within which the field of view can be varied). Intuitively, a larger part is more

likely to be visible in a larger number of observations, compared to a smaller part. Thus, one may form an approximate estimate of the *a priori* probability of observing the part as the ratio of its area, to the ratio of the total area of the object being imaged. This argument can now be extended to a configuration of parts, instead of a single part.

We have experimented with three architectural models (Figure 6.4). Such assumptions about the nature of the object model are valid for this domain of objects. The parts are all 2-D entities – doors and windows of different shapes. We have experimented with a 4-DOF setup between the camera and the object – three translational, and rotation about the **Y**-axis. *We emphasize, however that our formulation is independent of the particular setup – the only requirement is the availability of* a priori *configuration probabilities.*

a posteriori Probability Computations

For an observation, we compute the *a posteriori* probability of each hypothesized configuration using the Bayes rule:

$$P(\rho_{i,j_1},\ \rho_{i,j_2},\ \ldots\ \rho_{i,j_m} \mid \mathcal{C}_{k_1},\ \mathcal{C}_{k_2},\ \ldots\ \mathcal{C}_{k_m}) = \ Numerator/Denominator \qquad (6.2)$$

where $Numerator$ is given by

$$P(\rho_{i,j_1},\ \rho_{i,j_2},\ \ldots\ \rho_{i,j_m})\ \cdot\ P(\mathcal{C}_{k_1},\ \mathcal{C}_{k_2},\ \ldots\ \mathcal{C}_{k_m} \mid \rho_{i,j_1},\ \rho_{i,j_2},\ \ldots\ \rho_{i,j_m})$$

and $Denominator$, by

$$\sum\ [\ P(\rho_{l,j_1},\ \rho_{l,j_2},\ \ldots\ \rho_{l,j_m})\ \cdot\ P(\mathcal{C}_{k_1},\ \mathcal{C}_{k_2},\ \ldots\ \mathcal{C}_{k_m} \mid \rho_{l,j_1},\ \rho_{l,j_2},\ \ldots\ \rho_{l,j_m})\ ]$$

The summation in $Denominator$ is for all objects O_l, and all possible configurations of parts within the object. Our knowledge representation scheme simplifies the calculations of Equation 6.3. Because of the $IS-A$ relation between a part and a part-class in our knowledge representation scheme (Section 6.2), each of the terms

$P(\mathcal{C}_{k_1},\ \mathcal{C}_{k_2},\ \ldots\ \mathcal{C}_{k_m} \mid \rho_{l,j_1},\ \rho_{l,j_2},\ \ldots\ \rho_{l,j_m})$ is 1 for all parts belonging to a particular part-class and 0, otherwise.

We now compute the *a posteriori* probability of each object in the model base:

$$P(O_l) = \sum P(\rho_{l,j_1},\ \rho_{l,j_2},\ \ldots\ \rho_{l,j_m} \mid \mathcal{C}_{k_1},\ \mathcal{C}_{k_2},\ \ldots\ \mathcal{C}_{k_m}) \tag{6.3}$$

The summation is for all configurations of parts $\rho_{l,j_1},\ \rho_{l,j_2},\ \ldots\ \rho_{l,j_m}$ belonging to object O_l, which could have given rise to the given view containing part-classes $\mathcal{C}_{k_1},\ \mathcal{C}_{k_2},\ \ldots\ \mathcal{C}_{k_m}$. Each object node in the knowledge representation scheme updates its probability with values from Equation 6.3.

For a hypothesis list of size K, the calculations of Equations 6.1 and 6.3 take $O(K)$ time. For Equation 6.2, the denominator calculation (sum) takes $O(K)$ time. The division has to be performed for each of the K hypotheses. Hence calculations for Equation 6.2 also take $O(K)$ time.

6.3.2 Next View Planning

If the probability of no hypothesis is above a pre-determined threshold, we have to take the next view in order to disambiguate between the competing hypotheses. One needs to plan the best move out of the current state, subject to possible memory and processing limitations.

The Search Process

The state of the system may be described in terms of the the following parameters:

1. The competing view interpretation hypotheses, and
2. The set of $\mathbf{R}$ and $\mathbf{t}$ movements made thus far.

The planning process aims to determine a move from the current step, which would uniquely correspond to exactly one part-configuration for one object. As for the next view planning scheme for Chapter 4, we pose the planning problem as a forward

search in the state space. The aim is to get to a leaf node – one corresponding to a unique part-configuration. A search tree node is expanded for each part in a view interpretation hypothesis. The moves from a viewpoint are based on the pose information calculated using inner camera invariants (Sections 5.2 and 6.3.1). All these 'moves' are logical, or virtual moves in the search space.

First, we construct a search tree node corresponding to the first observation. The search process proceeds as follows. Here, we assume that the principal point of the camera is somewhere near the centre of the image. However, we do not assume that we know its value in any way, nor do we assume it to be fixed. The first move gets the expected part in the camera's line of view. The subsequent logical moves are from the current expected part to its neighbours, using the $\mathbf{R}$ and $\mathbf{t}$ relations between parts in the knowledge representation scheme (Section 6.2). Thus, the only significance of the above assumption is *to maximize the chance of the expected part to be always present in the camera's field of view.* This is important for the following two reasons:

1. Getting the expected part into the camera's line of view offers some robustness to subsequent movement errors, due to which the camera may be positioned at a point slightly off from its intended position.

2. A zoom-in/zoom-out, or focusing operation may be performed. This changes the effective focal length of the camera system and hence, the field of view. If the principal point is near the centre of the expected part, chances of having the expected part in the camera's field of view and hence detecting it, are higher than otherwise.

Figure 6.5 shows an example of a partially constructed search tree. In this example, the root node corresponds to an observation. This view contains 4 parts, belonging to part-classes $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_1$, and $\mathcal{C}_1$, respectively. The system generates three hypotheses corresponding to this observation. The search tree root node is at level 0 of the search tree. The search tree nodes at level 1 are on basis of the (logical) moves to get the camera in line with the centre/centroid of the expected part. Let

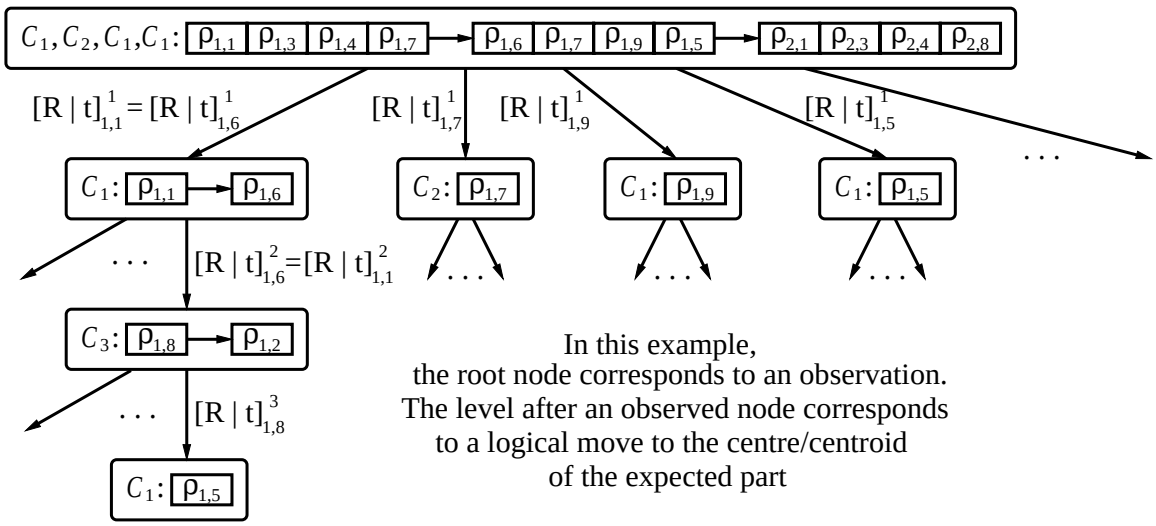


Figure 6.5: A partially constructed search tree

us consider search tree node expansion along a particular path. In this example, the estimated pose for the two parts $\rho_{1,1}$ and $\rho_{1,6}$ (the first observed parts in the first two view interpretations, respectively) is the same, and the parts have the same dimensions. Hence, there are two possible interpretations in the corresponding first-level node. One particular movement to an adjacent part does not remove the view interpretation ambiguity for this node. Hence, again there are two possible interpretations in the corresponding node on level 2. One particular move finally resolves the ambiguity, resulting in the corresponding leaf node at level 3.

Due to the exponential space and time complexity corresponding to search tree expansion, one may expand the search tree only to a fixed depth (MAX_LEVELS in Figure 6.3). We now use three stages of filtering to get the best leaf node or pseudo-leaf node (a node which has no child nodes, but does not correspond to a unique part – it has not been expanded due to the fixed maximum search depth from a node). First, we consider those leaf/pseudo-leaf nodes which lie along a path from the most probable hypothesized view interpretation in the search tree node corresponding to the previous observation (the 'previously observed node'). For each node in the search tree, we assign the weight s^{level}, where s represents the number of

hypothesized view interpretations corresponding to this node, and *level* is the search tree level (depth) the node lies on. The rationale behind this strategy is to favour nodes with low ambiguity among the different hypothesized view interpretations, and those corresponding to less movement cost. A leaf/pseudo-leaf node also stores the sum total of the weights of all nodes which lie on the path to it, from the previously observed search tree node. From among those leaf and pseudo-leaf nodes selected in the first stage, we select those with minimum total weight. The third stage of filtering concerns a setup limitation – our camera setup can achieve more accurate translational movement compared to a rotational one. From among the second stage selections, we choose a node having the least number of rotational movements.

The Second Phase of the Object Recognition Algorithm

Using the above three steps of filtering, we determine the best leaf node in the search tree. This represents the best move from the current viewpoint which aims to get us to a position corresponding to a unique view interpretation. Now, the system makes an actual move from the current viewpoint, corresponding to the best leaf node. The system makes the required movements $\{\langle R_x, R_y, R_z, t_x, t_y, t_z \rangle\}$, and takes an image at this position (Step 8 in the algorithm of Figure 6.3). We then find out the part-class information corresponding to this image.

Similar to the process in Section 6.3.1, we generate different interpretation hypotheses corresponding to this view, for the particular sequence of $\mathbf{R}$ and $\mathbf{t}$ movements taken to reach this particular viewpoint. We now check if this observed configuration of parts corresponds to the best leaf/pseudo-leaf node. Since we do not make any assumption regarding the knowledge of the camera internal parameters or their constancy, we do not make any assumptions about the field of view of the camera. We simply check if the observed configuration of parts corresponds to the expected node. This is a simple way to make the system robust to slight movement errors, or intentional/unintentional changes in the focusing and field of view.

Since we do not predict any view which might be observed, even if some parts in the vicinity of the expected part are not detected (due to feature detection errors), this does not affect the system in any way. *Another important consequence of this fact is the robustness of the system to the presence of clutter in a view.*

If the current observation corresponds to the expected search tree node, we update the search tree node with the information from the current view (Step 12 in Figure 6.3). Otherwise, we construct a new search tree node corresponding to the part-class information, and the sequence of $\mathbf{R}$ and $\mathbf{t}$ moves made to reach this position from the previous one. This new node is made a child node of the previously observed node. We compute the probabilities of each view interpretation hypothesis, as in Section 6.3.1. If the probability of some hypothesis is above the pre-determined threshold, the algorithm terminates. The pose of the camera with respect to the object is the one corresponding to this hypothesis, and the parts corresponding to the view are the ones in this view interpretation.

If the probability of no hypothesis is above the threshold, this node needs to be expanded further. The system finds out the best leaf node again, and repeats the entire process.

6.3.3 The Object Recognition Algorithm: A Discussion

An active object recognition system needs to deal with uncertainty in feature detection, pose estimation, sensor movement, as well as the interpretation of a particular view. In this section, we summarize some salient features of our object recognition strategy.

Our formulation is independent of the particular setup used for experimentation. We consider the most general case – 6 degrees of freedom between the object and the camera. The entire object may not lie within the camera's field of view. We make no assumptions about the knowledge of the internal camera parameters, or their constancy. The use of inner camera invariants makes the algorithm invariant

to any zoom-in/zoom-out, or focusing operation of the camera. Our part pose estimation procedure itself accounts for a first level pruning of different hypotheses for the interpretation of the given view. We directly use the 3-D pose estimation information in conjunction with our hierarchical knowledge representation scheme, for getting consistent view interpretation hypotheses. The probability values of different entities in our probabilistic hypothesis generation scheme also help in pruning the search space. We have a simple strategy to account for small pose estimation errors – we just check if the $\mathbf{R}$ and $\mathbf{t}$ link information between parts matches the estimated pose values within fixed error bounds. We make no assumptions about field of view of the camera at any stage – the planning is with respect to each observed part. The planning to get in line with the centre/centroid of the expected part provides some immunity to small movement errors and the changes in the camera's field of view. Since we do not predict any particular view that may be observed, even if some parts are not detected due to feature detection errors, it does not affect the system in any way. This fact also imparts *robustness to the system in the presence of clutter.*

In the search process, search tree node expansion is always finite because of the following reason. The number of parts in any object is finite. Further, there are no cycles in the search tree. No part is repeated along any path in the search tree. Thus, there can be no search tree expansion indefinitely oscillating between a set of parts. The next view planning module acts in conjunction with our probabilistic hypothesis generation scheme. We use three levels of filtering to obtain the best leaf node – which will best disambiguate between the competing view interpretation hypotheses.

The reactive nature of our strategy incorporates all previous movements and observations in the probabilistic hypothesis generation scheme (Section 6.3.1), as well as in the next view planning process (Section 6.3.2). If the observed view corresponds to the most probable view interpretation hypothesis at a particular stage, our search process uniquely identifies the object and its pose, in the following step (assuming no feature detection error for the expected part). Even if the observed view does not correspond to the most probable view interpretation hypothesis, our algorithm re-

fines the list of hypotheses at each stage. The advantage of a reactive system over an off-line planner is the ability to react to unplanned situations. At the end of each observation, we create a new view interpretation hypothesis list. This is based on both the current observation, as well as the past observation history and the movements made thus far.

6.4 Experimental Results and Discussion

Our experimental setup has a camera system with 4 degrees of freedom - translations along the **X**-, **Y**- and **Z**-axes, and rotation about the **Y**-axis (Figure 6.1). Hence, we use the 4-DOF formulation for the pose estimation of a part (Section 6.2.3). We have experimented with a set of architectural models shown in Figure 6.4.

We have chosen as (2-D) parts the doors and windows of different shapes and sizes in the models. We have chosen this set of models because of the large number of parts grouped into a few part-classes – this ensures a very high degree of interpretation ambiguity associated with a particular view of a few parts of the given object. Model *LH* (Figure 6.4(a)) has 167 parts, model *DS* (Figure 6.4(b)) has 170, while model *GH* (Figure 6.4(c)) has 122. Figure 6.6 shows the 7 different part-classes these 459 parts (of different sizes) correspond to. The 7 part-classes, with the number of parts corresponding to each, are $DW4(374)$, $DW6L(24)$, $DW6R(24)$, $OPEN(21)$, $DW8HANDLE(6)$, $DW8T(6)$, and $DW12(4)$, respectively.

Given a particular view of the object, we first segment the image using sequential labeling [103], [97]. Then we detect corners as intersection of lines on the boundaries of 'dark' regions. A requirement for the pose estimation of a part (Section 6.2.3) is for the part to have at least 5 vertices for a general 6-DOF case, and 4 vertices, for a 4-DOF case. We use 2-D projective invariants using the canonical frame construction method [175] for recognizing all part-classes (except the 4-cornered ones – $DW4$ and $OPEN$). This involves transforming the 2-D coordinates of any *four* points corresponding to the corners, to the corners of a unit square. This determines a projective

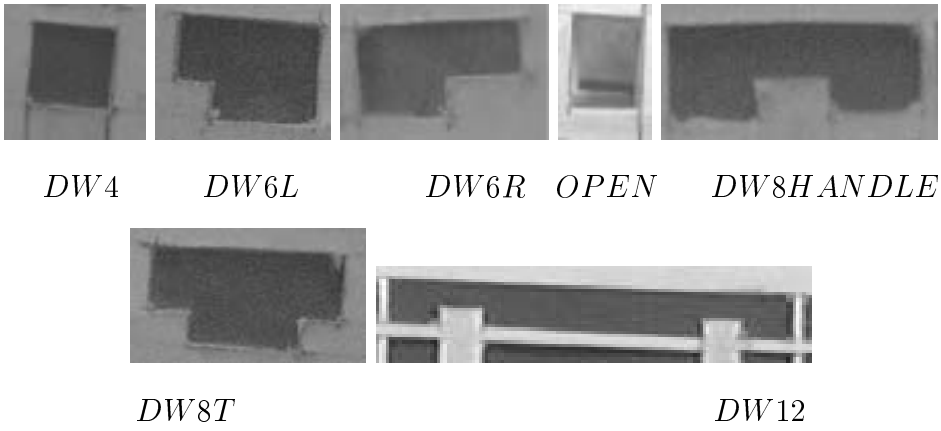


Figure 6.6: The 7 part-classes which the 459 parts belong to, for our model base: $DW4$, $DW6L$, $DW6R$, $OPEN$, $DW8HANDLE$, $DW8T$, and $DW12$, respectively in row-major order.

transformation to a canonical frame. We compute the coordinates of the rest of the n - 4 points in this coordinate frame. These coordinates are projective invariants [175]. Each of our parts thus gives a different vector of invariant coordinates. We use these invariant values, to identify part-classes which have more than 4 corners. To identify the 4-cornered part-classes $DW4$ or $OPEN$, we use the grey level information at a region near its centroid. *We emphasize that our recognition scheme is independent of the particular type of parts used, and the method to recognize a part-class.*

Sections 6.4.1 and 6.4.2 show some results of experimentation with the objects in our model base. The detected corners and parts are shown superimposed. Each of these experiments shows that *the planning to get to the centre of the expected part (Section 6.3.2) provides some immunity to small movement errors and changes in the camera's field of view.* For our experiments, we have adopted a stricter criterion for program termination than the probability of a particular hypothesis in an observed node being above a threshold. We stop when there is exactly one hypothesis possible for the observed node.

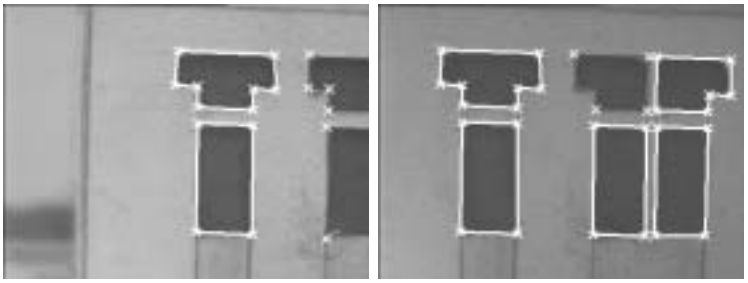


Figure 6.7: Experiment 1: The sequence of moves required to identify the object and its pose. The failure to detect a part does not affect the system (details in text).

6.4.1 Experiments with a Small Degree of Ambiguity Corresponding to the First View

In this section, we present results of experiments where the interpretation ambiguity associated with the first view is relatively small. We show an example where the non-detection of a part (due to a feature detection error) does not affect the performance of the system.

Experiment 1

The initial view in Figure 6.7 shows two parts with part-classes $DW8T$ and $DW4$. For the first part alone, there are 6 hypotheses. Now, the system considers the hypotheses when information about the second part is also included. Four of these are pruned out since the estimated pose of the second part comes out to be invalid (Section 6.3.1). The system proposes a move to identify the pose uniquely, and takes the required movements. The view observed is the second image in Figure 6.7.

This view contains the expected part, as well as a couple of neighbouring parts. Here, the system fails to detect the part corresponding to part-class $DW6L$. The presence of the neighbouring parts (their part-classes, and their pose information) is consistent with that of the expected part (centre of the bottom row). *Thus, this feature detection error does not affect the performance of our algorithm in any way.*

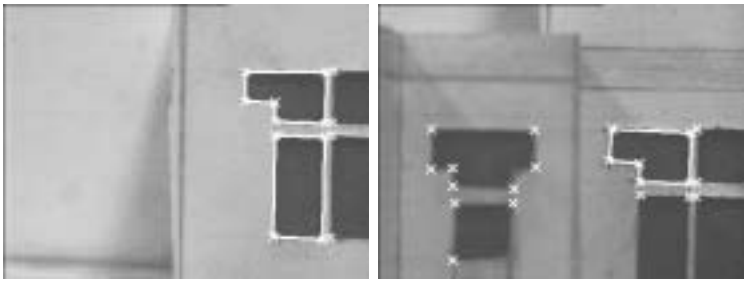


Figure 6.8: Experiment 2: The sequence of moves required to identify the object and its pose. The failure to detect a part does not affect the system (details in text).

Experiment 2

Figure 6.8 shows another example of this robustness to feature detection errors. For the upper part in the first image, 24 hypotheses are proposed. The pose estimation phase prunes out 22 of them – the 3-D vertex coordinates of these parts could not have resulted in the observed image coordinates of the detected part (Section 6.3.1). Thus, the first image results in two consistent hypotheses for the two parts. The system takes the next move in accordance with the best leaf node.

The hypothesized view interpretation for the second view (the second image in Figure 6.8) is consistent with the part-class and pose information extracted from the image. This move corresponds to a unique hypothesis. Consequently, probability calculations result in the probability of object LH being 1. The pose of the camera with respect to the identified part *LH_L_14* is $\langle\ -4.6^o, -2.54mm, 15.02mm, 139.98mm\ \rangle$. This second view illustrates another example of feature detection errors – the two windows on the left (corresponding to part-classes $DW8T$ and $DW4$, respectively) are not detected. This, however does not affect the performance of the system in any way.

6.4.2 Experiments with a High Degree of Ambiguity Corresponding to the First View

Scenes containing a small number of parts belonging to part-class $DW4$, have a very high degree of ambiguity associated with their interpretation. This is due to the large number of parts belonging to part-class $DW4$ – 374. Due to this reason, we use a depth-restricted search tree expansion method (Experiments 1 and 2 consider no restriction on the size of the search tree that can be created at a particular stage – the total number of search tree nodes for the two are 51 and 60, respectively.) Even with the depth restriction, the number of search tree nodes is quite large for these experiments *e.g.*, 987 for Experiment 5. In this section, we present results of experiments where the first view contains a pair of $DW4$ parts (Experiments 3 – 5), and finally, only one $DW4$ part (Experiments 6 – 11). We present an example of the invariance of our algorithm to the field of view of the camera. We also show an example where all the parts in the first view do not come from one plane. Our algorithm has no restriction regarding where the parts could lie in 3-D space.

Experiment 3

Figure 6.9 shows results of an experiment, where the initial view contains two adjacent parts belonging to part-class $DW4$. For the leftmost part, 374 hypotheses are proposed in all. This reduces to a total of 328 hypotheses for the first view, when the part beside it is also considered. The system plans the next move accordingly. This process is repeated till the system identifies a part and its pose, uniquely. The figure shows the sequence of moves (in row major order) in order to recognize the object and its pose. In this case, a total of 6 moves from the first position are required for unambiguous recognition.

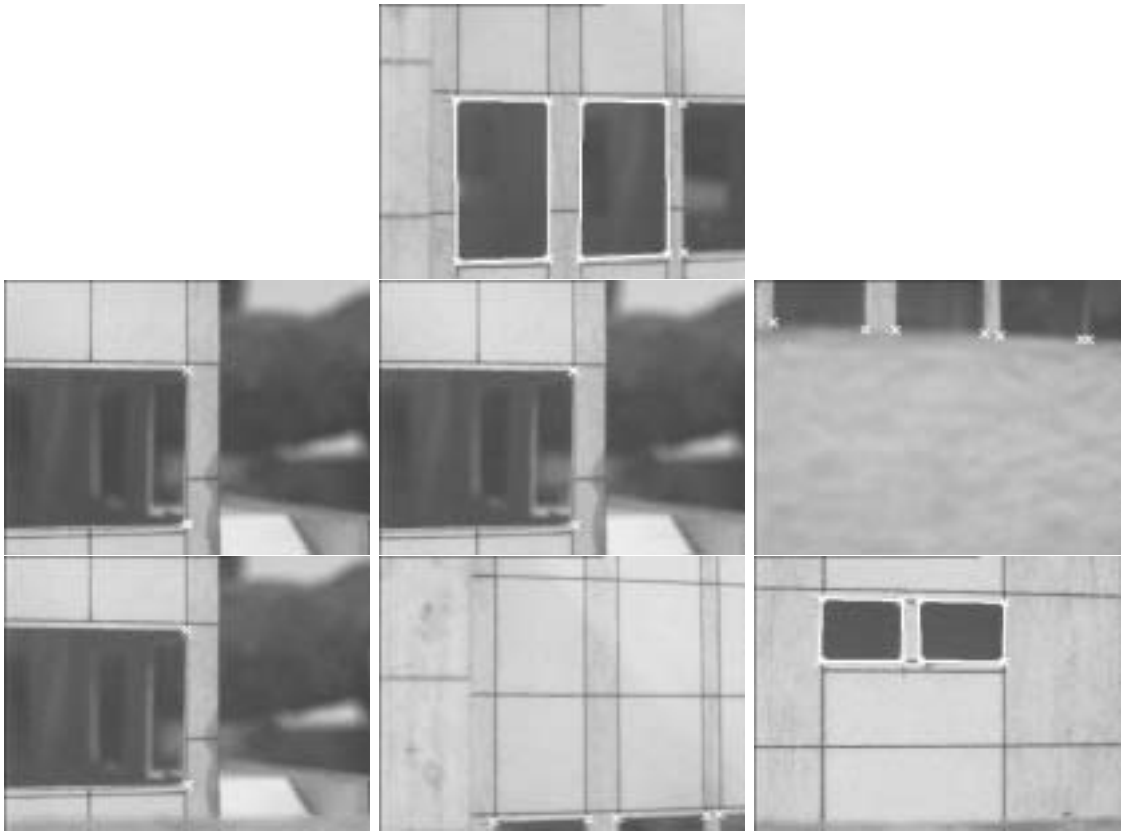


Figure 6.9: Experiment 3: The sequence of moves (in row major order) required to identify the object and its pose

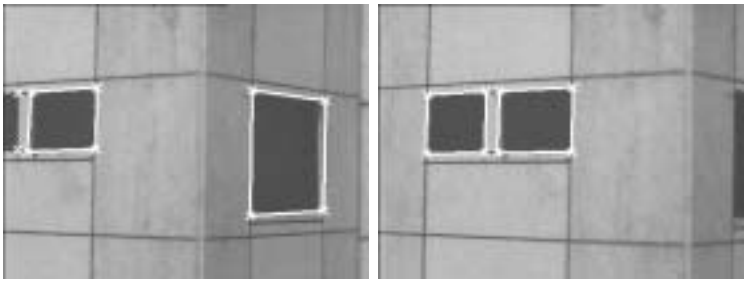


Figure 6.10: Experiment 4: The sequence of moves required to identify the object and its pose. The parts in the initial view do not lie in the same plane.

Experiment 4

The parts visible to the system in a view could have come from any 3-D configuration. Here, we present an example where the parts in the initial view do not come from the same plane. Figure 6.10 shows the moves taken by the system to identify the object, and the pose of the camera with respect to it. For the first part in the first image, 374 hypotheses are proposed, out of which the part pose estimation prunes out all but 115 hypotheses. The information from the second part results in a hypothesis list of size 87. The system plans a move to disambiguate between the different hypotheses. This corresponding move takes us to a view (the second image in Figure 6.10), whose view interpretation is unique.

Experiment 5

The use of inner camera invariants for pose estimation allows us to consider situations where the internal parameters of the camera may be varied on purpose, or unintentionally. Further, our recognition strategy does not make any assumptions about the field of view of the camera. For this experiment (Figure 6.11), we changed the zoom parameter of the camera, thus changing the effective focal length of the camera system and consequently, its field of view. The first view could have come from 257 configurations of two adjacent parts with part-class $DW4$. We need three image processing operations (2 moves) to recognize the object and its pose, uniquely.

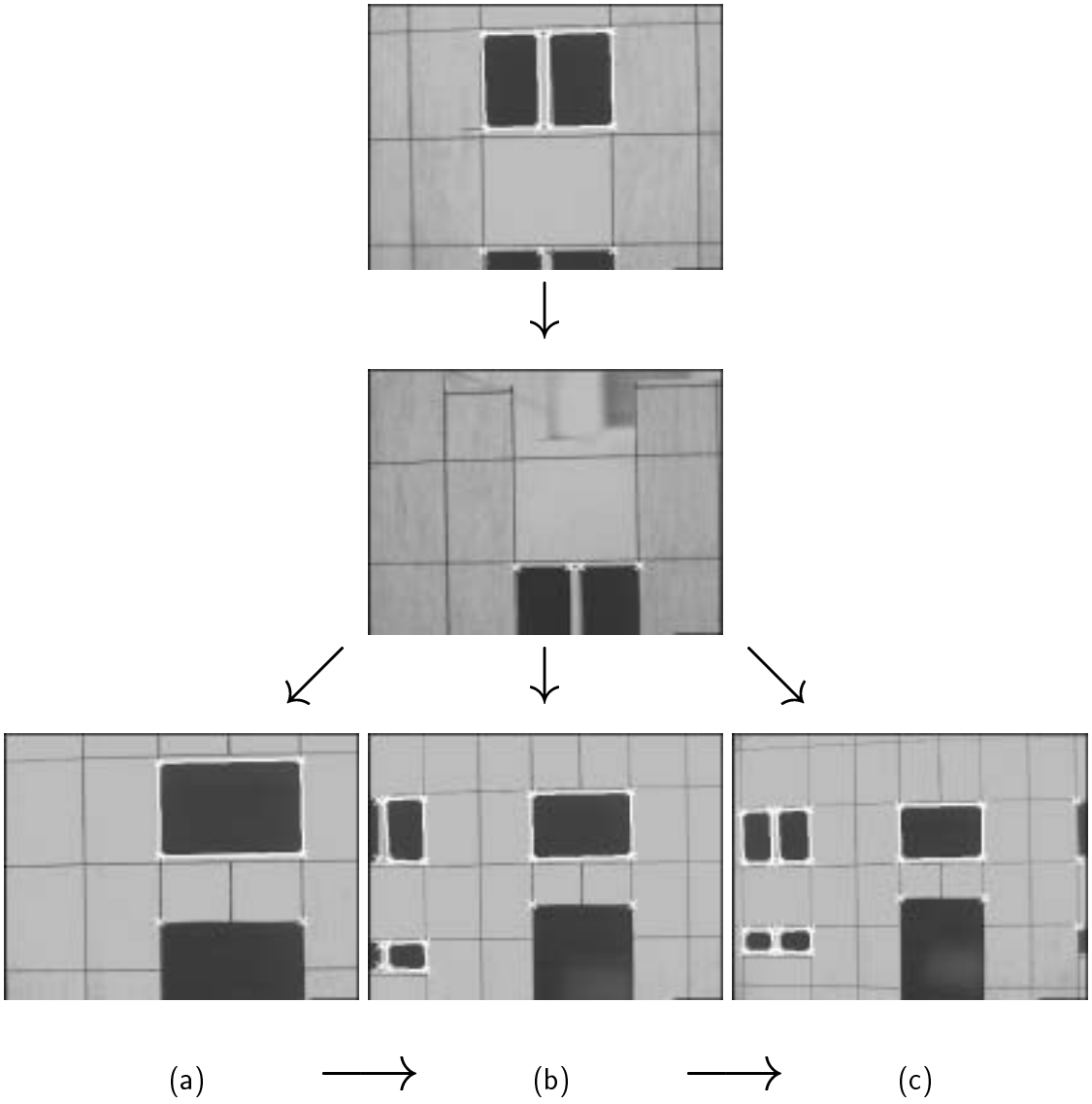


Figure 6.11: Experiment 5: The sequence of moves required to identify the object and its pose. In this experiment, three views are needed to uniquely recognize the object and its pose. For the same first two views, we progressively zoom out the camera in three stages. (a), (b) and (c) depict the three views which the camera sees, for the third view. This does not affect the recognition system in any way (details in text).

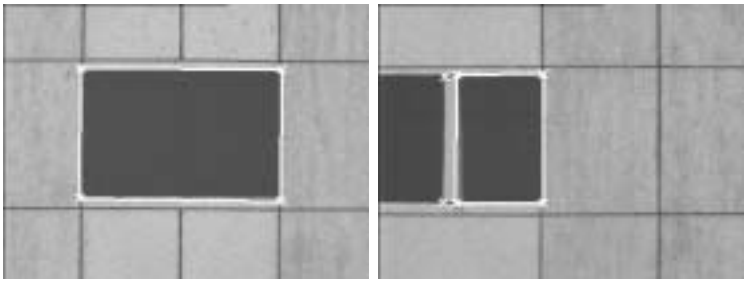


Figure 6.12: Experiment 6: The sequence of moves required to identify the object and its pose.

In this case, we repeated the experiment for various values of the camera zoom-out at the third camera station. The expected part is the large 4-cornered window, *GH_W_15*. Since our strategy does not make any assumptions about the field of view of the camera, the recognition results are the same in each of the cases — (a), (b) and (c) in Figure 6.11. Further, the camera pose with respect to part *GH_W_15* in these three cases are

$\langle\ 9.425^o,\ -22.000mm,\ -9.999mm,\ 150.000mm\ \rangle$,

$\langle\ 9.888^o,\ -22.000mm,\ -9.999mm,\ 150.000mm\ \rangle$, and

$\langle\ 9.896^o,\ -22.000mm,\ -9.999mm,\ 150.000mm\ \rangle$, respectively.

This shows the advantage of inner camera invariants for pose estimation.

Experiments 6 – 11

From Experiment 6 onwards, we start with an initial view where only one part with part-class $DW4$ is visible. In each of these cases, the visible part could have corresponded to any of the 374 parts with part-class $DW4$. For Experiment 6 (Figure 6.12), the first level pruning using our part pose estimation procedure (Section 6.2.3) prunes the number of hypotheses to 189. The next view finds the expected part, *GH_W_12*.

The first level pruning for the first view in Experiment 7 (Figure 6.13) results in the number of hypotheses reducing from 374 to 344. In this case also, two views are

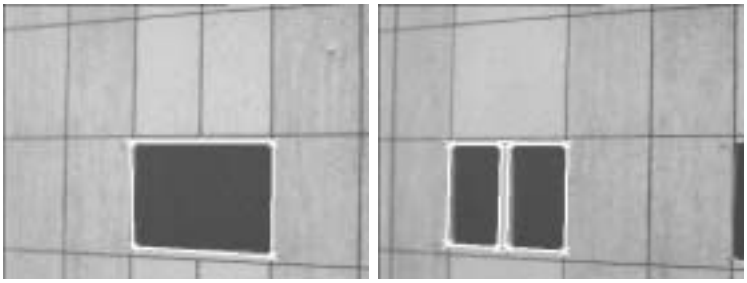


Figure 6.13: Experiment 7: The sequence of moves required to identify the object and its pose.

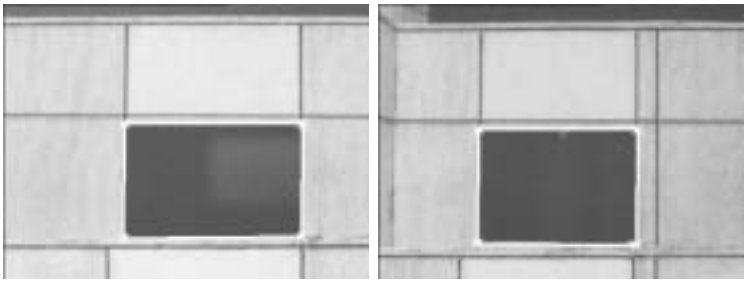


Figure 6.14: Experiment 8: The sequence of moves required to identify the object and its pose.

necessary for unique identification of the object and the camera pose. For the second view, 8 hypotheses are left. Of these, the part pose information results in the unique identification of the parts as being *GH_W_10* and *GH_W_12*, respectively.

Experiment 8 (Figure 6.14) again shows an example of two views sufficing for unique recognition of the object and its pose. After the first level pruning on the basis of part pose estimation, we are left with 71 hypotheses. The system finds the expected part *GH_W_9* in the second view.

Experiments 9 and 10 (Figures 6.15 and 6.16) show examples where the system requires 5 moves to recognize the object and its pose. The first level pruning figures for these two experiments are 226 and 304, respectively. This pruning on the basis of part pose estimation leaves only 5 hypotheses for the third view in Figure 6.16.

In these experiments, we show an example of the ability of our system to perform

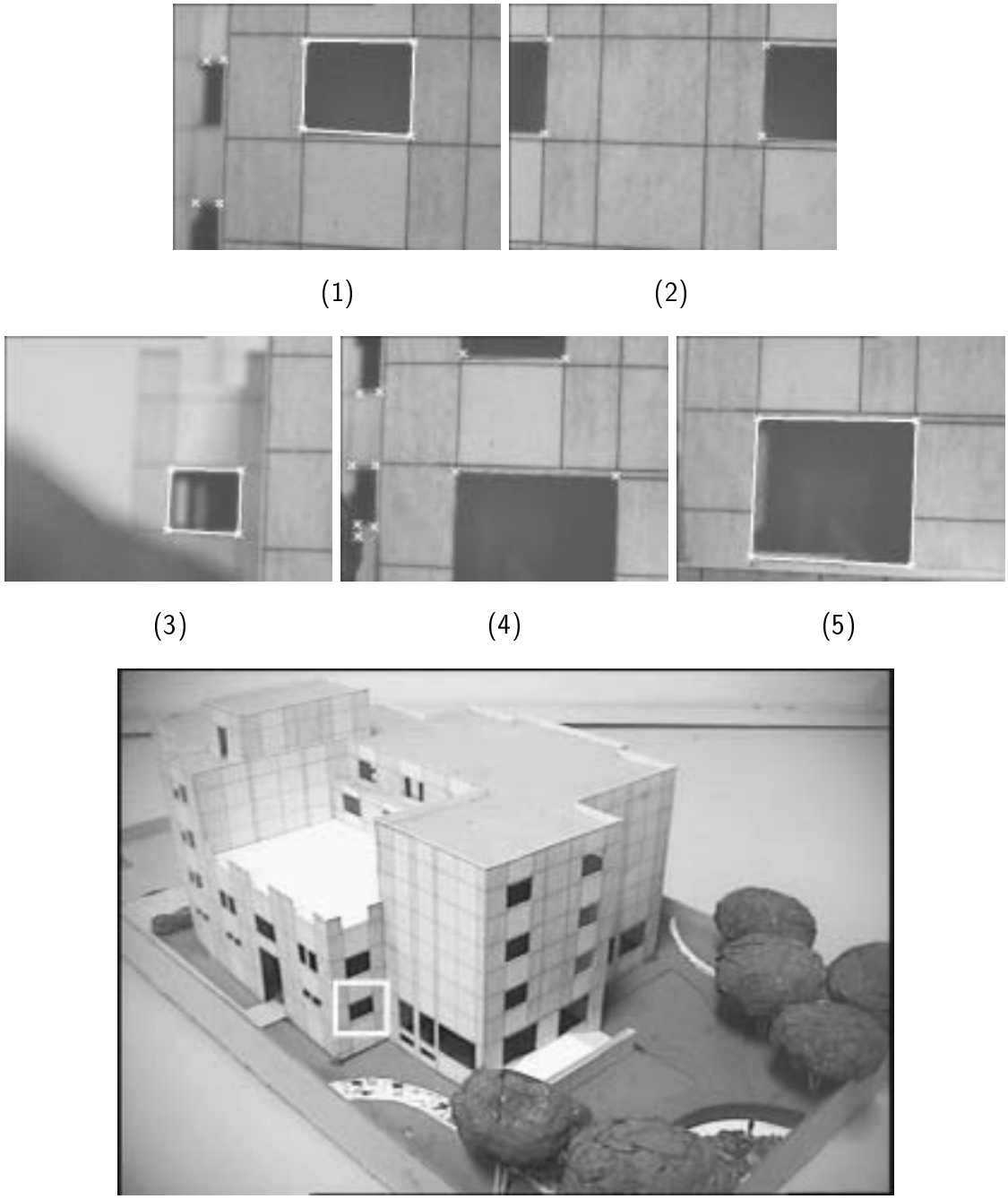


Figure 6.15: Experiment 9: The sequence of moves (in row major order, marked here with numbers (1) to (5)) required to identify the object and its pose. (3) shows a view cluttered by the presence of a tree. The image at the bottom shows an overall view. The trees are in the foreground. The corresponding window is highlighted with a white square.

correct recognition even in the case of clutter. The presence of a tree (an unmodeled object) accounts for clutter in Figure 6.15(3), and the Figures 6.16(1), (3) and (4). The large images at the bottom of Figures 6.15 and 6.16 show the corresponding overall views of the model. The trees are visible in the foreground. The corresponding windows are highlighted in white and black, respectively. This clutter does not affect the recognition performance of the system.

Experiment 11 (Figure 6.17) shows the use of 12 views for unambiguous recognition. The part pose estimation-based first level pruning results in a reduction of the number of hypotheses from 374 to 190. The size of the view interpretation hypothesis list at the last-but-one observation is 2. The last movement and corresponding last view enables the system to recognize the observed part uniquely as *DS_W_19*.

6.5 Conclusions

This chapter presents a new scheme for the recognition of an isolated 3-D object through on-line next view planning, when only a portion of it is visible to the camera. This recognition scheme is completely different from the aspect graph-based recognition scheme of Chapter 4, which assumes that the complete object fits into the camera's field of view. The prediction of a part rather than a view, imparts robustness to the system – the presence or absence (due to feature detection errors) of adjacent parts does not affect the recognition system. *An important consequence of our part-pased approach is the application of this system for a cluttered environment.* The system uses an uncalibrated camera, and uses inner camera invariants for pose recognition. Our knowledge representation scheme is used both for probabilistic hypothesis generation, as well as in planning the next view. Our experiments show the ability of the system to correctly identify objects and their pose even when there is a high degree of interpretation ambiguity associated with the initial view – using simple features, and suitably planned multiple views.

We have described various stages of this work in [66], [67], [69].

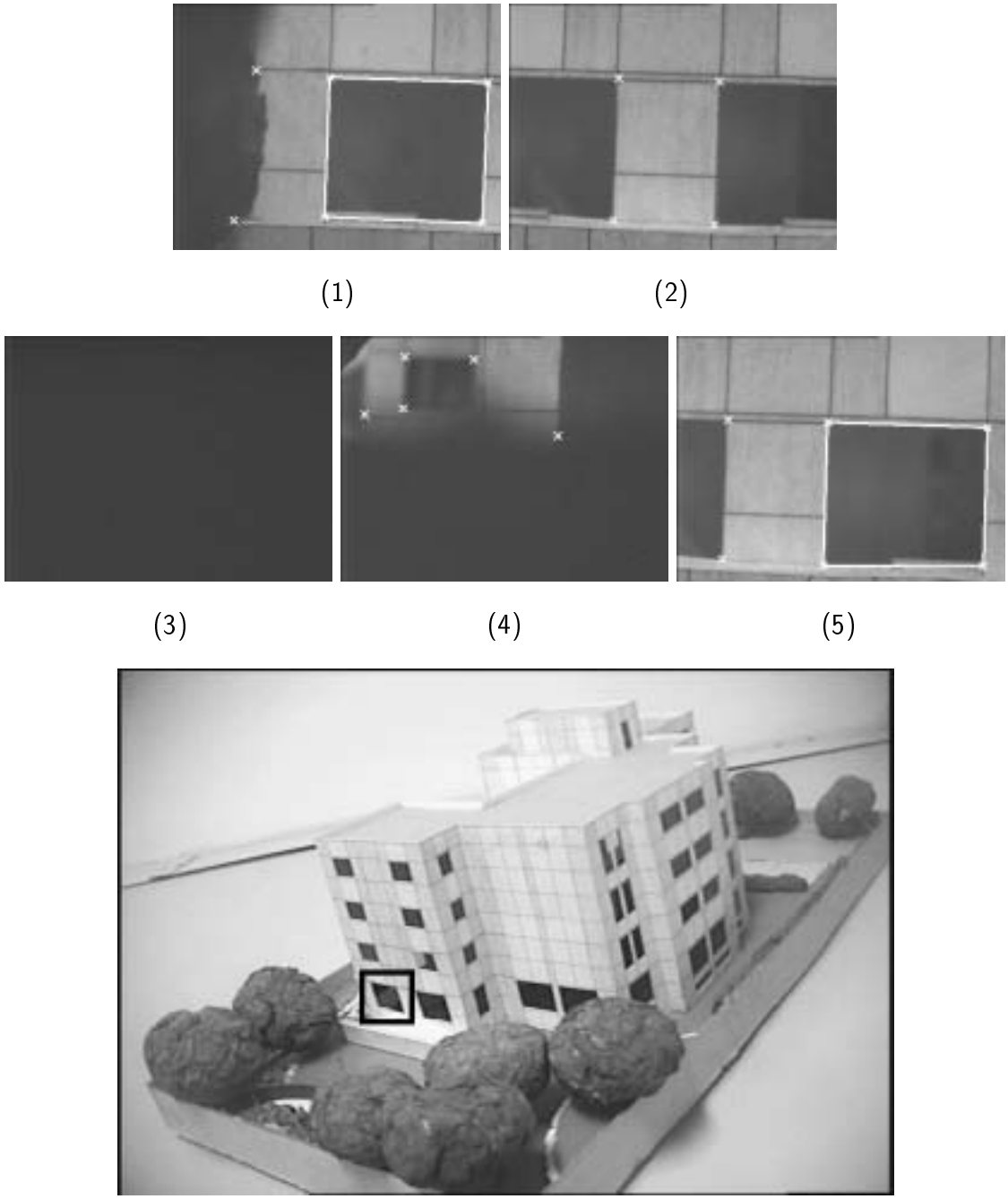


Figure 6.16: Experiment 10: The sequence of moves (in row major order, marked here with numbers (1) to (5)) required to identify the object and its pose. (1), (3) and (4) show views cluttered by the presence of a tree. The image at the bottom shows an overall view. The trees are in the foreground. The corresponding window is highlighted with a black square.

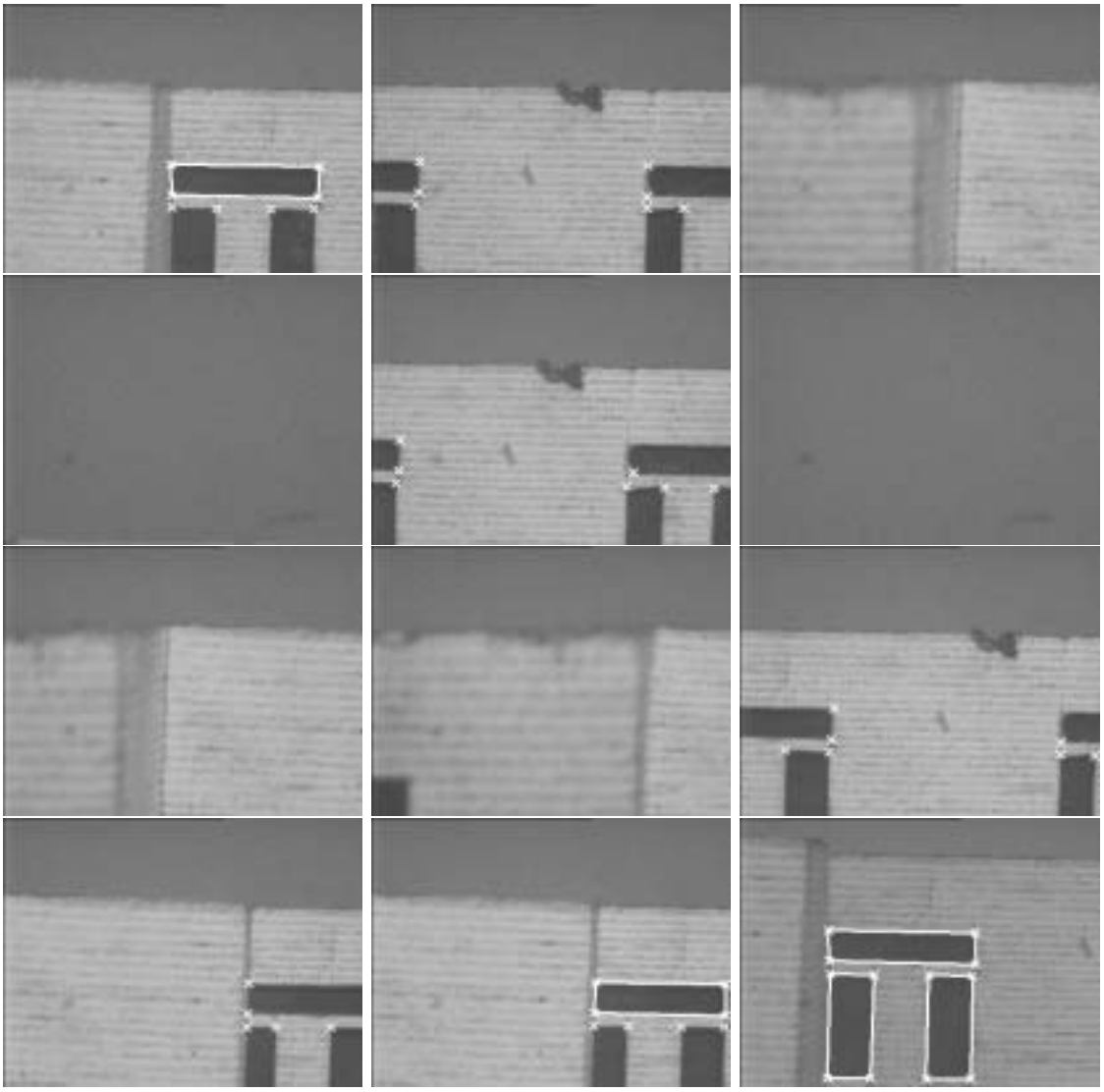


Figure 6.17: Experiment 11: The sequence of moves (in row major order) required to identify the object and its pose.

Chapter 7

Conclusions

In this thesis, we address the problem of recognizing an isolated 3-D object when a single view does not contain sufficient features to recognize it unambiguously. We propose two new on-line schemes for this purpose:

- Aspect Graph-based reactive object recognition using noisy feature detectors, and
- Part-based reactive object recognition, using inner camera invariants.

We assume the availability of an active sensor (camera). With such a sensor, the recognition problem involves taking a minimal sequence of views around the object, in order to recognize the object unambiguously.

For the first problem, we assume that the object fits into the camera's field of view. We consider the case of recognizing an object unambiguously, by planning a sequence of views around it. Our approach to this recognition problem is based on aspect graphs. The first step in our scheme is constructing AAGs for the objects in our model base. We tessellate the viewpoint space around an object, and collect feature data at each site. We deal with two cases – first, when the viewpoint space is a tessellated circle of constant radius with the object at its centre, and second, when the viewpoint space is a tessellated sphere. The feature detectors could be noisy. Hence,

the resultant aspect data could be corrupted with feature detection errors. First, we propose a classification of different types of errors in aspect graphs. We base our classification on experimental observations, and the statistical nature of occurrence of these errors. We propose a new AAG construction algorithm that converts noisy raw aspect data into an AAG. The algorithm maintains estimates of feature detection errors. We use these estimates in our object recognition strategy. We propose a new coefficient to evaluate the output of an AAG construction algorithm. We also examine the issue of the suitability of a feature detector for such a purpose.

The AAG construction algorithm uses the output of a set of noisy feature detectors. We use the same noisy feature detectors for our object recognition scheme. Here, we consider a single rotational degree of freedom between the object and the camera. We propose a new hierarchical knowledge representation scheme based on the relations between image-based features, aspects, classes, and the object models themselves. Our system uses the knowledge representation scheme in generating hypotheses corresponding to the interpretation of a view, as well as in planning the next view. Our probabilistic hypothesis generation mechanism has low-order polynomial time complexity. Based on the information from the current view, the algorithm computes the probability of the different objects in the model base. If the probability of no object is above a particular threshold, the algorithm plans the next view. This planning is subject to memory and processing limitations, if any. The planning process uses the current observation, as well as the past history for identifying a sequence of moves in order to distinguish between objects with similar views. An important feature of our formulation is that it is independent of any particular feature set. We show the results of numerous experiments where the system is able to uniquely identify fairly complex-shaped objects, which have a high degree of interpretation ambiguity associated with a view.

For the second recognition problem, we consider the case when an object may not fit in the camera's field of view. For the earlier problem, we had assumed an orthographic camera. Here, we consider the more general projective case. We further

relax the assumption of the first problem namely, that of the fixed distance between the camera and the object. Here, we consider the 6-degree of freedom case – we permit all rotations and translations about, and along all the three axes. The most important part of this work is the use of inner camera invariants. Neither do we assume a knowledge of the internal parameters of the camera, nor do we assume their constancy. We address cases where the internal parameters of the camera may be varied either unintentionally, or on purpose. We propose a method for Euclidean pose estimation, based on inner camera invariants.

For the recognition problem, we propose a hierarchical knowledge representation scheme based on an object and its parts. We adopt a probabilistic framework for generating hypotheses corresponding to the identity of a view. The current view may not result in the probability of any configuration of parts of a particular object, exceeding a particular threshold. The planning module plans the next view to uniquely identify the object and its pose, subject to possible memory and processing limitations. Here also, the planning process is reactive – it uses information from the current observation, as well as the past history. Our method is not affected by small movement errors, or some feature detection errors. We show recognition results corresponding to a number of experiments, with a reasonably complex domain of architectural models. We show results of successful recognition even in the presence of clutter.

7.1 Contributions

We summarize the main contributions of our work as follows:

- **Aspect Graph Construction with Noisy Feature Detectors**
  - A classification of errors in raw aspect data
  - A new algorithm to construct an aspect graph, given noisy raw aspect data
  - Formulation of a new coefficient to evaluate the performance of different aspect graph construction algorithms

  - An evaluation function for the suitability of a particular feature detector for the task

- **Aspect Graph-based Isolated 3-D Object Recognition through Next View Planning**

  - A hierarchical knowledge representation scheme to represent domain knowledge
  - A probabilistic hypothesis generation mechanism
  - The use of error information from the aspect graph construction phase for handling cases of feature detection errors
  - A reactive next view planning algorithm for the recognition of an isolated 3-D object
  - The use of the hierarchical knowledge representation scheme in probabilistic hypothesis generation, as well as the next view planning module

- **Inner Camera Invariants as a Tool for Next View Planning**

  - Inner Camera Invariants: a powerful tool for vision applications where the camera internal parameters may be varied, either unintentionally or on purpose
  - Formulation for the projective case, allows us to work independent of the knowledge of the value, or the constancy of the internal parameters of the camera
  - The use of Inner Camera Invariants for Pose Estimation and Euclidean Structure Estimation

- **Part-based Isolated 3-D Object Recognition through Next View Planning**

- Handling cases when the complete object does not fit inside the camera's field of view. Further, no assumption is made about either the knowledge of the internal camera parameters, or their constancy.
- A hierarchical knowledge representation scheme to represent domain knowledge
- The use of Inner Camera Invariants for pose estimation of a part, and at the hypothesis filtering stage in the course of arriving at a view interpretation
- A reactive next view planning algorithm for recognizing the given isolated 3-D object
- The use of the hierarchical knowledge representation scheme in probabilistic hypothesis generation, as well as the next view planning module
- Accounting for cases of feature detection errors and small movement errors

We support our proposed strategies with a large number of experiments. We show the results of over 100 experiments on two sets of models, illustrating how our aspect graph construction algorithm removes error regions. We show that same aspect graphs lead to faster and more reliable recognition, as compared to noisy raw aspect data. Over 100 experiments with two sets of models illustrate the efficacy of our algorithm in recognizing 3-D objects which have a large number of views in common. We show experiments of both pose and 3-D structure estimation, using inner camera invariants. A large number of experiments with reasonably complex architectural models show the advantage of using our part-based recognition strategy, when a complete view of the object does not fit in the (uncalibrated) camera's field of view. We also show results of successful recognition even in the presence of clutter. *An important contribution of the work is demonstrating the use of reactive next view planning in conjunction with suitable knowledge representation schemes, to recognize 3-D objects with fairly complex shapes, using simple features.*

7.2 Scope for Future Work

In this thesis, we examine two problems in the active recognition of an isolated 3-D object using suitably planned views. Some interesting extensions of the ideas proposed here, are as follows:

- **An explicit modeling of movement errors**

  Any active sensor is associated with odometric errors. Such a modeling will enhance the capabilities of an active recognition system.

- **Handling scenes with multiple objects**

  An interesting extension of this work is to the case of more than one object in a scene. This may require completely new representation and recognition schemes.

- **Handling Occlusions**

  An important subproblem of the above case is the case when one object occludes another. A single view may not suffice to get enough information about an occluded object. It is possible to suitably position an active sensor in order to get the required information about the objects which may be occluded from some particular viewpoints.

- **Object recognition in a cluttered environment**

  The previous point considers the case when all image information comes from known objects. However, all objects in a scene may not be modeled. We show some experimental results of the success of the second algorithm in the presence of clutter – a consequence of its robustness to some feature detection errors. These ideas can be examined further to explicitly account for clutter in the algorithm itself.

- **Map Building and Autonomous Robot Navigation**

  An interesting application of the ideas presented here, is to an autonomous

guided vehicle. An environment may contain modeled objects, as well as unmodeled objects. The first application area is constructing the map of an unknown environment. One may use our recognition strategies for modeled objects. For unmodeled objects, the use of inner camera invariants (which are based on image-based information alone) permits pose estimation, as well as 3-D Euclidean structure estimation. These two ideas may also be used by a robot navigating in a known environment – combining recognition with pose and 3-D structure estimation.

Bibliography

[1] Y. Aloimonos. Purposive and Qualitative Active Vision. In *Proc. International Conference on Pattern Recognition (ICPR)*, pages I:346 – 360, 1990.

[2] Y. Aloimonos. Purposive and Qualitative Active Vision. In Y. A. Feldman and A. Bruckstein, editors, *Artificial Intelligence and Computer Vision*, pages 455 – 464. Elsevier Science Publishers, 1991.

[3] Y. Aloimonos, I. Weiss, and A. Bandopadhyay. Active Vision. *International Journal of Computer Vision*, 1(4):333 – 356, 1987.

[4] M. Armstrong, A. Zisserman, and R. Hartley. Self-Calibration from Image Triplets. In *Proc. European Conference on Computer Vision (ECCV)*, pages 3 – 16, 1996.

[5] M. Asada. Map Building for a Mobile Robot from Sensory Data. *IEEE Transactions on Systems, Man and Cybernetics*, 37(6):1326 – 1336, November/December 1990.

[6] R. Bajcsy. Active Perception. *Proceedings of the IEEE*, 76(8):996 – 1005, August 1988.

[7] R. Bajcsy and M. Campos. Active and Exploratory Perception. *Computer Vision, Graphics and Image Processing*, 56(1):31 – 40, July 1985.

[8] D. H. Ballard. Animate Vision. *Artificial Intelligence*, 48(1):57 – 86, 1991.

[9] D. H. Ballard and C. M. Brown. *Computer Vision.* Prentice Hall, Englewood Cliffs, New Jersey 07632, 1982.

[10] D. H. Ballard and C. M. Brown. Principles of Animate Vision. *Computer Vision, Graphics and Image Processing: Image Understanding*, 56(1):3 – 21, July 1992.

[11] S. Baluja and D. Pomerleau. Dynamic Relevance: Vision-Based Focus of Attention using Artificial Neural Networks. *Artificial Intelligence*, 97(1-2):381 – 395, 1997. Special Issue on Relevance.

[12] R. Basri and E. Rivlin. Localization and Homing using Combinations of Model Views. *Artificial Intelligence*, 78(1-2):327 – 354, 1995.

[13] R. Bergevin and M. Levine. Extraction of Line Drawing Features for Object Recognition. *Pattern Recognition*, 25:319 – 334, 1992.

[14] R. Bergevin and M. Levine. Part Decomposition of Objects from Single View Line Drawings. *Computer Vision, Graphics and Image Processing: Image Understanding*, 55(1):73 – 83, January 1992.

[15] R. Bergevin and M. Levine. Generic Object Recognition: Building and Matching Coarse Descriptions from Line Drawings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(1):19 – 36, January 1993.

[16] P. J. Besl and R. C. Jain. Three-Dimensional Object Recognition. *ACM Computing Surveys*, 17(1):76 – 145, March 1985.

[17] I. Biederman. Human Image Understanding: Recent Research and a Theory. *Computer Vision, Graphics and Image Processing*, 32:29 – 73, 1985.

[18] A. Blake and A. Yuille, editors. *Active Vision.* The MIT Press, 1992.

[19] A. Blake and A. Zisserman. *Visual Reconstruction.* The MIT Press, 1987.

[20] G. Borgefors. Distance Transform in Arbitrary Directions. *Computer Vision, Graphics and Image Processing*, 27:321 – 345, 1984.

[21] H. Borotschnig, L. Paletta, M. Prantl, and A. Pinz. Active Object Recognition in Parametric Eigenspace. In *Proc. British Machine Vision Conference (BMVC)*, pages 629 – 638, 1998.

[22] K. Bowyer, D. Eggert, J. Stewman, and L. Stark. Developing the Aspect Graph Representation for Use in Image Understanding. In *Proc. Image Understanding Workshop*, pages 831 – 849, 1989.

[23] K. Bowyer, M. Sallam, D. Eggert, and J. Stewman. Computing the Generalized Aspect Graph for Objects with Moving Parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):605 – 610, June 1993.

[24] K. W. Bowyer (Organizer), O. D. Faugeras, J. Mundy, N. Ahuja, C. R. Dyer, A. P. Pentland, R. C. Jain, and K. Ikeuchi (Participants). Why Aspect Graphs Are Not (Yet) Practical for Computer Vision (Workshop Panel Report). *Computer Vision, Graphics and Image Processing: Image Understanding*, 55:212 – 218, March 1992.

[25] T. M. Breuel. *Geometric Aspects of Visual Object Recognition.* PhD thesis, MIT, 1992.

[26] T. M. Breuel. View-Based Recogntion. In *Proc. IAPR Workshop on Machine Vision Applications*, 1992.

[27] T. M. Breuel. View-Based Recogntion. Technical Report – Memo # 93-09, IDIAP, Switzerland, July 1993.

[28] R. A. Brooks. Intelligence without Representation. *Artificial Intelligence*, 47:139 – 159, January 1991.

[29] W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an Interactive Museum Tour-Guide Robot. *Artificial Intelligence*, 114(1-2):3 – 55, October 1999.

[30] J. B. Burns, R. S. Weiss, and E. M. Riseman. The Non-Existence of General-Case View-Invariants. In A. Zisserman and J. Mundy, editors, *Geometric Invariance in Computer Vision*. MIT Press, 1992.

[31] H. Buxton and S. Gong. Visual Surveillance in a Dynamic and Uncertain World. *Artificial Intelligence*, 78:431 – 459, 1995.

[32] J. Callahan and R. Weiss. A Model for Describing Surface Shape. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 240 – 245, 1985.

[33] F. G. Callari and F. P. Ferrie. Active Recognition: Using Uncertainty to Reduce Ambiguity. In *Proc. International Conference on Pattern Recognition (ICPR)*, pages 925 – 929, 1996.

[34] O. I. Camps. *PREMIO: The Use of Prediction in a CAD-Model-Based Vision System*. PhD thesis, Department of Electrical Engineering, University of Washington, 1992.

[35] O. I. Camps, C. Y. Huang, and T. Kanungo. Hierarchical Organization of Appearance-Based Parts and Relations for Object Recognition. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 685 – 691, 1998.

[36] O. I. Camps, L. G. Shapiro, and R. M. Haralick. PREMIO: The Use of Prediction in a CAD-Model-Based Vision System. Technical Report EE-ISL-89-01, Department of Electrical Engineering, University of Washington, 1989.

[37] O. I. Camps, L. G. Shapiro, and R. M. Haralick. PREMIO: An Overview. In *Proc. IEEE International Workshop on Directions in Automated CAD Based Vision*, pages 11 – 21, 1991.

[38] G. Castore and C. Crawford. From Solid Models to Robot Vision. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 90 – 92, 1988.

[39] I. Chakravarty. A Generalized Line and Junction Labeling Scheme with Applications to Scene Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI–1(2), April 1979.

[40] I. Chakravarty and H. Freeman. Characteristic Views as a Basis for Three Dimensional Object Recognition. In *Proc. SPIE Conference on Robot Vision*, volume 336, pages 37 – 45, 1982.

[41] C. H. Chen and A. C. Kak. A Robot Vision System for Recognition 3-D Objects in Low-Order Polynomial Time. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6):1135 – 1563, November/December 1989.

[42] G. Y. Chen and W. H. Tsai. An Incremental-Learning-by-Navigation Approach to Vision-Based Autonomous Land Vehicle Guidance in Indoor Environments Using Vertical Line Information and Multiweighted Generalized Hough Transform Technique. *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics*, 28(5):740 – 748, 1998.

[43] S. Chen and H. Freeman. On the Characteristic Views of Quadric Surfaced Solids. In *Proc. IEEE Workshop on Directions in Automated CAD-Based Vision*, pages 21 – 31, 1991.

[44] F. Chenavier and J. L. Crowley. Position Estimation for a Mobile Robot using Vision and Odometry. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages III:2588 – 2593, 1992.

[45] R. T. Chin and C. R. Dyer. Model Based Recognition in Robot Vision. *ACM Computing Surveys*, 18(1):67 – 108, March 1986.

[46] R. Choudhury. *Reconstruction Based Recogntion of 3D Objects using Invariants.* PhD thesis, Department of Mathematics, Indian Institute of Technology, Delhi, 1997.

[47] C. Colombo and J. L. Crowley. Uncalibrated Visual Tasks via Linear Interaction. In *Proc. European Conference on Computer Vision (ECCV)*, pages II:583 – 592, 1996.

[48] C. K. Cowan and P. D. Kovesi. Automatic Sensor Placement from Vision Task Requirements. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):407 – 416, May 1988.

[49] J. L. Crowley. ECVNet Tutorial on Active Computer Vision. `http://www-prima.imag.fr/ECVNet/Tutorial/av_tutorial.html`.

[50] J. L. Crowley. Navigation for an Intelligent Mobile Robot. *IEEE Journal of Robotics and Automation*, RA-1(1):31 – 41, March 1985.

[51] J. L. Crowley, J. M. Bedrune, M. Bekker, and M. Schneider. Integration and Control of Reactive Visual Processes. In *Proc. European Conference on Computer Vision (ECCV)*, pages II:47 – 58, 1994.

[52] J. L. Crowley, P. Bobet, and C. Schmid. Auto-Calibration by Direct Observation of Objects. *Image and Vision Computing*, 11:67 – 81, 1993.

[53] J. L. Crowley, P. Bobet, and C. Schmid. Dynamic Calibration of an Active Stereo Head. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 734 – 739, 1993.

[54] J. L. Crowley, P. Bobet, and C. Schmid. Maintaining Stereo Calibration by Tracking Image Points. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 483 – 488, 1993.

[55] A. P. Dempster. A Generalization of Bayesian Inference. *Journal of the Royal Statistical Society*, 30 (Series B):205 – 247, 1968.

[56] S. J. Dickinson, H. I. Christensen, J. Tsotsos, and G. Olofsson. Active Object Recognition Integrating Attention and View Point Control. In *Proc. European Conference on Computer Vision (ECCV)*, pages 3 – 14, 1994.

[57] S. J. Dickinson, H. I. Christensen, J. Tsotsos, and G. Olofsson. Active Object Recognition Integrating Attention and View Point Control. *Computer Vision and Image Understanding*, 67(3):239 – 260, September 1997.

[58] S. J. Dickinson, A. P. Pentland, and A. Rosenfield. Qualitative 3D Shape Reconstruction using Distributed Aspect Graph Matching. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 257 – 262, 1990.

[59] S. J. Dickinson, A. P. Pentland, and A. Rosenfield. From Volumes to Views: An Approach to 3D Object Recognition. *Computer Vision, Graphics and Image Processing: Image Understanding*, 55(2):198 – 211, March 1992.

[60] S. J. Dickinson, A. P. Pentland, and A. Rosenfield. Qualitative 3D Shape Reconstruction using Distributed Aspect Graph Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):174 – 198, February 1992.

[61] S. J. Dickinson, D. Wilkes, and J. K. Tsotsos. A Computational Model of View Degeneracy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):673 – 689, August 1999.

[62] S. Dutta Roy, S. Chaudhury, and S. Banerjee. 3D Object Recognition through Next View Planning. In P. P. Das and B. N. Chatterji, editors, *Pattern Recog-*

nition, Image Processing and Computer Vision: Recent Advances, pages 241 – 246. Narosa Publishing House, 1996.

[63] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Isolated 3-D Object Recognition through Next View Planning. In M. Vidyasagar, editor, *Intelligent Robotic Systems*, pages 494 – 501. Tata McGraw-Hill Publishing Company Limited, 1998.

[64] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Aspect Graph Construction with Noisy Feature Detectors. In S. Chaudhury and S. K. Nayar, editors, *Computer Vision, Graphics and Image Processing: Recent Advances*, pages 166 – 172. Viva Books Private Limited, 1999.

[65] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Isolated 3-D Object Recognition through Next View Planning. *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, 30(1):67 – 76, January 2000.

[66] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Part-based Isolated 3-D Object Recognition through Next View Planning using Inner Camera Invariants. In *Proc. Indian Conference on Computer Vision, Graphics and Image Processing*, pages 87 – 94, 2000.

[67] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Recognizing Large 3-D Objects through Next View Planning using an Uncalibrated Camera. *IEEE International Conference on Computer Vision (ICCV)*, 2001. (Communicated).

[68] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Aspect Graph Based Modeling and Recognition with an Active Sensor: A Robust Approach. *Proc. Indian National Science Academy, Part A*, (Accepted for Publication).

[69] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Recognizing Large Isolated 3-D Objects through Next View Planning using Inner Camera Invariants. *Computer Vision and Image Understanding*, (Communicated).

[70] S. Dutta Roy, S. Chaudhury, and S. Banerjee. Aspect Graph Construction with Noisy Feature Detectors. *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics*, (Revised version submitted).

[71] D. Eggert and K. Bowyer. Computing the Orthographic Projection Aspect Graph of Solids of Revolution. *Pattern Recognition Letters*, 11:751 – 763, 1990.

[72] D. Eggert and K. Bowyer. Computing the Perspective Projection Aspect Graph of Solids of Revolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(2):109 – 128, February 1993.

[73] D. W. Eggert, K. W. Bowyer, C. R. Dyer, H. I. Christensen, and D. B. Goldgof. The Scale Space Aspect Graph. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 1992.

[74] D. W. Eggert, K. W. Bowyer, C. R. Dyer, H. I. Christensen, and D. B. Goldgof. The Scale Space Aspect Graph. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1114 – 1130, November 1993.

[75] A. Elfes. Sonar-Based Real-World Mapping and Navigation. *IEEE Journal of Robotics and Automation*, RA-3(3):249 – 265, June 1987.

[76] F. Ennesser and G. Medioni. Finding Waldo, or Focus of Attention using Local Color Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):805 – 809, August 1995.

[77] W. Faig. Calibration of Close Range Photogrammetry Systems: Mathematical Formulation. *Photogrammetric Engineering and Remote Sensing*, 41(12):1479 – 1486, 1975.

[78] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. The MIT Press, 1996.

[79] O. D. Faugeras, N. Ayache, and B. Faverjon. Building Visual Maps by Combining Noisy Stereo Measurements. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages III:1433 – 1438, 1986.

[80] G. Fekete and L. S. Davis. Property Spheres: A New Representation for 3-D Object Recognition. In *Proc. IEEE Workshop on Computer Vision*, pages 192 – 201, 1984.

[81] C. Fennema, A. Hanson, E. Riseman, J. Ross Beveridge, and R. Kumar. Model-Directed Mobile Robot Navigation. *IEEE Transactions on Systems, Man and Cybernetics*, 20(6):1352 – 1369, November/December 1990.

[82] P. J. Flynn and A. K. Jain. BONSAI: 3-D Object Recognition Using Constrained Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1066 – 1075, October 1991. Special Issue on Interpretation of 3-D Scenes - Part I.

[83] P. J. Flynn and A. K. Jain. CAD-Based Computer Vision: From CAD Models to Relational Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(2):114 – 132, February 1991.

[84] B. V. Funt and G. D. Finlayson. Color Constant Color Indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:522 – 529, 1995.

[85] M. A. García, S. Velázquez, and A. D. Sappa. A Two-Stage Algorithm for Planning the Next View From Range Images. In *Proc. British Machine Vision Conference (BMVC)*, pages 720 – 729, 1998.

[86] J. A. Gaultieri, S. Baugher, and M. Werman. The Visual Potential: One Convex Polygon. *Computer Vision, Graphics and Image Processing*, 46(1):96 – 130, 1989.

[87] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI–6(6):721 –741, November 1984.

[88] Z. Gigus, J. Canny, and R. Seidel. Efficiently Computing and Representing Aspect Graphs of Polyhedral Objects. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 30 – 39, 1988.

[89] Z. Gigus and J. Malik. Computing the Aspect Graph for Line Drawings of Polyhedral Objects. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 654 – 661, 1988.

[90] Z. Gigus and J. Malik. Computing the Aspect Graph for Line Drawings of Polyhedral Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2):113 – 122, February 1990.

[91] C. Goad. Special Purpose Automatic Programming for 3D Model-Based Vision. In *Proc. DARPA Image Understanding Workshop*, pages 94 – 104, June 1983.

[92] K. Goldberg and M. Mason. Bayesian Grasping. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 1264 – 1269, 1990.

[93] K. D. Gremban and K. Ikeuchi. Appearance-Based Vision and the Automatic Generation of Object Recognition Programs. In A. K. Jain and P. J. Flynn, editors, *Three-Dimensional Object Recognition Systems*, pages 229 – 258. Elsevier-Science Publishers, 1993.

[94] K. D. Gremban and K. Ikeuchi. Planning Multiple Observations for Object Recognition. *International Journal of Computer Vision*, 12(2/3):137 – 172, April 1994. Special Issue on Active Vision II.

[95] W. E. L. Grimson. *Object Recognition by Computer*. The MIT Press, 1990.

[96] W. E. L. Grimson, A. Lakshmi Ratan, P. A. O'Donnel, and G. Klanderman. An Active Vision System to "Play Where's Waldo". In *Proc. DARPA Conference on Image Understanding*, 1994.

[97] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*, volume I. Addison-Wesley Publishing Company, Inc., 1992.

[98] R. I. Hartley. Self-Calibration from Multiple Views with a Rotating Camera. In *Proc. European Conference on Computer Vision (ECCV)*, pages I:470 – 478, 1994.

[99] G. Healey and D. Slater. Using Illumination Invariant Color Histogram Descriptors for Recognition. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 355 –360, 1994.

[100] J. Hertz, A. Krough, and R. G. Palmer. *Introduction To The Theory of Neural Computation*. Addison-Wesley Publishing Company, 1991.

[101] R. Horaud and G. Csurka. Self-Calibration and Euclidean Reconstruction Using Motion of a Stereo Rig. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 96 – 104, 1998.

[102] B. K. P. Horn. Extended Gaussian Image. *Proceedings of the IEEE*, 72(12):1671 – 1686, 1984.

[103] B. K. P. Horn. *Robot Vision*. The MIT Press and McGraw-Hill Book Company, 1986.

[104] C.-Y. Hsia and C.-L. Huang. Visual Events' Identification of Solids of Revolution from Perspective Views. *Pattern Recognition*, 26(2):333 – 349, February 1993.

[105] C. Y. Huang, O. I. Camps, and T. Kanungo. Object Recognition Using Appearance-Based Parts and Relations. In *Proc. IEEE International Confer-*

ence on Computer Vision and Pattern Recognition (CVPR), pages 877 – 883, 1997.

[106] S. A. Hutchinson and A. C. Kak. Planning Sensing Strategies in a Robot Work Cell with Multi-Sensor Capabilities. *IEEE Transactions on Robotics and Automation*, 5(6):765 – 783, December 1989.

[107] D. P. Huttenlocher and S. Ullman. Recognizing Solid Objects by Alignment within an Image. *International Journal of Computer Vision*, 5(2):195 – 212, November 1990.

[108] K. Ikeuchi. Determining Attitude of Object from Needle Map Using Extended Gaussian Image. Technical Report Memo No. 714, MIT Artificial Intelligence Laboratory, April 1983.

[109] K. Ikeuchi and T. Kanade. Automatic Generation of Object Recognition Programs. *Proceedings of the IEEE*, 76(8):1016 – 1035, August 1988.

[110] K. Ikeuchi and T. Kanade. Modeling Sensors: Towards Automatic Generation of Object Recognition Programs. *Computer Vision, Graphics and Image Processing*, 48:50 – 79, 1989.

[111] K. Ikeuchi and J. C. Robert. Modeling Sensor Detectability with the VANTAGE Geometric/Sensor Modeler. *IEEE Transactions on Robotics and Automation*, 7(6):771 – 784, December 1991.

[112] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data.* Prentice Hall, Englewood Cliffs, New Jersey 07632, 1988.

[113] A. K. Jain and R. L. Hoffman. Evidence-based Recognition of 3-D Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):783 – 802, November 1988.

[114] F. Jensen, H. Christensen, and J. Nielsen. Bayesian Methods for Interpretaiton and Control in Multiagent Vision Systems. In K. W. Bowyer, editor, *SPIE Applications of AI X: Machine Vision and Robotics*, volume 1708, pages 536 – 548, 1992.

[115] R. Kasturi and R. C. Jain. *Computer Vision: Principles*, chapter 5: Three-Dimensional Object Recognition. IEEE Computer Society Press Tutorial, 1991.

[116] G. Kinoshita, E. Mutoh, and K. Tanie. Haptic Aspect Graph Representation of 3-D Object Shapes. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 1648 – 1653, 1992.

[117] J. J. Koenderink and A. J. van Doorn. The Singularities of the Visual Mapping. *Biological Cybernetics*, 24:51 – 59, 1976.

[118] J. J. Koenderink and A. J. van Doorn. The Internal Representation of Solid Shape with Respect to Vision. *Biological Cybernetics*, 32:211 – 216, 1979.

[119] M. R. Korn and C. R. Dyer. 3-D Multiview Object Representations for Model-Based Object Recognition. *Pattern Recognition*, 20(1):91 – 103, 1987.

[120] A. Kosaka and A. C. Kak. Fast Vision-Guided Mobile Robot Navigaiton Using Model-Based Reasoning and Prediction of Uncertainties. *Computer Vision, Graphics and Image Processing: Image Understanding*, 56(3):271 – 329, November 1992.

[121] D. J. Kriegman and J. Ponce. Computing Exact Aspect Graphs of Curved Objects: Solids of Revolution. *International Journal of Computer Vision*, 5(2):119 – 135, November 1990.

[122] K. N. Kutulakos and C. R. Dyer. Recovering Shape by Purposive Viewpoint Adjustment. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16 – 22, 1992.

[123] K. N. Kutulakos and C. R. Dyer. Toward Global Surface Reconstruction by Purposive Viewpoint Adjustment. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 726 – 727, 1993.

[124] K. N. Kutulakos and C. R. Dyer. Global Surface Reconstruction by Purposive Control of Observer Motion. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 331 – 338, 1994.

[125] K. N. Kutulakos and C. R. Dyer. Occluding Contour Detection using Affine Invariants and by Purposive Viewpoint Control. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 323 – 330, 1994.

[126] K. N. Kutulakos and C. R. Dyer. Recovering Shape by Purposive Viewpoint Adjustment. *International Journal of Computer Vision*, 12(2/3):113 – 136, April 1994. Special Issue on Active Vision II.

[127] K. N. Kutulakos, C. R. Dyer, and V. J. Lumelsky. Provable Strategies for Vision-Guided Exploration in Three Dimensions. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 1365 – 1372, 1994.

[128] K. N. Kutulakos, V. J. Lumelsky, and C. R. Dyer. Vision-Guided Exploration: A Step Toward General Motion Planning in Three Dimensions. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 289 – 296, 1993.

[129] K. N. Kutulakos, W. B. Seales, and C. R. Dyer. Building Global Object Models by Purposive Viewpoint Control. In *Proc. CAD Based Vision Workshop*, pages 169 – 182, 1994.

[130] A. Laurentini. Comments on "Efficiently Computing and Representing Aspect Graphs of Polyhedral Objects". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(1):57 – 58, January 1996.

[131] X. Lebègue and J. K. Aggarwal. Extraction and Interpretation of Semantically Significant Line Segments for a Mobile Robot. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages I:1778 – 1785, 1992.

[132] X. Lebègue and J. K. Aggarwal. Generation of Architectural CAD Models Using a Mobile Robot. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages I:711 – 717, 1994.

[133] T. S. Levitt and D. T. Lawton. Qualitative Navigation for Mobile Robots. *Artificial Intelligence*, 44:305 – 360, 1990.

[134] C-H. Liu and W-H. Tsai. 3D Curved Object Recognition from Multiple 2D Camera Views. *Computer Vision, Graphics and Image Processing*, 50:177 – 187, 1990.

[135] H. Lu and L. G. Shapiro. A Relational Pyramid Approach to View Class Determination. In *Proc. International Conference on Pattern Recognition (ICPR)*, pages 379 – 381, 1988.

[136] H. Lu, L. G. Shapiro, and O. I. Camps. A Relational Pyramid Approach to View Class Determination. In *Proc. IEEE Workshop on Interpretation of 3D Scenes*, pages 177 – 183, 1989.

[137] E. Marchand and F. Chaumette. An Autonomous Active Vision System for Complete and Accurate 3D Scene Reconstruction. *International Journal of Computer Vision*, 32(3):171 – 194, 1999.

[138] S. O. Mason and A. Grun. Automatic Sensor Placement for Accurate Dimensional Inspection. *Computer Vision and Image Understanding*, 61:454 – 467, May 1995.

[139] N. A. Massios and R. B. Fisher. A Best Next View Selection Algorithm Incorporating a Quality Criterion. In *Proc. British Machine Vision Conference (BMVC)*, pages 780 – 789, 1998.

[140] M. J. Mataric. Integration of Representation Into Goal-Driven Behaviour-Based Robots. *IEEE Transactions on Robotics and Automation*, 8(3):304 – 312, June 1992.

[141] J. Maver and R. Bajcsy. Occlusions as a Guide for Planning the Next View. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):76 – 145, May 1993.

[142] S. Maybank and O. Faugeras. A Theory of Self-Calibration of a Moving Camera. *International Journal of Computer Vision*, 8(2):123 – 151, 1992.

[143] D. P. Mukherjee and D. Dutta Majumder. On Shape from Symmetry. *Proc. Indian National Science Academy*, 62, A(5):415 – 428, 1996.

[144] D. P. Mukherjee, A. Zisserman, and J. M. Brady. Shape from Symmetry: Detecting and Exploiting Symmetry in Affine Images. *Phil. Trans. R. Soc. London. A*, 351:77 – 106, 1995.

[145] O. Munkelt. Aspect-Trees: Generation and Implementation. *Computer Vision and Image Understanding*, 61(3):365 – 386, May 1995.

[146] H. Murase and S. K. Nayar. Visual Learning and Recognition of 3-D Objects from Appearance. *International Journal of Computer Vision*, 14:5 – 24, January 1995.

[147] D. W. Murray, P. F. McLauchlan, I. D. Reid, and P. M. Sharkey. Reactions to Peripheral Image Motion using a Head/Eye Platform. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 403 – 411, 1993.

[148] F. Nashashibi, M. Davy, and P. Fillatreau. Indoor Scene Terrain Modeling using Multiple Range Images for Autonomous Mobile Robots. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages I:40 – 46, 1992.

[149] S. K. Nayar and R. M. Bolle. Reflectance Based Object Recognition. *International Journal of Computer Vision*, 17(3):219 – 240, March 1996.

[150] R. E. Neapolitan. *Probabilistic Reasoning In Expert Systems: Theory and Algorithms*. John Wiley and Sons, Inc., 1990.

[151] A. Noble, D. Wilson, and J. Ponce. On Computing Aspect Graphs of Smooth Shapes from Volumetric Data. *Computer Vision and Image Understanding*, 66(2):179 – 192, May 1997.

[152] K. Onoguchi, M. Watanabe, Y. Okamoto, Y. Kuno, and H. Asada. A Visual Navigation System Using a Multi-Information Local Map. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages II:767 – 774, 1990.

[153] G. Oriolo, G. Ulivi, and M. Vendittelli. Real-Time Map Building and Navigaiton for Autonomous Robots in Unknown Environments. *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics*, 28(3):316 –333, June 1998.

[154] S. Pae and J. Ponce. Toward A Scale-Space Aspect Graph: Solids of Revolution. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages II: 196 – 201, 1999.

[155] D. Pagac, E. M. Nebot, and H. Durrant-Whyte. An Evidential Approach to Map-Building for Autonomous Vehicles. *IEEE Transactions on Robotics and Automation*, 14(4):623 –629, August 1998.

[156] J. Pearl. Fusion, Propagation and Structuring in Belief Networks. *Artificial Intelligence*, 29:241 – 288, 1986.

[157] C. G. Perrot and L. G. C. Hamey. Object Recognition, A Survey of the Literature. Technical Report 91-0065C, School of MPCE, Macquarie University, NSW 2109, Australia, January 1991.

[158] N. Pillow. *Recognition of Generalized Cylinders using Geometric Invariance.* PhD thesis, University of Oxford, 1996.

[159] N. Pillow, S. Utcke, and A. Zisserman. Viewpoint Invariant Representation of Generalized Cylinders uing the Symmetry Set. *Image and Vision Computing*, 13(5):355 – 365, June 1995.

[160] H. Plantinga and C. R. Dyer. An Algorithm for Constructing the Aspect Graph. In *Proc. 27th Symp. on F.O.C.S.*, pages 123 – 131, 1986.

[161] M. Pollefeys, R. Koch, and L. Van Gool. Self-Calibration and Metric Reconstruction in spite of Varying and Unknown Internal Camera Parameters. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 90 – 95, 1998.

[162] J. Ponce and D. J. Kriegman. Computing Exact Aspect Graphs of Curved Objects: Parametric Surfaces. In *Proc. 8th National Conference on Artificial Intelligence*, pages 340 – 350, 1987.

[163] J. Ponce and D. J. Kriegman. On Recognizing and Positioning Curved 3-D Objects from Image Contours. In *Proc. Image Understanding Workshop*, pages 461 – 470, 1989.

[164] J. Ponce, S. Petitjean, and D. J. Kriegman. Computing the Exact Aspect Graphs of Curved Objects: Algebraic Surfaces. In *Proc. European Conference on Computer Vision (ECCV)*, pages 599 – 614, 1992.

[165] A. R. Pope. Model-Based Object Recognition: A Survey of Recent Research. Technical Report 94-04, UBC Department of Computer Science, University of British Columbia, January 1994.

[166] J. H. Rieger. On the Classification of Views of Piecewise Smooth Objects. *Image and Vision Computing*, 5(2):91 – 97, 1987.

[167] J. H. Rieger. The Geometry of View Space of Opaque Objects Bounded by Smooth Surfaces. *Artificial Intelligence*, 44:1 – 40, 1990.

[168] J. H. Rieger. Global Bifurcation Sets and Stable Projections of Nonsingular Algebraic Sets. *International Journal of Computer Vision*, 7:171 – 194, 1992.

[169] J. H. Rieger. On the Complexity and Computation of View Graphs of Piecewise Smooth Algebraic Surfaces. *Phil. Trans. R. Soc. London. A*, 354:1899 – 1940, 1996.

[170] R. Rimey and C. Brown. Task-Oriented Vision with Multiple Bayes Nets. In A. Blake and A. Yuille, editors, *Active Vision*, pages 217 – 236. The MIT Press, 1992.

[171] R. D. Rimey and C. M. Brown. Where to Look Next Using a Bayes Net: Incorporating Geometric Relations. In *Proc. European Conference on Computer Vision (ECCV)*, pages 542 – 550, 1992.

[172] R. D. Rimey and C. M. Brown. Control of Selective Perception using Bayes Nets and Decision Theory. *International Journal of Computer Vision*, 12(2/3):173 – 207, April 1994. Special Issue on Active Vision II.

[173] D. R. Roberts and A. D. Marshall. Viewpoint Selection for Complete Surface Coverage of Three Dimensional Objects. In *Proc. British Machine Vision Conference (BMVC)*, pages 740 – 750, 1998.

[174] M. Robey, G. West, and S. Venkatesh. An Investigation into the Use of Physical Modeling for the Prediction of Various Feature Types Visible from Different Viewpoints. *Computer Vision and Image Understanding*, 61(3):417 –429, May 1995.

[175] C. A. Rothwell. *Recognition using Projective Invariance*. PhD thesis, University of Oxford, 1993.

[176] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Inc., 1995.

[177] K. Sato, K. Ikeuchi, and T. Kanade. Model Based Recognition of Specular Objects Using Sensor Models. *Computer Vision, Graphics and Image Processing: Image Understanding*, 55(2):119 – 129, March 1992.

[178] B. Schiele and J. L. Crowley. A Comparison of Position Estimation Techniques using Occupancy Grids. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages II:1628 – 1634, 1994.

[179] B. Schiele and J. L. Crowley. Object Recognition using Multidimensioanl Receptive Field Histograms. In *Proc. European Conference on Computer Vision (ECCV)*, pages I:610 – 619, 1996.

[180] B. Schiele and J. L. Crowley. Transinformation for Active Object Recognition. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 249 – 254, 1998.

[181] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, N. J., 1976.

[182] I. Shimshoni and J. Ponce. Finite-Resolution Aspect Graphs of Polyhedral Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):315 – 327, April 1997.

[183] C. C. Slama(Ed.). *Manual of Photogrammetry*. American Society of Photogrammetry and Remote Sensing, Falls Church, Virginia, USA, 1980.

[184] T. Sripradisvarakul and R. Jain. Generating Aspect Graphs for Curved Objects. In *Proc. IEEE Workshop on Interpretation of 3D Scenes*, pages 109 – 115, 1989.

[185] J. Stewman and K. Bowyer. Aspect Graphs for Planar-Face Convex Objects. In *Proc. IEEE Workshop on Computer Vision*, pages 123 – 130, 1987.

[186] J. Stewman and K. Bowyer. Creating the Perspective Projection Aspect Graph of Polyhedral Objects. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 494 – 500, 1988.

[187] J. H. Stewman and K. W. Bowyer. Direct Construction of the Perspective Projection Aspect Graph of Convex Polyhedra. *Computer Vision, Graphics and Image Processing*, 51(1):20 – 37, July 1990.

[188] M. J. Swain and D. H. Ballard. Color Indexing. *International Journal of Computer Vision*, 7(1):11 – 32, 1991.

[189] M. J. Swain and M. A. Stricker. Promising Directions in Active Vision. *International Journal of Computer Vision*, 11(2):109 – 126, October 1993.

[190] P. Swain and F. P. Ferrie. From Uncertainity to Visual Exploration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1038 – 1049, October 1991. Special Issue on Interpretation of 3-D Scenes - Part I.

[191] K. A. Tarabanis, P. K. Allen, and R. Y. Tsai. A Survey of Sensor Planning in Computer Vision. *IEEE Transactions on Robotics and Automation*, 11(1):86 – 104, February 1995.

[192] K. A. Tarabanis, R. Y. Tsai, and P. K. Allen. The MVP Sensor Planning System for Robotic Vision Tasks. *IEEE Transactions on Robotics and Automation*, 11(1):72 – 85, February 1995.

[193] G. H. Tarbox and S. N. Gottschlich. Planning for Complete Sensor Coverage in Inspection. *Computer Vision and Image Understanding*, 61(1):84 – 111, January 1995.

[194] S. Thrun. A Bayesian Approach to Landmark Discovery and Active Perception in Mobile Robot Navigation. Technical Report CMU-CS-96-122, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, May 1996.

[195] S. Thrun. Learning Metric Topological Maps for Indoor Mobile Robot Navigation. *Artificial Intelligence*, 99(1):21 – 71, February 1998.

[196] S. Thrun, W. Burgard, and D. Fox. A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots. *Machine Learning*, pages 55 – 85, 1998.

[197] R. Y. Tsai. A Versatile Camera Calibration Technique for High-Accuracy 3-D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses. *IEEE Journal of Robotics and Automation*, RA-3(4):323 – 344, August 1987.

[198] S. Ullman and R. Basri. Recognition by Linear Combination of Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:992 – 1006, 1991.

[199] L. Van Gool, T. Moons, D. Ungureanu, and E. Pauwels. Symmetry from Shape and Shape from Symmetry. *International Journal of Robotics Research*, 14(5):407 – 424, 1995.

[200] P. von Kaenel, C. M. Brown, and R. D. Rimey. Goal-Orientation Dynamic Vision. Technical Report # 466, University of Rochester Computer Science Department, August 1993.

[201] R. Wang and H. Freeman. Object Recognition based on Characteristic Views. In *Proc. International Conference on Pattern Recognition (ICPR)*, pages 8 – 12, 1990.

[202] N. A. Watts. Calculating the Principal Views of a Polyhedron. In *Proc. International Conference on Pattern Recognition (ICPR)*, pages 316 – 322, 1988.

[203] D. Weinshall and M. Werman. On View Likelihood and Stability. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):97 – 109, February 1997.

[204] M. Werman, S. Banerjee, S. Dutta Roy, and M. Qiu. Robot Localization Using Uncalibrated Camera Invariants. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages II: 353 – 359, 1999.

[205] M. Werman, M. Qiu, S. Banerjee, and S. Dutta Roy. Inner Camera Invariants and their Applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (Communicated).

[206] D. Wilkes, S. J. Dickinson, and J. K. Tsotsos. A Quantitative Analysis of View Degeneracy and its Use for Active Focal Length Control. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 938 – 944, 1995.

[207] L. E. Wixson. Exploiting World Structure to Efficiently Search for Objects. Technical Report # 434, University of Rochester Computer Science Department, July 1992.

[208] L. E. Wixson and D. H. Ballard. Using Intermediate Objects to Improve the Efficiency of Visual Search. *International Journal of Computer Vision*, 12(2/3):209 – 230, April 1994. Special Issue on Active Vision II.

[209] B. Yamauchi and R. Beer. Spatial Learning for Navigation in Dynamic Environments. *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics*, pages 496 – 505, June 1996. Special Issue on Learning Autonomous Robots.

[210] L. A. Zadeh. Fuzzy Sets. *Information and Control*, 8:338 – 353, 1965.

[211] L. A. Zadeh. Knowledge Representation in Fuzzy Logic. *IEEE Transactions on Knowledge and Data Engineering*, 1:89 –98, 1989.

[212] A. Zelinsky. A Mobile Robot Exploration Algorithm. *IEEE Transactions on Robotics and Automation*, 8(6):707 – 717, December 1992.

[213] A. Zisserman, D. Forsyth, J. Mundy, C. Rothwell, J. Liu, and N. Pillow. 3D Object Recognition using Invariance. *Artificial Intelligence*, 78:239 – 288, 1995.

Publications Related to the Thesis

Papers Published/Accepted for Publication

Journal Papers

1. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "Isolated 3D Object Recognition through Next View Planning"
   IEEE Transaction on Systems, Man and Cybernetics - Part A: Systems and Humans, vol. 30, no. 1, pp. 67 – 76, January 2000.

2. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "Aspect Graph Based Modeling and Recognition with an Active Sensor: A Robust Approach"
   Proc. Indian National Science Academy (INSA) Part A, Special Issue on Image Processing, Vision and Pattern Recognition
   (Accepted for Publication)

Refereed Conference Papers and Book Chapters

1. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "Part-based Isolated 3-D Object Recognition through Next View Planning using Inner Camera Invariants"
   Proc. Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP), pp. 87 – 94, 2000.

2. M. Werman, S. Banerjee, S. Dutta Roy and M. Qiu.
   "Robot Localization Using Uncalibrated Camera Invariants"
   Proc. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp. II: 353 – 359, 1999.

3. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "Aspect Graph Construction with Noisy Feature Detectors"
   In *Computer Vision, Graphics and Image Processing: Recent Advances*, S. Chaudhury and S. K. Nayar (Eds.), Viva Books Private Limited, pp. 166 – 172, 1999.

4. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "Isolated 3D Object Recognition through Next View Planning"
   In *Intelligent Robotic Systems*, M. Vidyasagar (Ed.), Tata McGraw-Hill Publishing Company Limited, New Delhi, pp. 494 – 501, 1998.

5. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "3D Object Recognition through Next View Planning"
   In *Pattern Recognition, Image Processing and Computer Vision: Recent Advances*, P. P. Das and B. N. Chatterji (Eds.), Narosa Publishing House, pp. 241 – 246, 1996.

Papers Communicated/In Review

Papers Communicated to Journals/In Review

1. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "Aspect Graph Construction with Noisy Feature Detectors"
   Revised version sent to the *IEEE Transactions on Systems, Man and Cybernetics, Part B*

2. M. Werman, M. Qiu, S. Banerjee, and S. Dutta Roy.
   "Inner Camera Invariants and their Applications"
   Communicated to the *IEEE Transactions on Pattern Analysis and Machine Intelligence*

3. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "Recognizing Large Isolated 3-D Objects through Next View Planning using Inner Camera Invariants"
   Communicated to the Journal *Computer Vision and Image Understanding*

Paper Communicated to Conference

1. S. Dutta Roy, S. Chaudhury, and S. Banerjee.
   "Recognizing Large 3-D Objects through Next View Planning using an Uncalibrated Camera"
   Communicated to the *IEEE International Conference on Computer Vision (ICCV)*, 2001.

Brief Bio-data

Sumantra Dutta Roy was born in New Delhi, India in 1971. He received his B. E. in Computer Engineering from the Delhi Institute of Technology in 1993. He completed his M. Tech in Computer Science and Engineering from the Indian Institute of Technology, Delhi in 1994. His interest in computer vision dates back to his B. E. final year project. His hobbies include painting and sketching, and playing the flute.