Object Recognition Using Color And Geometry Indexing

by

Lily Lee

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1995

Author .....
Department of Electrical Engineering and Computer Science
September 1, 1995

Certified by .....
W. Eric L. Grimson
Professor
Thesis Supervisor

Accepted by .....
Frederic R. Morgenthaler
Chairman, Departmental Committee on Graduate Students

Object Recognition Using
Color And Geometry Indexing

by

Lily Lee

Submitted to the Department of Electrical Engineering and Computer Science
on September 5, 1995, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science and Engineering

Abstract

This thesis addresses the problem in computer vision of recognizing a library of objects in a scene without knowing which objects are in the scene or where they are located. Inherent in this topic is the massive number of possible sets of features from both the scene and the known objects that need to be identified. For an object recognition system to function under these conditions, it must be able to quickly eliminate most of the features in the scene which are unlikely to be a part of any objects known to the system. The few features that remain will require much less time for the system to identify, thus making such a recognition system viable in its computing requirement. I approach the problem of eliminating most of the unlikely features with the use of the color in the scene and of the model objects. Any portion of the scene that contains a color that does not appears on any of the models is eliminated. A 3D object recognition theory developed by David Jacobs is then used on the remaining portions of the scene to recognize objects by using their shape features. The work in this thesis is at an exploratory stage. Nevertheless, I will show some preliminary results in which the use of color can greatly reduce the number of possibilities that a geometric system has to consider, thereby demonstrating that this approach is promising.

Thesis Supervisor: W. Eric L. Grimson
Title: Professor

Acknowledgments

I would like to thank my thesis advisor, Professor Eric Grimson, for his support and encouragement. He gave me the freedom to explore the topics in which I am most interested and provided his deep insights to guide me through this project.

I would also like to thank my undergraduate thesis advisor, Professor Ellen Hildreth of Wellesley College. Without her, I probably never would have gone on to research in computer vision. Her influence on my life extends beyond the academic realm.

There are a number of people in the MIT Artificial Intelligence Lab from whom I have benefited a great deal through numerous discussions. In particular, Tao Alter and Kah-kay Sung have helped me to clarify my ideas and to suggest alternatives. I would also like to thank Greg Klanderman and Sundar Narasimhan for providing good C libraries which saved me from having to write everything from scratch. I have spent two exciting years with the students in the lab. My officemates, Tina Kapur and Jose Robles, and our honorary officemate, Raquel Romano, have given me much moral support and have made the past two years fun. Tao Alter and Jose Robles have been most generous to read drafts of my thesis when they were both very busy.

Outside of the AI Lab, I would like to thank David Jacobs of NEC Research for promptly answering my questions on his thesis work which I have used in this thesis. My good friends Mark Smith and Nate Osgood have provided me with moral support and insights on life that I shall treasure for a long time to come. Mark has helped me in the research and the writing of this thesis from very early on. He has been a great sounding-board to bounce off ideas, and has given great suggestions in solving a number of problems that I have encountered while writing this thesis.

I am grateful to my parents for teaching me early on the value of knowledge and education. They have always expected me to perform my best and always believed that I could. I thank them for having such confidence in me.

Contents

Chapter 1

Introduction

1.1 The Problem of Object Recognition

Object recognition is the task of identifying which objects are present in a scene and the pose of each of the objects relative to the sensor. Model-based object recognition involves obtaining a description of the object(s) of interest and finding areas in an image which satisfy the stored ideal descriptions of the objects.

Object recognition has four sub-problems. The correspondence problem produces a correspondence between model features and image features when given the model. The localization problem produces the location of a given model in an image. The identification problem outputs the model given a subpart of an image. The general recognition problem involves identifying the models in a scene, their locations, and their poses. This thesis proposes an approach to solve the last problem, with a library of models.

The proposed system stores the descriptions of models in a library (see Figure 1-1). The recognition is done by selecting image features with the use of data-driven selection and searching the library for model objects consistent with the image data. Specifically, the system uses color information first to segment an image into regions, and second to determine whether a region is part of a known model object, and if so, to produce a hypothesis for which objects the region may have come. The system then uses a shape-based recognition scheme to test the hypothesis given by the color

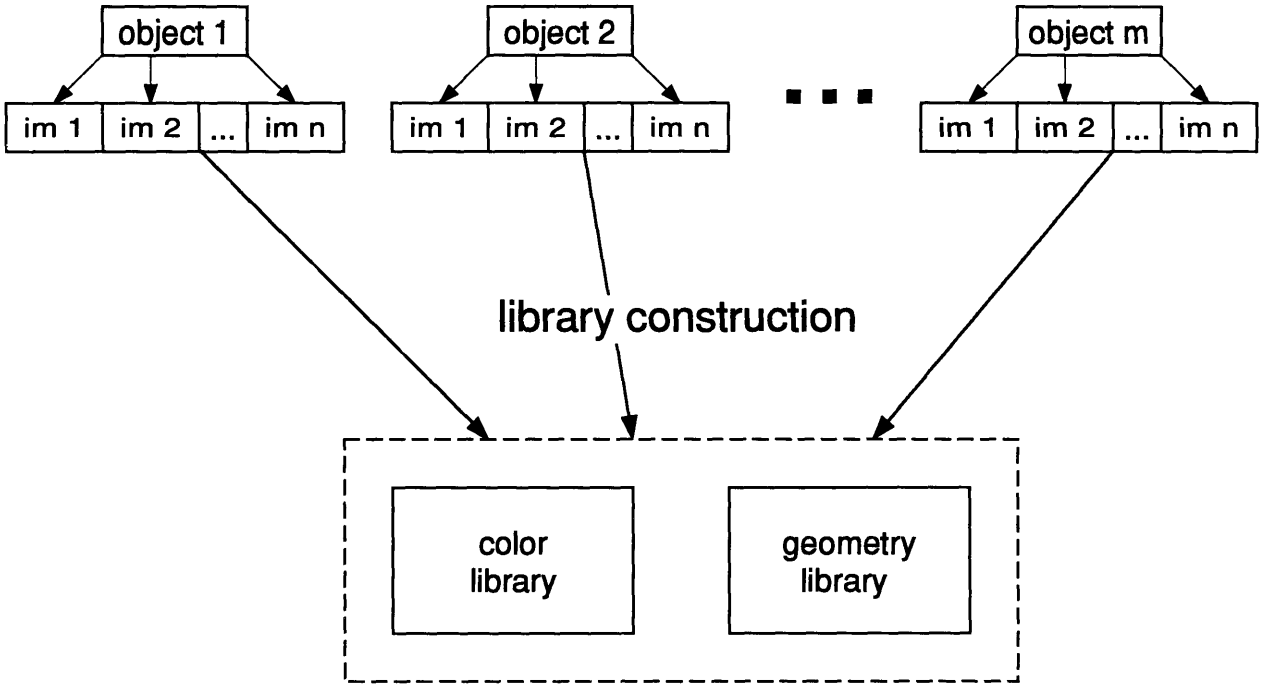


Figure 1-1: The model library.

information. Figure 1-2 illustrates the relationship between the color and geometry components of the recognition system. The advantages of using color information to aid the recognition system are that the use of color can steer the shape-based scheme to focus on regions of the image that might contain an object from the library, and that color can give a hypothesis of what the objects in those regions might be. As the reader shall see, reducing the work done by a shape-based recognition system is essential to the viability of a system capable of recognizing a library of objects.

In any realistic image of 3D scenes, the image data may contain effects of occlusions, noise, and spurious data. The complication caused by error in data on the complexity of an object recognition system is enormous. Grimson [9] showed for the constrained search on an interpretation tree method of object recognition [12, 13] that if all the data under consideration by the recognition system are from a single object, then the complexity of the search is quadratic in the size of the input. However, if there is spurious data in the set under consideration, then the complexity of the

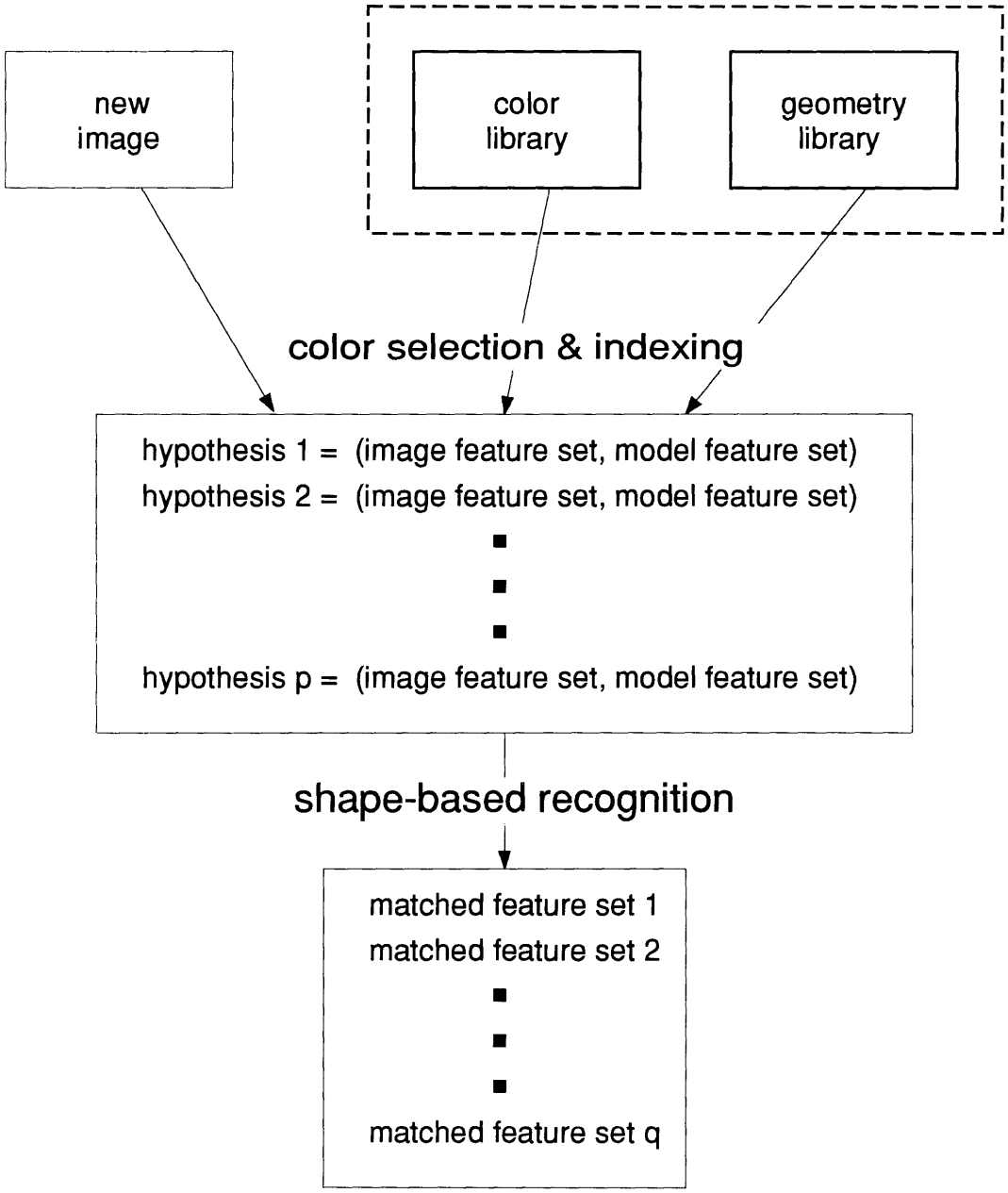


Figure 1-2: The recognition system.

system is exponential in the size of the set. The implication of Grimson's analysis is that a recognition system must be able to group features of the image data to form sets that are likely to come from individual objects.

The difficulties presented by recognition with a library of model objects and unknown scene locations is that the recognition system does not know which model objects are in a scene, nor does it know the approximate locations of the objects. Given that matching model and image data is expensive even when the data is from a single object and the model is known, a recognition system can not hope to succeed by searching through the entire library comparing each model to each group of features in the image. Thus a recognition system must be able to quickly produce a hypothesis of which model objects are in the scene and their locations, and then focus on those locations for further processing.

Given the above criteria for a recognition system, Grimson [10] breaks down the task of object recognition into three components:

- Selection—searching for subsets of the data that are believed to have come from a single object;
- Indexing—searching for the object from a library that is believed to have given rise to the data subset;
- Correspondence—searching for a match between the data subset from the selection process and a subset of the features of the model from the indexing process.

The object recognition system presented in this thesis attempts to use a number of cues, such as color and clusters of corner features, to guide the selection, indexing, and correspondence processes. In particular, the color segmentation of an image is used to select subparts of an image that may have come from a single object; the color of each segmented region can give a hypothesis of the model from which it came; finally, a geometric indexing scheme developed by Jacobs [18] can quickly produce a hypothesis of the correspondence between the image features and the model features.

1.2 Related Work

The literature on object recognition using multiple cues to guide the recognition process is sparse. However, there are a number of previous works on object recognition using individual cues, such as color, shape, texture, and on using multiple cues to retrieve images.

The traditional object recognition schemes have relied on shape information of the model object to identify and locate it. In the realm of recognition of 3D objects, there are three distinct approaches to the task:

- Sample the view sphere for discrete appearances of the model object and capture the 2D properties of clusters of features from images of the object. Recognition is achieved by matching the object in an image to its closest appearance in the pre-computed poses. Examples of this approach are [3, 19, 27]

- Store projective invariant 2D properties of the object, such as features on a planar surface or symmetrical features on an isotropic object. Recognition is achieved by searching the image for the same properties and matching them to model objects. An example of this approach is [8].

- Characterize all the appearances of the object. Recognition is achieved by searching the image for 2D structures that match an appearance of the model. Examples of this approach are [18, 28].

Since this thesis uses a geometric indexing system, we will give a summary of the above approaches and their possible use in geometric indexing.

Each of the schemes in [3, 19, 27] are initially proposed for use on 2D objects, then applied to 3D object recognition by sampling the view sphere and recognizing the object as an instance of its closest view. Breuel [3] presented an indexing system for 2D object recognition with a library of models which is invariant to scaling, translation, rotation, ordering of features, and occlusion. Scaling and translational invariance is achieved by using relative angles for spatial relationship rather than distances. Rotational invariance is achieved by fixing a canonical orientation, such as the major

axis of the object, and representing other features with respect to the angle of the canonical orientation. Invariance with respect to feature renumbering is achieved by computing a representation relative to each feature. Occlusion and spurious features are tolerated by matching subsets of the image with subsets of the model. A hash code is computed for all of the properties mentioned and stored in a table. During recognition, the same properties are computed from the image and converted to a hash code for look up in the pre-computed table.

Lamdan and Wolfson [19] present a 2D recognition system, geometric hashing, which achieves scaling, translational, rotational invariance with the use of an affine coordinate system. The system selects a set of three stable points to form the coordinates, and represents the remaining points with respect to those. Chapter 3 gives a more detailed explanation for the construction of the affine coordinate system. The geometric hashing system computes the same affine coordinates for each subset of model features and stores them in a table. During recognition, the system computes these quantities from an image and looks up the corresponding entry from the table and then enters the content of the entry in a histogram. The histogram bin with the highest vote is the most plausible model object. Both Lamdan-Wolfson and Breuel's work are extended to recognize 3D objects by sampling the view sphere for the appearances of the object and then storing each view as a separate entity in the lookup table. The systems then proceed to recognize the object by voting for the view that is closest to the appearance of the object in the image.

Thompson and Mundy's system [27] selects one vertex pair to form a spine, and computes the angle between the lines formed by two other vertex pairs and the spine. This pair of angles is invariant to in-plane rotation, but changes as a function of the two out-of-plane rotations. The system then quantizes the view sphere and represents for each view the angle-pair and the two out-of-plane rotations that produce the view. The recognition stage then votes for the two out-of-plane rotations that constitute the viewpoint from which the object is seen.

Each of the above three methods must approximate the 3D object recognition problem by a large number of 2D recognition problems. While this works when the

number of views considered when building the library is large enough, the amount of preprocessing is quite large, and the systems must tolerate for error beyond the image error and model error—they must account for the error produced by the quantized view sphere.

Forsyth and others [8] used 2D projective invariant properties for the recognition of objects. Under projective transformation, properties such as cross ratios of collinear points, parallel structures, and pairs of conics are invariant to view change and scale. The system recognizes objects in a scene by searching for such properties. The advantages of using projective invariants are (1) projective transformation is a realistic model of image formation, (2) the invariants are precisely described with no approximation needed, and (3) they account for more image properties than just lines and points—in particular, the conics. However, fitting conics to a curve is difficult. Since the invariants are all in 2D, this system cannot capture any property which describes the relationship of 3D model features.

Ullman and Basri [28] introduced a 3D object recognition system using a linear combination of views of the model object. Invariance with respect to translation, rotation, and scaling, are achieved by using an affine coordinate system. The method requires three views (in non-degenerate cases, it needs two views) and the correspondence of points for input. Given those, the system computes a rigid transformation matrix from the model to the given view. Recognition is done by applying the transformation on the model features and projecting them into the image space. The system then searches the image for features that are consistent with the model features that are projected into the image. The system assumes orthographic projection for the image formation process. This assumption simplifies the the descriptions of the images of the model. Though it is an approximation to the imaging process, the orthographic projection is very accurate if the object is far away from the sensor.

The method proposed by Ullman and Basri is not suited for recognition with a large library because for each hypothesized match between image features and model features, the recognition system must compute a rigid transformation from the model to the image. The process to compute the transformation is time consuming.

Jacobs [18] proposed a method related to Ullman and Basri's idea, but it does not require computing the transformation matrix. Details of Jacobs's method is discussed in Chapter 3. Jacobs used an affine coordinate system to describe the projection of all points into the affine plane, and a fourth point to describe the relative positions of points out of the plane. Given this setup, it can be shown that all views of the model object can be characterized as a pair of points on two equal-slope lines in affine subspaces. During recognition, the system computes a representation of the image in the affine subspaces and determines if the points fall on the correct lines. Jacobs proposed this scheme as an geometric indexing system. It is particularly suited for that purpose because the system can pre-compute and store all appearances of the objects. Recognition is simplified just to looking up a location in the affine subspace and determining if it is on any pre-computed lines. For these reasons, we have chosen Jacobs's system as the geometry component of the overall recognition system.

Researchers have suggested that color is a viable cue for object recognition. Swain and Ballard [25] used the color histogram of model objects as the only information for recognition. They computed the color histogram of a given scene, and then using the most prominent color bin in the histogram of the scene to hypothesize a set of models that also have the same color as their prominent colors. The color recognition system refines the hypothesis by matching bins that are less and less prominent. Swain and Ballard claimed that color histogram is relatively stable with respect to occlusion and change in pose of the model object. The system they designed is capable of recognizing a number of objects. A drawback of their system is that the object intended for recognition has to be segmented from the background before the system can be used. Thus it solves the identification problem.

Building upon Swain and Ballard's work, Ennesser and Medioni [6] employed color histograms to solve the problem of localization. The particular problem they worked on was to find Waldo in the "Where Is Waldo" books. A template of Waldo was given to the system to compute the color histogram. The system then computes a color histogram for each (overlapping) patch of a given image, and compares it to the model histogram. The center of the image patch receives a score based on how

closely its histogram resembles the model histogram. The patch containing Waldo, in theory, should be the centered around a point which has the highest match score, though the result is not always so.

A fundamental drawback in using color histograms in recognition is that it has no geometric information. Suppose that the pixels in an image are permuted. A color histogram matching scheme would produce the same result as with the original image, but the object in the image is certainly unrecognizable. Moreover, using color in general as the only source for recognition is inadequate because it does not give any pose information. Syeda-Mahmood [26] used color as a method to guide the vision system to parts of the image for further processing. The vision system segments the image by color, then depending on which mode, the pay-attention mode or the attract-attention mode, it will select the parts of image that contain the colors of the model object, or the parts that look the most conspicuous, respectively. The role that color plays in this system is one of grouping features and guiding the next step of recognition to the selected portions of the image. The recognition system described in this thesis will essentially use color in the same way—to guide the next step of the recognition. In addition, the color component in our system will also produce a hypothesis of the identity of the object in a region.

An example of a vision system that employs multiple cues is the QBIC project by Niblack *et al.* [23]. The image database system they designed is used to retrieve images from a library. The quantities computed for each image are its color histogram, texture, and shape. The system computes the color histograms in several color spaces, the coarseness and directionality of texture, and the high order moments of shape of user defined parts of an image. They demonstrated that the image retrieval system can produce a small superset of the images that the user is searching. Though this system is in the image database domain, and not object recognition, its use of multiple cues and their results is a good indication that the same approach could work well in the object recognition realm.

1.3 Overview

In the remainder of this thesis, we will present our approach to solving the object recognition with a library of models, the details of implementation, some experimental results and the future prospects of this approach. The remainder of this report is divided into the following chapters. Chapter 2 presents a high level view of how we approach the problem. Our approach depends on the use of color to guide the recognition system in getting the initial hypothesis and in grouping features for further processing. In addition, Our approach then applies an existing shape recognition system. With the construction of the color library and geometry library, search is reduced to table lookups. Chapter 3 will give the theoretical background for the shape recognition system developed by David Jacobs [18]. Chapter 4 gives a detailed view of how the subparts of the recognition system interact with each other to reduce the amount of search required for recognition. Chapter 5 gives the details of how the libraries and the recognition engine are implemented. Readers who are only interested in a high-level understanding of the system may skip this chapter. Chapter 6 presents some of the experimental results testing the recognition system. Chapter 7 presents the conclusions that are drawn from this expedition into recognition using both color and geometry, also discusses the problems that we have encountered in the process of developing this system, suggests possible solutions to some of them, and indicates future extensions to the overall recognition system.

Chapter 2

Object Recognition Using Color And Geometry Indexing

The objective of this thesis is to explore a possible solution to recognizing objects in a scene with a recognition system that uses a library of known models. The approach taken here is to use color information from the image and from the models as a coarse filter to select portions of the image for further processing with geometric information. In this chapter, we will justify our approach, state what we hope to accomplish through this investigation, and outline the limitations of this method.

2.1 The Approach

Shape information is crucial to the recognition of rigid objects.[1] Given a recognition system with a library of objects, a straightforward strategy for the system to recognize objects in a scene is to try to find each known object. However, Grimson [10] showed that while locating a correct object is quadratic in time, trying to locate an object not present in the scene is exponential in time. This analysis suggests that a recognition system must be able to group features that come from a single object, and that the system must be able to hypothesize what objects are in the scene. In other words,

[1] An exception to this statement is looking for a red cube in a world of green cubes. In this case, geometric information is useless, and the color cue is the only information needed.

the system must be able to perform selection and indexing.

Grimson points to two approaches to selection: (1) model-based selection, and (2) data-based selection. Model-based selection involves using the knowledge of the model to select parts of the image that the model could have come from. However, with this approach we are still stuck with having to test each model in turn, thus making the system run in time linear in the number of models. Grimson advocates using data-driven selection, thus making the selection process independent of the size of the model library.

There are several cues that can be used in grouping or selection of image features. The notables of these cues are stereo, texture, formations of geometric features, reflectance properties, and color. Stereo has the ability to group features that are at roughly the same depth, or that are on a surface without depth discontinuity. However, it lacks the ability to segment features that belong to multiple objects that lie at the same depth. Intuitively, parts of the scene with the same texture are likely to come from the same object. However, the existing texture segmentation methods can be quite time-consuming to compute [26, 20, 29]. Grouping features by their geometric relation is a very direct method of selection [17, 18, 26]. However, geometric grouping requires the researcher to invoke heuristic rules to determine the grouping. Reflectance properties of surfaces cannot be recovered from a single image. This is because the brightness seen in an image is the result of two factors: the lighting conditions, and the reflectance properties of the surface. Researchers have developed ways to use relative reflectance properties; in particular, reflectance ratios [22]. Reflectance ratios use gray levels only, and they can only be computed at the boundary between two regions. Color is the selection cue of choice in this thesis.

Color has been used for image segmentation[26]. Though color lacks the ability to discriminate between objects that may have the same color but different shapes, its selection ability is quite powerful. In general, we can assume that neighboring parts of an image that have the same color come from the same object. This is only violated if there are two neighboring regions from two different objects having the same color, which by our intuition we conclude does not occur very often. Color is relatively easy

to obtain compared to texture information. In digital images, color information is represented with the red, green, and blue (RGB) channels. Having multiple channels makes color more robust than any single channel representation. Using color also has the advantage that since the surface color of an object is fixed, knowing the color of a region in an image also gives the recognition system a hypothesis of which objects could be in the scene. Thus color serves as a cue for selection and as a cue for a partial indexing into the model library. Using a single color for selection is likely to over-segment the region that belongs to a single object, because most objects have multiple colors. Hence it is useful to use a number of color regions together to select parts of the image and to hypothesize the presence of a model object. Because different objects are less likely to have a combination of colors in common than to have one color in common, a hypothesis generated with the use of multiple color regions is more reliable than that generated using a single color region.

Many different representations of color information are possible. To choose one for the purpose of selection and indexing, the representation must meet certain criteria. The appearance of a color region to a digital camera depends on the reflectance properties of the surface and the lighting condition. Under white light, changes in the brightness of the light source affect the intensity component of the color appearance of the surface the most. Thus the color representation must be stable with respect to brightness change in lighting conditions. However, intensity information should still be retained for the purpose of distinguishing between a very dark and a very bright shade of the same color, for example, purple and lavender. Given these conditions, the possible choices of color representations are HSV [7], color opponent [2], and color ratio [21]. The color opponent model has three channels that are some linear combinations of RGB, thus any one channel is still affected by changes in lighting condition. The HSV representation factors the intensity information into one single channel and therefore is a viable choice. The color ratio representation is shown to be very stable with respect to changes in lighting condition [21]. Unfortunately, it removes intensity information altogether. It is possible to use the color ratio representation with an additional channel for the intensity information. However, for the sake of having a

uniform representation, the HSV representation is preferred.

The issue of color constancy always arises whenever color is used as a source for recognition. Color constancy refers to the stability of color under lighting condition changes. If a color surface is viewed under some colored light source, clearly the appearance of the color surface will differ from its appearance under white light. In the case of the HSV representation, the hue and saturation component will be shifted by the change in the color of the light source. However, it is questionable whether humans can actually perform color constancy in the low-level vision system.[2] Even when the viewer is aware of the colored light source, he/she still perceives the color as it appears, not as the true color of the surface. It is plausible that there may be some high level mechanism external to the vision system that takes into account the effect of non-white light source.

Given that the color component is capable of grouping features that are likely to come from a single object and can give a hypothesis of what objects are in the scene, a shape-based recognition scheme is needed. As indicated in Chapter 1, the recognition method proposed by Jacobs [18] is used in this thesis. This particular method is chosen for the reason that the decision of whether a cluster of image features is consistent with any model features can be done as a look up into a pre-computed table containing model feature clusters. A detailed explanation of the theoretical background of Jacobs's work will be presented in Chapter 3. The scheme involves using four non-coplanar model points to form an affine coordting all other model points with respect to that coordinate frame. This representation requires that a cluster of features must include at least five points, the first four of which must span 3D. However, it is possible to represent coplanar points with clusters containing a minimum of 4 points [19]. In this thesis, both representations will be used. Therefore, the system given here is capable of recognizing both 2D and 3D objects.

[2]This occurred to me as I had on a pair of orange-brown sunglasses—the sky looked light green, and there was no distinction between the bright yellow Ryder Rental sign and a white wall.

2.2 Objectives

We have presented evidence in favor of a recognition system which uses multiple cues to recognize a library of models. In particular, such a system is viable if it is capable of eliminating most of the regions of the image that do not contain any object that is known to the system, and the system can produce a hypothesis of possible models for the regions that need to be further investigated. The objective of this thesis is to build a recognition system to show that this approach to recognition with a library of models is feasible, and that color is a very important cue to limiting the amount of search that a shape-based recognition system has to do to make correspondence between model and image features and to improve the accuracy of the shape-based component.

The system built for this thesis uses color to group features that might come from a single object and to produce a hypothesis of which objects are in the scene. The geometry component of the system then takes the result of color selection and indexing to make lookup operations into a pre-computed geometry library. We will show through experiments that the color component is capable of guiding the geometry component to parts of the image that are likely to contain a known object, and it can also give the geometry component a hint of what the known object might be. In doing the selection, the color component gives the geometry component a set of features that is most likely from one object and contains very little spurious data, thus reducing the amount of search that the geometry component has to do. In doing the partial indexing, the color component produces a hypothesis of the object in a region of focus, so that the geometry component will only match those suggested models to the data in the region. This reduces the probability that the geometry component will match those features to model features that come from an unrelated model that might have some similar geometric configurations, hence improving the accuracy of the geometry component.

Since the goal of this thesis is to investigate the role of color in guiding the shape-based recognition system, we will implement a crude version of the scheme proposed

by Jacobs. In particular, only clusters containing five point features will be used, instead of larger clusters, which have greater discrimination power. Jacobs implemented the pre-computed library as a set of tables containing quantized representations of all possible appearances of the model features. Several tables were used, each containing clusters of features of different sizes. In this thesis work, only the analytical representations of the appearances of the model cluster will be used.

2.3 Scope

The cues used for selection, indexing and correspondence of our object recognition system are color and point features. Therefore, the types of objects that the system is capable of recognizing must have

- Rigidity
- Solid colored surfaces
- Object body has more than one color
- No specularity
- Well defined corners
- Straight edges, for obtaining corner features.

In addition, these model objects must be seen under roughly white light conditions.

Chapter 3

Theoretical Background

Object recognition involves identifying the projections of a model in a 3D world onto a 2D image. The model can undergo transformations such as rotations, scaling, and translations. The idea of an invariant function is very intriguing in this context. An invariant function is one which when applied to any transformation of a model, the result remains the same. Or,

$$f(t_1(m)) = f(t_2(m))$$

where m is the model, $t_1 \neq t_2$ are the transformations, and f is the invariant function. However, it has been shown that there is no such invariant function on 3D models [18, 5].

David Jacobs in his thesis [18] proposes a representation of 3D models that can characterize all appearances of the models using only 1D manifold0. He employs clusters of point features of the model object to produce an affine representation. The appearances of the model points with respect to the affine coordinates and a special point is characterized by a pair of lines in the affine subspaces. These lines not only represent all appearances of the models under rigid transformation plus uniform scaling, but also independent shear and scale in each axis. In short, any arbitrary nonsingular linear transformations are permitted. Thus they represent some images that a rigid model can never produce under scaled orthographic projections. Jacobs

suggested that humans are capable of recognizing non-planar views of pictures of a 3D scene which contain effects of non-rigid transformations, as shown in Figure 3-1. Thus it is possible that humans may employ a recognition strategy that involves affine transformations.

The remainder of this chapter will give an introduction to affine projection, part of the error analysis on affine matching given by Grimson, Huttenlocher, and Jacobs [11], the model-based invariant proposed by David Jacobs, and the geometric indexing and verification method Jacobs used. This chapter is adapted from Chapter 2 and 4 of [18].

3.1 Affine Projection

We introduce here the affine projection of 2D points. Unlike the 3D case where there is no invariant function, the affine projection in the 2D case is invariant to any affine distortion. This is the fact that Lamdan and Wolfson [19] employed in their recognition system using geometric hashing.

Let $\mathbf{p_1}$, $\mathbf{p_2}$, $\mathbf{p_3}$, be three non-collinear points in 2D space. These three points form an affine coordinate system with the origin, $\mathbf{o}$, and the two axes $\mathbf{u}$ and $\mathbf{v}$ defined by

$$\mathbf{o} = \mathbf{p_1}, \mathbf{u} = \mathbf{p_2} - \mathbf{p_1}, \mathbf{v} = \mathbf{p_3} - \mathbf{p_1}$$

Any point $\mathbf{p_i}$ in the 2D space can be described as a linear combination of $\mathbf{o}$, $\mathbf{u}$, and $\mathbf{v}$,

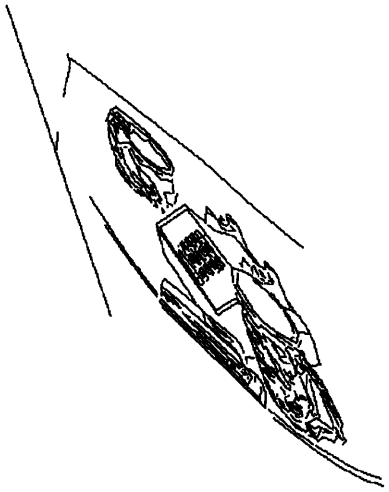
$$\mathbf{p_i} = \mathbf{o} + \alpha_i \mathbf{u} + \beta_i \mathbf{v}$$

and has the affine coordinates (α_i, β_i).

It can be shown that the affine coordinates of a point are invariant to any affine transformation on the 2D space. Let $\mathbf{q_i}$ be the projection of $\mathbf{p_i}$ after affine transformation, $\mathbf{A}$ a 2×2 non-degenerate matrix, and $\mathbf{t}$ a 2D vector. The basis triple after the affine transformation is

$$\mathbf{q_1} = \mathbf{A}\mathbf{p_1} + \mathbf{t}$$

(a)



(b)

Figure 3-1: Humans can recognize both an image of a 3D scene, as in (a), and image of a picture of a 3D scene, as in (b). Figures copied from [18] with permission of the author.

$$
\begin{aligned}
\mathbf{q_2} &= \mathbf{A p_2} + \mathbf{t} \\
\mathbf{q_3} &= \mathbf{A p_3} + \mathbf{t} \\
(\mathbf{o'}, \mathbf{u'}, \mathbf{v'}) &= (\mathbf{q_1}, \mathbf{q_2} - \mathbf{q_1}, \mathbf{q_3} - \mathbf{q_1})
\end{aligned}
$$

The coordinates of $\mathbf{q_i}$ after affine transformation are

$$
\begin{aligned}
\mathbf{q_i} &= \mathbf{A p_i} + \mathbf{t} \\
&= \mathbf{A}(\mathbf{o} + \alpha_i \mathbf{u} + \beta_i \mathbf{v}) + \mathbf{t} \\
&= \mathbf{A p_1} + \mathbf{t} + \alpha_i \mathbf{A}(\mathbf{p_2} - \mathbf{p_1}) + \beta_i \mathbf{A}(\mathbf{p_3} - \mathbf{p_1}) \\
&= \mathbf{o'} + \alpha_i \mathbf{u'} + \beta_i \mathbf{v'}
\end{aligned}
$$

Thus the affine coordinates of $\mathbf{p_i}$ remains the same under affine transformation.

Lamdan and Wolfson [19] used this fact in their recognition system. They used scaled orthographic projection as an approximation of the imaging process. The objects their system recognized are planar. Thus the affine coordinates of point features on the objects remain constant from all viewpoints.

Lamdan and Wolfson suggested three ways to recognize 3D objects. One could segment the image into surfaces and apply to each surface the planar object recognition system. One could construct an affine coordinate system in 3D with 4 non-coplanar points and represent all other points with respect to the 3D affine coordinate system. One could also sample the view sphere of the appearances of the 3D object and represent each view as would a planar object. They implemented the third method, and claimed that the second method is highly inefficient. However, Jacobs [18] demonstrated that the second method is a viable solution to 3D object recognition.

3.2 Error Analysis on Affine Matching

The previous section presented an invariant function for the recognition of planar objects. The formulation of the method assumes perfect data. However, in any real image, there is error. Grimson, Huttenlocher, and Jacobs [11] studied the effect of

bounded error on affine matching. They presented a rough upper bound and a precise error bound on the effects of noise on the affine coordinates of any point with respect to the basis triple. In this section, we will summarize the rough bound only, since that is what is used in this thesis.

Assuming that sensor error can be bounded by a vector, $\mathbf{e_i}$, with length less than ϵ_i, the error-free position of point features can be written as $\mathbf{q_i} = \mathbf{q_i'} + \mathbf{e_i}$, where $\mathbf{q_i'}$ is the measured position of the point in the image. Given that an error-free point $\mathbf{x}$ has affine coordinates (α, β) with respect to the error-free basis triple formed by $(\mathbf{q_1}, \mathbf{q_2}, \mathbf{q_3})$, the possible locations of the measured point $\mathbf{x'}$, with a bounded error of $\mathbf{e}$, is given by

$$\begin{aligned}
\mathbf{x} &= \mathbf{q_1} + \alpha(\mathbf{q_2} - \mathbf{q_1}) + \beta(\mathbf{q_3} - \mathbf{q_1}) \\
\mathbf{x'} &= \mathbf{q_1'} + \alpha(\mathbf{q_2'} - \mathbf{q_1'}) + \beta(\mathbf{q_3'} - \mathbf{q_1'}) + \mathbf{e} \\
&= (\mathbf{q_1} - \mathbf{e_1}) + \alpha((\mathbf{q_2} - \mathbf{e_2}) - (\mathbf{q_1} - \mathbf{e_1})) + \beta((\mathbf{q_3} - \mathbf{e_3}) - (\mathbf{q_1} - \mathbf{e_1})) + \mathbf{e} \\
&= \mathbf{x} - \mathbf{e_1} + \alpha(\mathbf{e_1} - \mathbf{e_2}) + \beta(\mathbf{e_1} - \mathbf{e_3}) + \mathbf{e}
\end{aligned}$$

Thus the uncertainty in the location of model point $\mathbf{x}$ is

$$-\mathbf{e_1} + \alpha(\mathbf{e_1} - \mathbf{e_2}) + \beta(\mathbf{e_1} - \mathbf{e_3}) + \mathbf{e}$$

or equivalently,

$$-(1 - \alpha - \beta)\mathbf{e_1} + \alpha\mathbf{e_2} + \beta\mathbf{e_3} - \mathbf{e}$$

Assuming that error in each point is independent and is bounded by ϵ, the uncertainty region of $\mathbf{x}$ is a disc of radius

$$\epsilon(|1 - \alpha - \beta| + |\alpha| + |\beta| + 1)$$

This error bound indicates that the uncertainty region of any point with respect to the affine basis triple depends only on the affine coordinates of that point and the sensor error, irrespective of the location of the basis triple and thus the viewpoint.

Grimson *et al.* go on to present a more precise error bound where the viewpoint is taken into consideration. The formulation of the precise error bound indicates that it would require excessive computing time. Therefore, it will not be used in this thesis. The interested reader may refer to their article for the derivation of the precise error bound.

3.3 Affine Representation of 3D Objects

Jacobs [18] presents a model-based invariant for the recognition of 3D objects. His method involves using 3 points of the model to form an affine projection plane, and a fourth point out of the plane to represent the relative distance of all other points to the affine plane with respect to the distance of the fourth point. Under scaled orthographic projection and affine distortion (which describes the orthographic projection of an image of a 3D scene), the affine representation used by Jacobs can represent all possible appearances of a 3D scene and images of a 3D scene.

To describe Jacobs's method in greater detail, let m be the model, $\mathbf{V}$ be the scaled orthographic transformation from 3D to 2D, and $\mathbf{A}$ be the 2D affine transform. Any image of the model can be represented by

$$\mathbf{A}(\mathbf{V}(m)) = i$$

where $\mathbf{V}$ accounts for the appearance of a 3D scene, and $\mathbf{A}$ accounts for the appearance of a picture of a 3D scene. Jacobs's method seeks to compactly represent the set of images of produced by the model and pictures of the model, the i's.

Let $\mathbf{p_1}$, $\mathbf{p_2}$, $\mathbf{p_3}$, and $\mathbf{p_4}$ be four non-coplanar points of the 3D model, and let $\mathbf{p_j}$, be any other point on the model object. Figure 3-2 shows a view of these points. The points $\mathbf{p_1}$, $\mathbf{p_2}$, and $\mathbf{p_3}$ form the basis triple of an affine coordinate system, $(\mathbf{o}, \mathbf{u}, \mathbf{v})$, as defined in Section 3.1. Call the plane formed by these three points the model plane. Let $\mathbf{p'_4}$ be the perpendicular projection of $\mathbf{p_4}$ onto the model plane, having affine coordinates (a_4, b_4); and let $\mathbf{p'_j}$ and (a_j, b_j) be similarly defined. Viewed at a

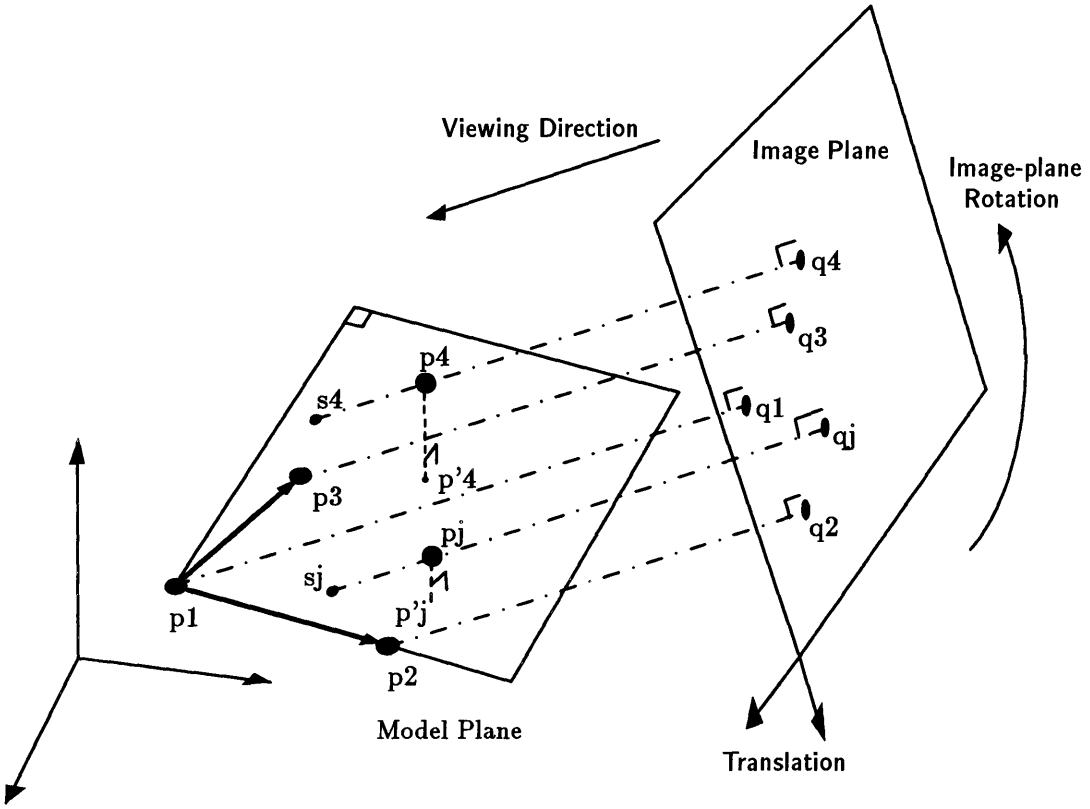


Figure 3-2: Construction of the affine representation of 3D models. Figure copied from [18] with permission of the author.

direction not perpendicular to the affine model plane, the model points $\mathbf{p_i}$'s produce image points $\mathbf{q_i}$'s under orthographic projection. Let $\mathbf{s_4}$ be the intersection of the model plane and the line through $\mathbf{p_4}$, $\mathbf{q_4}$, having affine coordinates (α_4, β_4); and let $\mathbf{s_j}$ and (α_j, β_j) be similarly defined. The triangles $\mathbf{p_4 p'_4 s_4}$ and $\mathbf{p_j p'_j s_j}$ are similar, because the angles are the same. Let r_i be the distance from $\mathbf{p_i}$ to the model plane, or $r_i = ||\mathbf{p_i} - \mathbf{s_i}||$, then the following equality results from the similarity of the triangles,

$$\frac{(\alpha_j, \beta_j) - (a_j, b_j)}{r_j} = \frac{(\alpha_4, \beta_4) - (a_4, b_4)}{r_4}$$

or

$$(\alpha_j, \beta_j) = r_j \frac{(\alpha_4, \beta_4) - (a_4, b_4)}{r_4} + (a_j, b_j)$$

Since the quantities (a_i, b_i) and r_i do not change with viewpoint and are constant, (α_j, β_j) are linear functions of (α_4, β_4). Since the α and the β dimensions are orthogonal, they can be separated into two subspaces containing two lines having the same slope,

$$\alpha_j = \frac{r_j}{r_4}\alpha_4 - \frac{a_4 r_j}{r_4} + a_j$$

$$\beta_j = \frac{r_j}{r_4}\beta_4 - \frac{b_4 r_j}{r_4} + b_j$$

Figure 3-3 shows the affine representation for a 3D model with 5 points. For a model with n points, its affine representation is a line in $n - 3$ dimensions in the α and β subspaces.

Given this affine representation of a 3D model and a particular ordering of the points of the model, a new image is consistent with the model if an ordering of the image points produces affine coordinates that lie on both α and β lines of the model. This method lends itself very well to a recognition system that pre-computes a library of such representations, and then uses the library during recognition for the purposes of checking the consistency between the image points and the model(s) in the library. Assuming that all points on the model object are equally distinctive, a set containing a larger number of points will be better at discriminating between model and non-model points than a set that contains fewer points. The difficulty of using

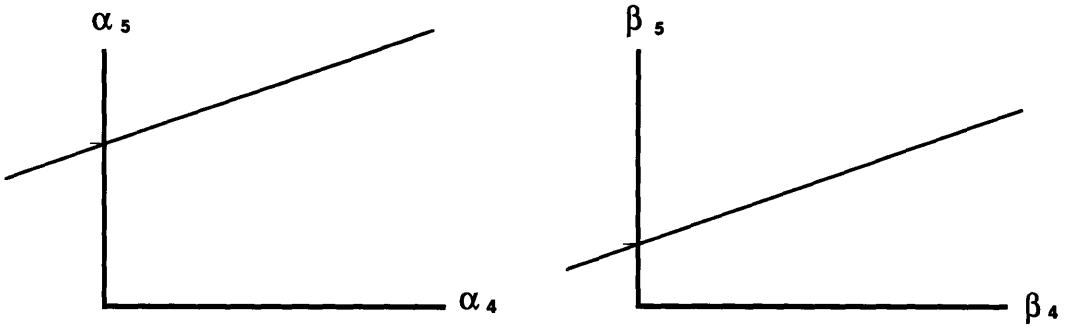


Figure 3-3: The representation of a five-point model in the α and β subspaces. The slope of the lines is $\frac{r_5}{r_4}$. The intercept in the α subspace is $-\frac{a_4 r_5}{r_4} + a_5$, and in the β subspace is $-\frac{b_4 r_5}{r_4} + b_5$.

this scheme for recognition arises from the fact that the model points and the image points must have the same ordering. A straightforward method to overcome this difficulty is to either represent all possible orderings of the model points, or to check all possible orderings of image points during recognition. However, the number of possible permutations of a given set is quite large if the set contains a large number of points. Thus there is a trade-off between the size of the library and the discrimination ability of the recognition system. To account for occlusion of feature points as well as the fact that not the same number of model points may appear at any instance, sets containing different numbers of feature points must be considered. The number of possible representations of a model becomes even larger. Jacobs proposes a grouping system to obtain sets of points that could have come from the same object, and to constrain the number of permutations of the points. His grouping method relies on the relative position of geometric features, such as sets of lines that form a convex polygon. The goal of this thesis is to investigate the role of color in grouping features and in selecting possible models from a library. Thus we will omit a summary of Jacobs's grouping system. The interested reader may refer to Chapter 6 of [18].

3.4 A Geometric Indexing Scheme

Using the affine representation for 3D models that he proposed, Jacobs introduced a recognition system. The system pre-computes the lines that represent all appearances of a model object and stores those lines in two subspaces, the α and β subspaces. It is possible to represent these lines analytically, and the recognition system can compute the distance from an affine representation of an image feature set to all the lines in the library. However, this process takes time linear in the number of models in the library. To facilitate fast recognition, the system must be able to compute the point representation for the image features and uses that as the index to look up entries in the subspaces. Thus the subspaces are quantized. To practically represent the subspace in digital form, it should also be bounded (note that the affine subspaces are not bounded).

The three issues involved in quantizing the subspaces are how to bound them, how fine should the quantization be, and what kind of quantization (such as linear, log, or exponential) it should be. Jacobs chose the bound of the subspaces to be ± 25. The precise error bounds given in [11] indicate that the amount of error in image position of feature points increase as their affine coordinates increase. In other words, beyond some threshold, the size of error in the affine space will be too large for accurate matching and should not be used. Thus the affine subspaces is bounded in practice. By the same token, the quantization of the subspaces should also be affected by the error analysis. Jacobs quantized the subspaces so that the bins are larger toward the edge of the bounded subspace than in the center of the subspace. It was experimentally determined that when the subspaces are quantized at 400×400, the performance of the system is almost as good as it is with analytical representations.

The sets of features are entered into all entries of the geometric indexing table which intersect the line representation of the set. The sets of features are at least five in number. The first four points of the ordered set are non-coplanar. Intuitively, one would want to use as large a set of feature points as possible, because the larger sets possess greater discrimination power. However, there are trade-offs between the

discrimination ability of the feature clusters and the amount of space required to store them, and the larger sets are more prone to be affected by occlusion. Jacobs used sets with between 5 to 7 features. He used a grouping system to constrain the number of possible orderings of these features in the set. Each of these ordered sets produces a pair of lines in the affine subspaces. These subsets are entered into each bin of the quantized geometric indexing table that the lines intersect. Errors in image and model data can be accounted for in two ways, by either building the error tolerance into the indexing table, or by searching for some neighborhood during lookup stage. Jacobs used the precise error bound during recognition. He argued for accounting for error during recognition, because the space requirement of storing entries within error tolerance is too large, and because the precise error bound depends on data that is only known from the image data.

3.5 Verification

Because the geometric indexing system can only feasibly represent a subset of the features from a model, a "hit" in the indexing table does not conclusively indicate the presence of a model. Jacobs proposed a verification scheme that is similar to Ullman and Basri's recognition system [28]. Once some of the feature points are identified, the system attempts to project all model points into the image space and looks for image features that match the model features. Ullman and Basri's method requires computing the transformation from the model to the image to project the model points. Jacobs's method calls for all features of the model represented as one cluster of feature in affine space, thus resulting in a line in high dimensional space. The points whose identities are known can be used to determine the position of the point on the high dimensional line containing all features of the model. Given this point, the affine coordinates of all features on the model are known. It is straightforward to convert the affine coordinates into image coordinates and then to perform verification by searching for points in the image consistent with the projection of the model.

Chapter 4

The Recognition System

4.1 Introduction

The recognition system presented here is to have the capability of recognizing a library of model objects. The shape information of an object is in general crucial to the ability of the system to recognize it. However, shape-based recognition is very time consuming if the data set considered by the shape-based system contains spurious features or is the wrong set altogether. In an effort to alleviate this problem, a color component is added to select sets of data from the image that are likely to have come from one object and to give the shape-based system a hint of what the object might be.

Our recognition system functions in two stages. The first stage is the library construction stage, during which the system uses a collection of images of the models to obtain color and geometry information about the model and stores the information into the respective libraries. The color information collected consists of pairs of adjacent color regions on the model, along with the range of color in which each region may appear. The shape information collected are sets of stable point features, namely corners, and the color regions surrounding them. They are stored in the libraries in such a way that given a pair of colors, a look-up in the color library will reveal whether there is a model with these two colors, and if so, which sets of features of the model could be in those color regions.

The second stage is the recognition stage, during which the recognition engine of the system takes a new image, obtains the color and geometry information from it, and looks up the appropriate entries of the libraries to produce a list of objects that are in the image, their image locations, and their correspondence with the model objects. The recognition engine takes every pair of adjacent color regions and looks up the pair of colors in the color library. If there is a model which has this pair of color regions, then the recognition engine goes on to extract features from the pair of regions and checks for consistency between the image features and the model features using the model-based invariant developed by Jacobs [18]. In the remainder of this chapter, we will give a high level view of how the libraries are constructed and how they are used by the recognition engine. Details of the implementations are left to Chapter 5.

4.2 Construction of the Libraries

The color and geometry recognition system pre-computes a model library for fast look up by the recognition engine. In particular, two libraries, the color and geometry libraries, each containing the respective model information is stored by the system. The recognition engine can perform look up in the color library, using a pair of colors from two adjacent image regions. The entries in the color library contain a list of hypothesized models and feature sets that could have come from a pair of adjacent model regions with the specified colors[1]. The result of the color library look up, together with a set of image features pertaining to the pair of regions, are fed into the geometry library to search for subsets of the hypothesized matches with which the image features are consistent.

The model library describes colored objects with sharply defined features. Each object is entered into the color and geometry libraries through a process which is

[1]The distinction between image region and model region is that an image region refers to a portion of the image which has roughly uniform color, and a model region refers to contiguous surfaces of a model with the same color and which may lie in 3D. In other words, the entire model region is not necessarily visible at any one instance, though parts of it may be visible.

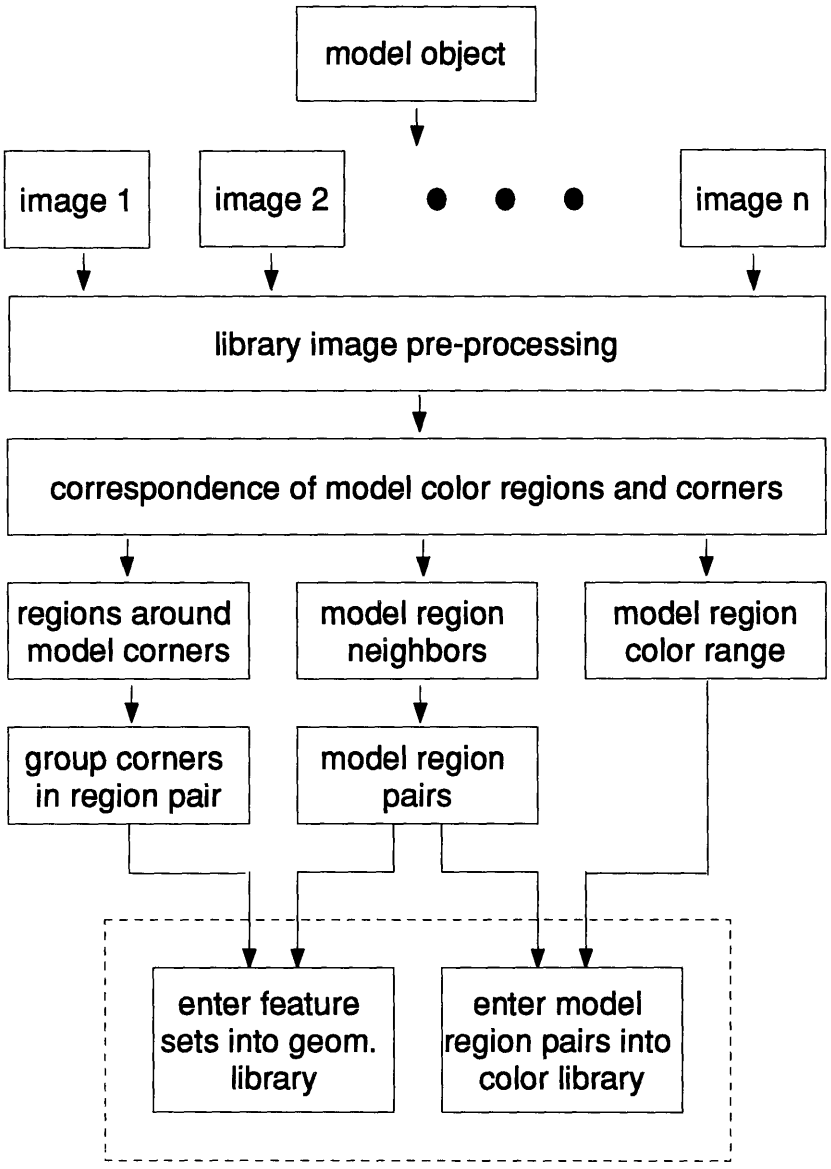


Figure 4-1: Creating library entries for a model. The output of processing on the model is shown in the dotted box.

depicted in Figure 4-1.

1. A set of RGB images of the object is taken, each from a different viewpoint.

2. Each of the images is pre-processed to obtain corner features and color regions for use in constructing the libraries.

3. The correspondence of the point features and image regions across the set of images is obtained.

4. The range of color for each model region, the adjacency relation between model regions and the surrounding model regions of each corner are computed.

5. For each pair of adjacent model regions, the colors of the two regions are used as indices to the color library and the model ID and the region IDs are entered into the library.

6. For each pair of adjacent model regions, compute the model-based invariant representations for each set of features that comes from the pair of regions, and enter into the geometry library.

We now explain in more detail what each of the above steps do.

4.2.1 Pre-processing of the Library Images

Figure 4-2 illustrates the pre-processing that is done to each library image. Each color image of a model is smoothed to remove some amount of noise from the image sensing process. To obtain color information, the smoothed RGB image is converted to the HSV representation of the color space. The image is segmented into regions that contain roughly the same color, and the adjacency relationship between regions is determined. The color statistics for each region are computed. The particular statistics collected are the average HSV description for each region and the standard deviation in each of the H, S, and V dimensions. The color statistics are used as part of the color description of the corresponding model region. To obtain geometry information, the color image is converted into a gray level image for use by the edge

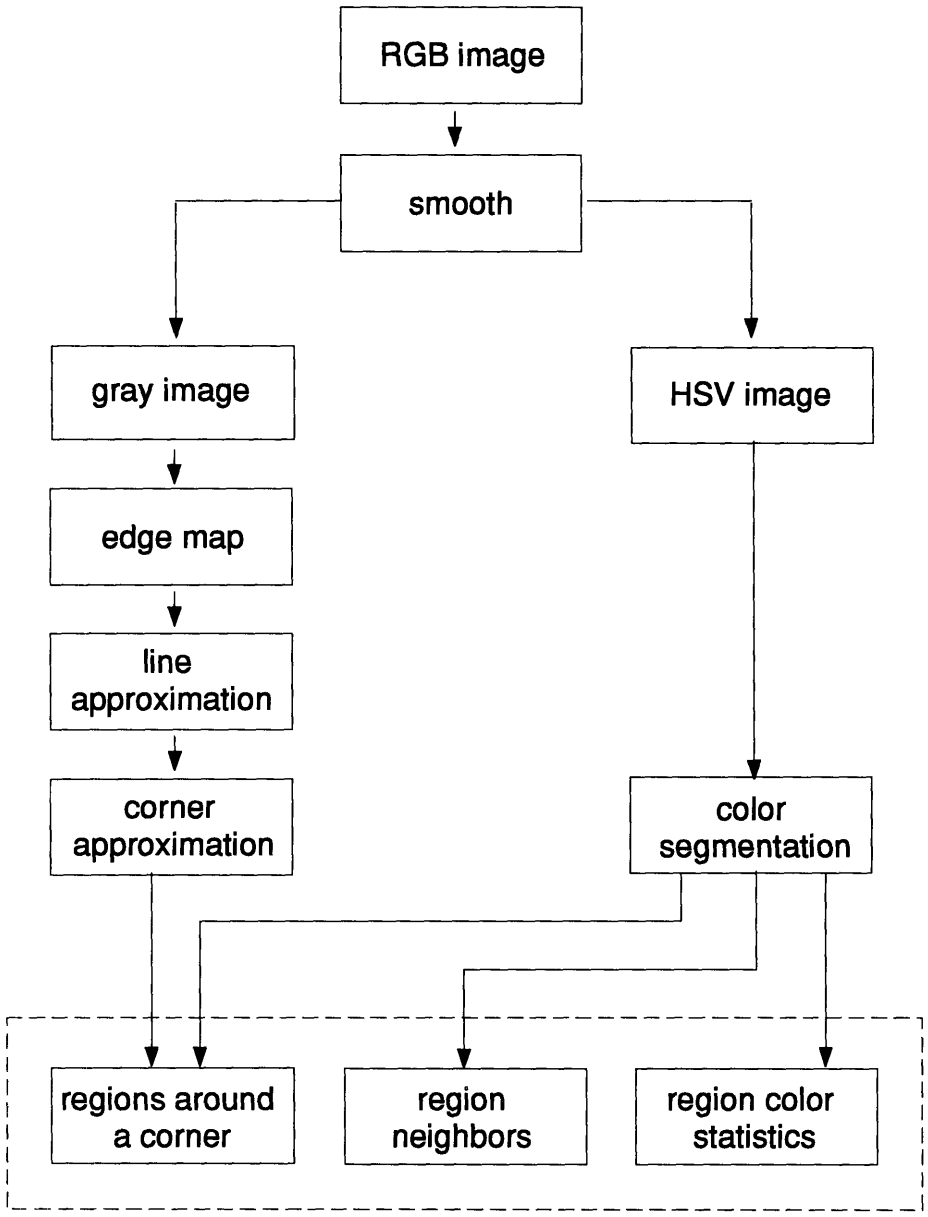


Figure 4-2: Pre-processing of a library image. The output of the pre-processing stage is shown in the dotted box.

detector. The edges in the edge map are approximately represented by line segments. Stable points on the model are detected by intersecting the line segments. Given that the models have straight edges, this method of corner detection is justified. The association between each image corner and all the segmented color regions surrounding it is established. This corner-region association is used to determine the grouping of point features to be entered into the geometry library.

4.2.2 Creating the Libraries

Given the information obtained from each individual image of a model, they need to be coordinated to form a library description of the model. The correspondence across the library images of points and regions in the sequence of images must be established. Given the correspondence and the output from the pre-processing stage, we can deduce the following information,

- *The color of the model regions.* If we assume that the set of images taken of the model has significant variations in the surface normals of each model surface and in the direction of light source, then the union of the description of the image region color should adequately represent all possible appearances of the model region under white light conditions.

- *The adjacency relationship of the model regions.* Assuming that the models are all convex, then the adjacency relationship between model regions is the union of the adjacency relationships of the image regions.

- *The model regions surrounding a model corner.* This is obvious, since we know the image regions surrounding an image corner.

Using the above information, the model can be entered into the color and geometry libraries. Figure 4-3 shows how the entries in the color and geometry libraries are related.

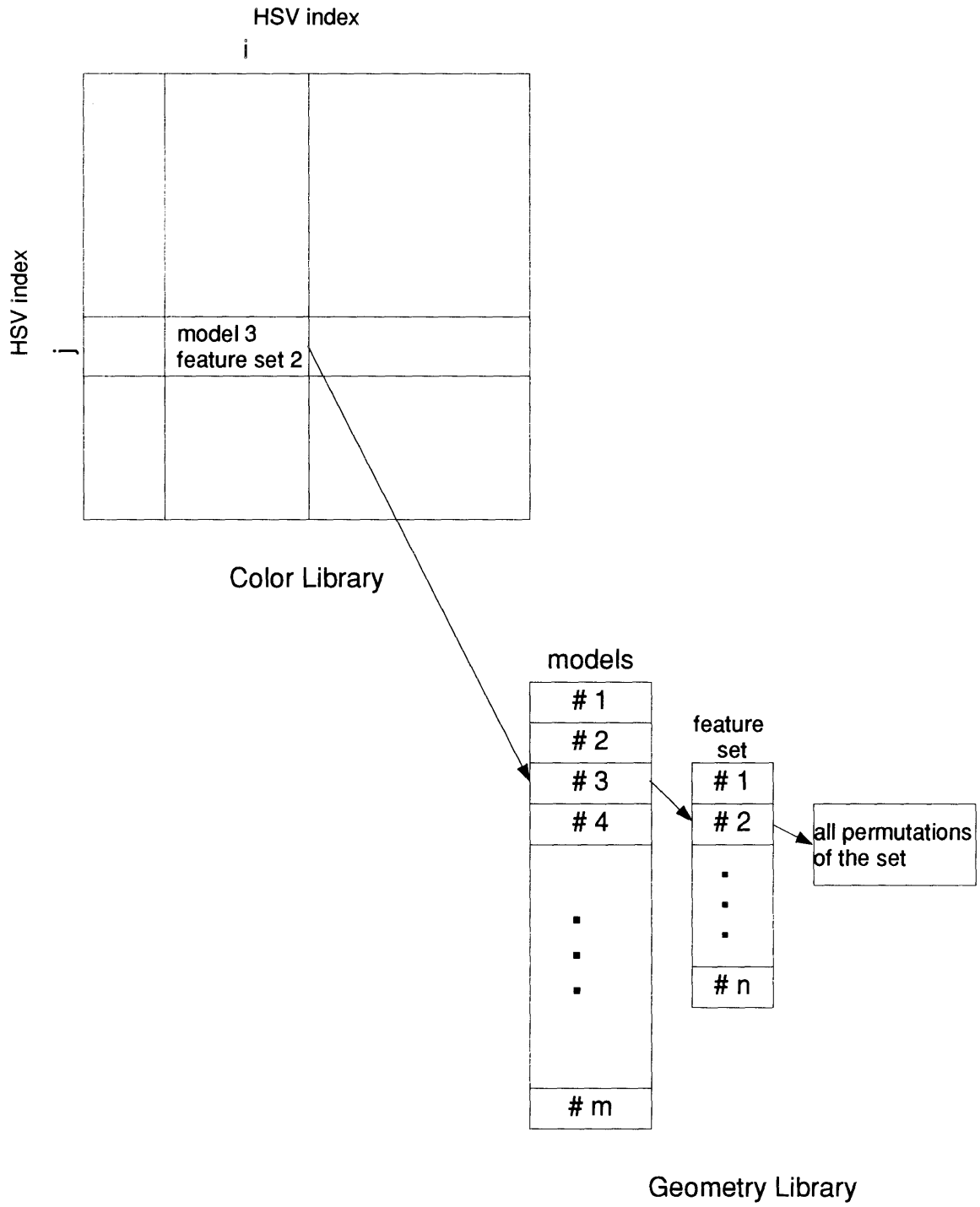


Figure 4-3: The color and geometry libraries. Shown above are the structures of the color and geometry libraries and how they relate to each other.

The Color Library

To represent the model object in the color library, the colors of the model regions and their adjacency relationships are needed. The color library is effectively a two dimensional table indexed by a pair of colors. For the purpose of using the colors as an index to look up entries in the color library, the model colors must be converted to a set of discrete HSV indices. The Cartesian product of the set of HSV indices for a pair of model regions then depicts all possible appearances of the pair of color regions in a image. Thus the model ID should be entered into every entry indicated by the Cartesian product of the HSV indices.

The Geometry Library

The geometry library contains the model-based invariant descriptions [18] of clusters of model features combined with the affine description of coplanar points. In this thesis, only sets containing five point features are used, though any size of sets with five or more features can be used. The decision of whether any set of five points from the model should be represented in the library hinges upon the following:

- that the five points appear together in at least one instance of the model images, assuming that the model images are taken from representative view points that can describe all configurations of model point features, or equivalently, any five points that can possibly appear together will be in one of the model images.
- that the five points appear in a pair of adjacent model regions.

Given an ordered set of five point features, the system computes the model-based invariant if the first four points are non-coplanar, or the affine coordinates of the fourth point if the first four points are coplanar, or the affine coordinates of the fourth and the fifth points if the five points are coplanar. The entry made into the geometry library contains information about into which of the three categories the five points fall.

4.3 The Recognition Engine

Figure 4-4 depicts the routines that the recognition engine invokes when given an image. The input image is first pre-processed for color and geometry information, similar to the pre-processing on library images. Then the colors of a pair of adjacent image regions are used to look up an entry in the color library. The entry will indicate whether or not there is a model in the library which has those two colors in a pair of adjacent model regions. If there is a hit in the color library, the color component will guide the recognition system in selecting a set of point features that should contain few spurious data and in giving the shape-based recognition component a hint of what the object might be.

4.3.1 Pre-processing of a Scene Image

The pre-processing done on a new scene image is similar to that of a library image, except that the corner features of the library images are computed at the pre-processing step, and the same is not done for a scene image. Corner feature detection is left to a later stage because in a scene there are line segments from different objects that intersect to form false points that do not belong to any stable feature of any model. Computing the corner features first would result in those points being counted toward an image region, thus adding spurious data to the sets of features that need to be looked up in the geometry library. On the other hand, the library images are known to contain one object only; thus there is no need to select corner features.

4.3.2 Using the Libraries

Once the color segmentation of a scene image is computed and the adjacency relationships between color regions are known, the recognition engine takes each pair of adjacent image regions, converts the average HSV value for each region into an HSV index, and uses the pair of indices to look up an entry in the color library. If that entry contains a list of model objects that have a pair of adjacent model regions that have similar colors, then the recognition engine will try to locate image features for the

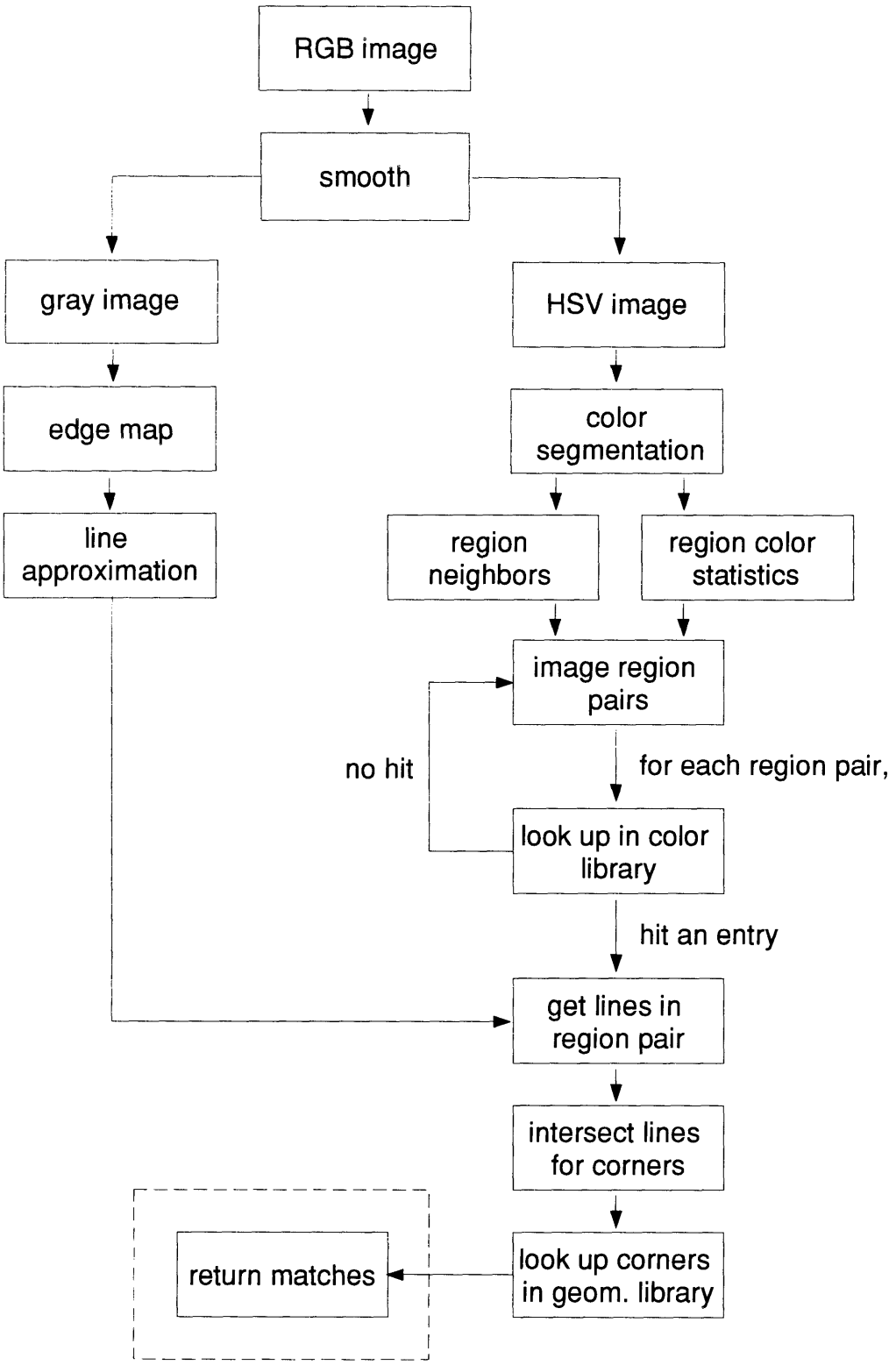


Figure 4-4: The recognition process on a scene image. The output of the recognition stage is a set of matches between the model objects and image features.

geometric indexing system. The system first locates all line segments that are inside or significantly border the image region pair, then intersects those line segments to obtain corner features. These corner features are more likely to be stable points from those two region than the corner features computed from line segments that may be from outside the image region pair. In doing this, the color component performs its selection function of grouping corner features. The color component also performs an indexing function because the entries in the color library indicate the model object that could have that pair of color regions.

The corner features selected by the color component is used to find consistent matches in the geometry library. Ideally, a grouping system is needed here to decide which subsets of image corner features should be used to look up consistent matches in the library, as is done in [18]. However, to simplify the coding, the recognition engine takes the simplest and most time-consuming strategy—to take every subset of five corner features and try to find consistent matches in the geometry library. The system then returns all matches between sets of model features and sets of selected corner features.

Note that since only clusters of a fixed number of points are used to check for consistency between model and image features, the matches produced by the recognition engine cannot be considered as the definitive solution to the recognition of objects in the image region pairs. Instead, a verification system is needed to compare the global matching of the model features and the selected image features and to determine conclusively the presence of a particular object at a particular location.

Chapter 5

The Implementation

In the previous chapter, we presented an overview of how the entire system is put together and how the subparts of the system interact with each other. In this chapter, we discuss how each process in the system is implemented. The structure of this chapter mirrors that of the previous chapter, so the reader may refer back to be reminded of the high level view of the system.

5.1 The Libraries

The recognition system pre-computes information to be stored in two libraries. The color library is a table indexed by a pair of colors, and each entry contains the IDs of the model and the corresponding regions that have this pair of colors. The geometry library contains the model-based invariant representations of sets of five point features. Since we are using the result of look-up on the color library to guide the shape-based recognition, the geometry library must be indexed by the model ID and the IDs of the model region pair. As discussed in the previous chapter, the library construction process can be divided into preprocessing of the library images and construction of a model representation.

5.1.1 Pre-processing of Library Image

A set of color (RGB) images of the model is obtained. These images are taken from views that are significantly different and representative of all possible appearances of the object. Each image is smoothed to remove some of the noise in the imaging process. The smoothing algorithm is based on the trimmed mean operator discussed in [14]. The trimmed mean smoothing is a compromise between the mean operator and the Gaussian operator. It gives a more stable estimate of the mean on fat-tailed distributions than the mean and Gaussian smoothing. We use an operator of size 3×3, and average the middle 3 values for each channel of the RGB image separately. To prevent smoothing at the edges between regions of an image, an edge map is given to the smoothing module, which will constrain smoothing when there are three or more edge pixels in the neighborhood operated on by the trimmed mean smoothing process.

The smoothed RGB image is converted to the HSV representation of the color space using a method presented in [7]. I will discuss in the following section how the HSV image is segmented to obtain regions that are of roughly uniform color.

Color Segmentation

Two color segmentation algorithms were tested, one based on local connectivity in the image space, and one based on quantization of the color space. The local connectivity in the image space algorithm involves using the color distance between two neighboring pixels. The two pixels belong to the same region if the distance between them is less than some threshold, and not the same region otherwise. The single scan connectivity algorithm presented in [15] is used to assign consistent labels to all pixels in the same region. Through experiments, we came to the conclusion that an algorithm based on the local connectivity in the image space fails too frequently. It often labels two neighboring regions of different colors as one single region. This failure occurs at locations where the transition of color between two regions is smooth. This smooth transition between two colors can possibly be attributed to color aberration at the edge between two true regions. Through this excursion, we came to the

conclusion that a successful color segmentation algorithm should be able to tolerate some amount of color variation within a single solid colored region, but should also limit the amount of "leaking" that will bring in regions of a different color.

We consider a second method for color segmentation which uses hierarchical quantized color space. To restrict the amount of color variations that a single region can have, we quantize the color space into bins and assign to each pixel the bin number that corresponds to its color. Each region of a segmented image is assigned a unique label using the connectivity algorithm in [15]. To group pixels from a region whose color straddles the boundary of a bin, the boundaries of the quantized color space are shifted by roughly 50 percent of the bin size in the hue dimension, and less in the S and V dimensions. We perform a first round of segmentation with the quantized space, then compute the average color of each region. With the average color of each region, we quantize the color space again, with a shift in the boundaries of the bins, and assign to each pixel of a region the color bin number of the average color of the region. Ideally, we want to group together pixels whose colors are more similar than those pixels whose colors are less similar. We approximate this idea with the use of a fine to coarse quantization of the color space. The first segmentation uses a fine quantization of the color space. The quantized color space is shifted, and color segmentation is performed again. The same two-step process is repeated with successively coarser quantization of the color space. Note that this process is monotonic in that regions can only grow in size, and can never be broken up by subsequent segmentations. Figure 5-1 illustrates the color segmentation using the quantized color space method. Both color and image spaces are simplified to one dimension. Figure (a) shows the 1D image signal. Clearly, this signal should be segmented into three regions. Figures (b) through (e) show partial results from the hierarchical-two-step-with-shifting segmentation scheme. Figure (f) shows what might happen without the hierarchical approach—a transition region that falls in the middle of a bin in a coarse quantization of the color space can itself form a large region.

The particular color segmentation algorithm used in this thesis involves three quantizations of the color space, each with its shifted counterpart. The color images

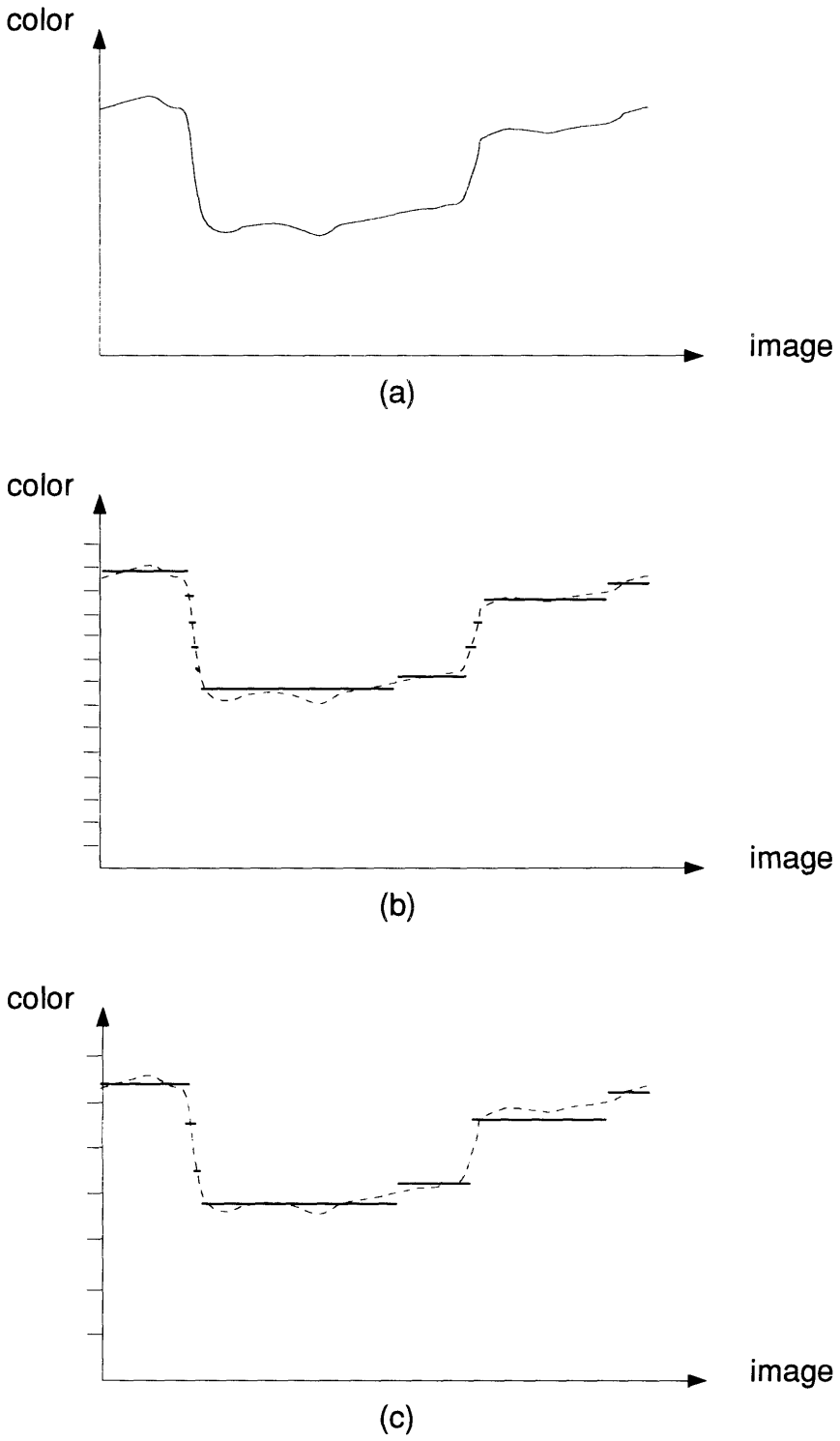


Figure 5-1: Color segmentation using quantized color space. (a) shows the 1D image. (b) shows the result of segmentation using a fine quantization of the color space. (c) shows the result of using a medium quantization of the color space. (continue on next page)

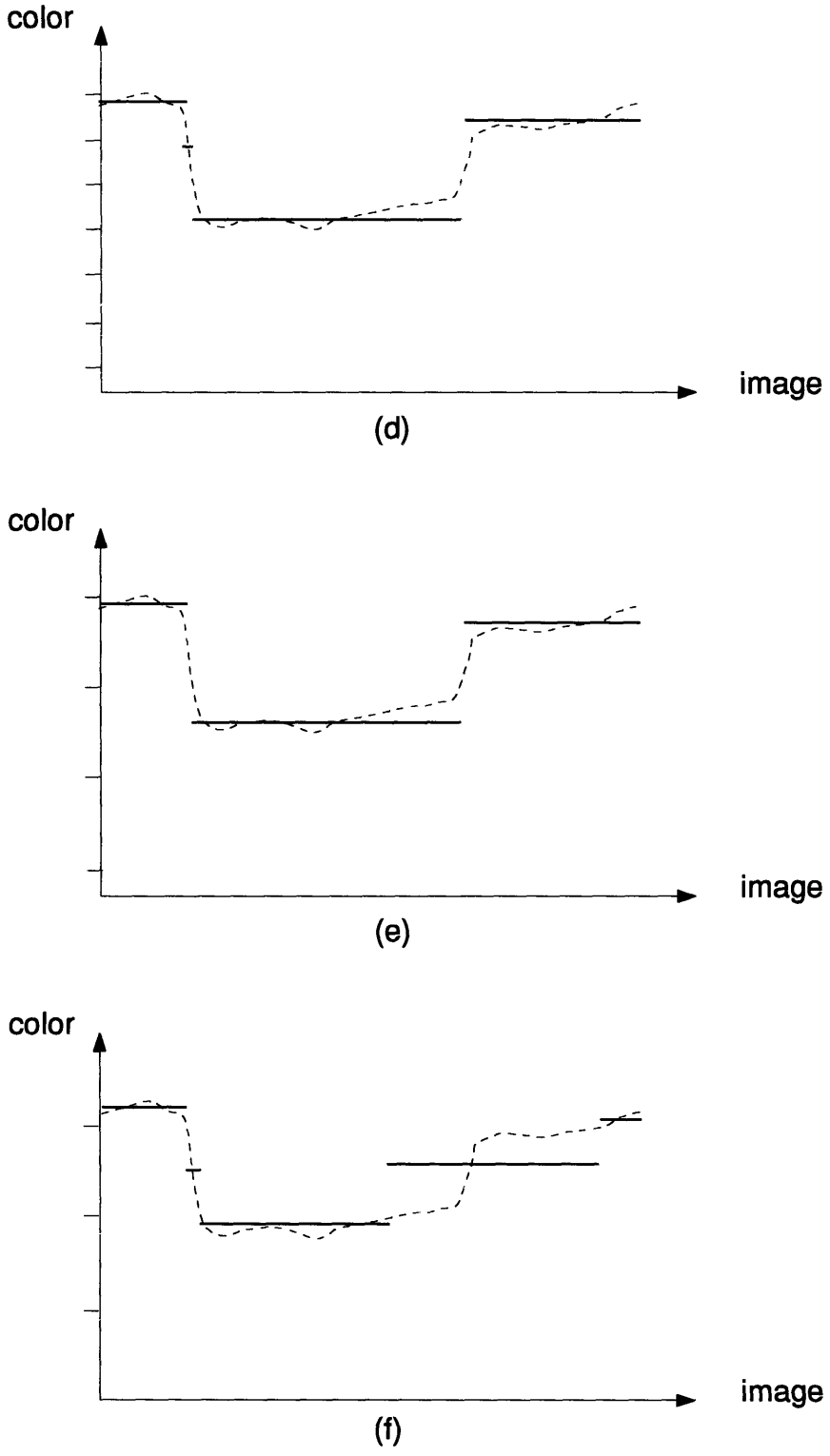


Figure 5-1 continued: (d) shows the effect on the color segmentation by shifting the quantized color space. (e) shows the effect of using a coarse and shifted quantization on the result of (d). (f) shows the effect of color segmentation without the hierarchical approach.

are represented in the HSV space, where H is between 0° and 360°, S and V are between 0 and 1. Because the intensity component V is more affected by lighting condition changes than the H and S components, it should be coarsely quantized. The hue component better describes color than the saturation component, thus H is finely quantized. We choose 5° in the H dimension of the first quantization of the color space, because we found experimentally that within a solid color region, the variation in the hue component is roughly between 2.5° and 7°. The upper bound on the H dimension is set at 10°, because we wish for the system to distinguish yellow from light orange, which differ by 30° in hue, and 10° is a conservative bound which can tolerate up to 7° in the variation of hue within a solid colored region. Table 5.1 gives the details of the quantization and shifting parameters for each step. The parameters appear in the order that the six color segmentation steps are done. These parameters have shown to work well in the experiments that we have run. We have not exhaustively tested all possible parameters.

Bin Size			Shifting		
H(deg)	S	V	H(deg)	S	V
5	0.1	0.1	0	0	0
			5	-0.01	0
8	0.1	0.125	1	0.02	-0.05
			5.5	-0.03	0
10	0.1	0.125	3	0.04	0.05
			6.5	-0.04	-0.05

Table 5.1: Quantization of the HSV color space for color segmentation.

The result of color segmentation using quantized HSV space is over-segmented. It contains many small regions that are not significant enough to use for geometric structure. There are also cases where the transition regions between two large regions are grouped into one region, thus removing the adjacency relationship between the two regions that we need in order to enter the model into the color library. These transition regions tend to be very thin and long. Thus the segmented image is processed to remove very small regions, and regions that are small and are very thin and long.

The gaps created by removing these insignificant regions are filled up by dilating the large regions. Some of the regions may still be over-segmented. A process to merge neighboring regions that have similar colors is applied to the segmented image. The average color and the standard deviation in each color component are computed. The adjacency relationship can be computed by using the connectivity algorithm of [15].

Shape Features

To obtain shape features, we need to compute the edge map of the image. On a gray level image, the Canny edge detector [4] is the most frequently used method. It is possible to apply the Canny edge detector on each individual channel of the RGB image. However, the result of this operation will have edges at different locations, resulting in thick edges and poorly localized lines. Another method to detect edges on a color image is to use the intensity values of the image only. Intensity value is a fixed linear combination of the R, G, and B components. The disadvantage of this method is that colors having equal intensity cannot be distinguished. We use a principle color projection presented in [24] to convert a color image into a gray level image. The method involves solving for the largest eigenvalue in the RGB space and projecting each (R, G, B) color description onto the eigenvector corresponding to the largest eigenvalue. The eigenvector in the color space indicates the direction in which the spread of the color pixels is the largest, thus maximizing the contrast between colors and reducing the probability that two color regions are assigned the same gray level. The Canny edge detector can then be applied to the gray level image.

The edges in the edge map are approximated with line segments. We use the split-and-merge algorithm proposed by [16]. Intuitively, this algorithm when given a string of connected edge pixels will find the "critical points" along the curve which best describe the shape of the curve. These critical points define the lines that best approximate the curve. The corner features in a library image can be found by intersecting line segments. The criteria we use to determine if an intersection of two lines is a valid corner are based on heuristic rules. An intersection is deemed valid if the two lines are not nearly parallel, the intersection is not too far from the closest

endpoints of both lines, and the triangle formed by the intersection and the two far endpoints of the two lines has a large area and is not too flat in any direction. A real corner is likely to have several lines intersecting at the same location. Error in the line approximation process can cause the lines not to intersect at the same position. To correct for this problem, clusters of intersection points that are within a small radius are average to obtain a single corner point. The regions surrounding a corner are obtained by searching a fixed neighborhood around a corner for color regions.

5.1.2 Creating the Library Representation of a Model

The input description of each model is a set of n images, where we have chosen in general to be 7 or 8. To obtain a library description of the model, the features computed from the pre-processing step on each individual image must be coordinated. To simplify the system, we obtained the correspondence between point features across the set of images by hand. The same is done to obtain the correspondence of region features across the images. Given the correspondence, we can now describe the color of the model regions, the adjacency between the regions, and the regions surrounding each model point. As described in the previous chapter, the adjacency relationship and the regions surrounding each corner are just the union of the same data computed for each image. The color description for each model region can be intuitively thought of as the union of the color descriptions of the corresponding image regions. To tolerate for the variation in the color appearance of a region, we use the range of color that is within two standard deviations (in each of the H, S, and V dimensions) from the average color of a region. The range of color is converted into a set of discrete HSV indices by quantizing the HSV space into bins of 5° in H, 0.1 in S, and 0.2 in V. The color description of the model region is the union of the color indices of all corresponding image regions.

The Color Library

The model is entered into the color library by the colors of each pair of adjacent regions. The set of Cartesian products of the set of HSV indices of a pair of adjacent

regions gives the indices on the color table to which the model should be entered. The entry includes the model ID, and the IDs of the two model regions, or (m, r_1, r_2), where m is the model ID, r_1 and r_2 are the IDs of the two regions. The color library is very sparsely populated, thus it is implemented as a hash table. The size of the hash table grows at most linearly in the number of models and the number of regions for each model, since some models may share entries with other models.

The Geometry Library

The information stored in the geometry library is a combination of the model-based invariant from [18] and the affine coordinates representation from [19]. The point features from the model are grouped into sets that share the same pair of regions. For each of these sets, all subsets of 5 points that appear together in at least one library image, and all permutations of the 5 points are used to compute the affine representations.

According to the formulation of the model-based invariant, the lines in the α and β subspaces are defined by two non-degenerate views of the model object. Jacobs constructed the library using multiple views of the model object. Since the models we used for testing were very simple in structure, we instead opted to measure the 3D coordinates of the model corner features and use them directly to compute the model-based invariant for the clusters of 5 points. As described in the previous chapter, three different representations are possible. If the first four points are non-coplanar, then the model-based invariant is computed. If the first four points are planar, the affine coordinates of the fourth point are stored. If the five points are coplanar, the affine coordinates of the fourth and the fifth points are stored. For the model-based invariant, the description we store in the geometry library is the analytical representations of the lines.

We discovered through trial and error that the discrimination ability of any set of four points in the parallelogram configuration is very weak. This is because the affine transformation allows scaling and shearing in each dimension independently. Thus any rectangle and parallelogram of any size and shear have the same representation

in the affine space. In a man-made world, the rectangle is a very dominant structure, thus making a lot of things "look" the same in the affine space. For this reason, we chose to eliminate the parallelogram from the geometry library.

5.2 The Recognition Engine

The preprocessing done on a scene image is the same as that for a library image. The only difference is that in a scene image, the corner features are detected after the regions for further processing are chosen. The recognition engine takes each pair of adjacent image regions, converts their color descriptions into the discrete HSV indices, and looks up the entry by the index pair in the color library. A hit in the library indicates that there is a model that has this pair of colors, and geometric information is needed for further processing.

The system chooses all line segments belonging to the pair of regions to intersect and get corner features. A line segment is a part of a region if both of its end points are within some distance of the region. This criterion is very simplistic because it is possible that line segments could significantly border on a region without having both ends inside or near the region. To correctly choose line segments inside a region, we must "walk" the entire line and check whether each point is close to the region of interest. If there are a large number of pixels close to the region, then the line is part of the region. As stated in Chapter 4, the advantage of corner detection by intersecting line segments from the region of focus is that there will be fewer spurious points, since intersections of line segments from outside of the region are not considered. The disadvantage of using only the line segments inside the regions of interest is that the localization of some of the corner features may not be as accurate as the corner features obtained with intersection of all nearby lines, because fewer lines are used to determine the position of those points.

Given the point features selected by the color component, the recognition engine takes every subset of 5 points and attempts to find matches in the geometry library. Since the geometry library represents the model-based invariant analytically, the sys-

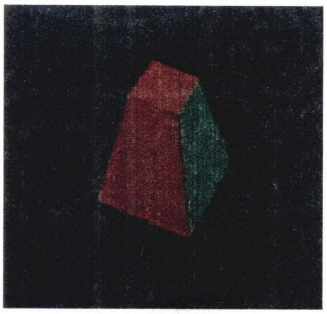
tem must determine for each line if the affine representation of a given set of 5 image points is on that line. The color component gives the geometry component a hypothesis of what the object might be, thus only features from those models need to be tested. To handle errors on the affine coordinates produced by the positional uncertainty of the point features, we use the upper bound on affine coordinates shown by [11].

To determine the bound of error on the image positional uncertainty of a point feature, we computed the difference between the position of corners detected by the system and the positions of corners determined by a human observer. The difference between the two sets of corner positions in general differ between 2 to 5 pixels.

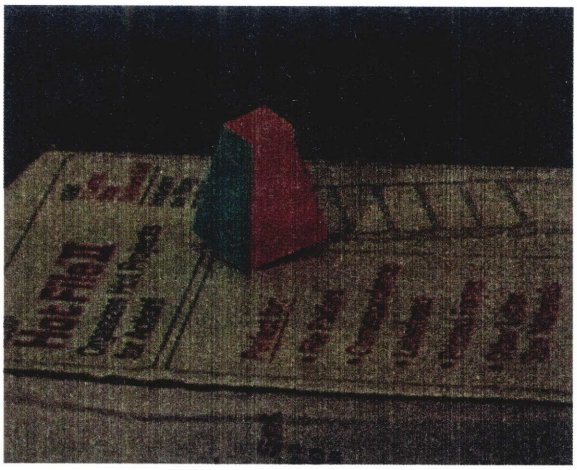
The recognition engine returns a list of all model features that match the image features. As stated in the previous chapter, this result cannot be used as the definitive solution to the recognition task, because only part of the image data is used. A verification system is needed to finish off the task.

5.3 Summary

We will show a list of figures that will give the reader a sense of what the input and the output of each stage look like, whenever pictorial output is appropriate. We will show the the processing done on a scene image for recognition. The processing done for library images mirrors that for a scene image.



An object in the model library.



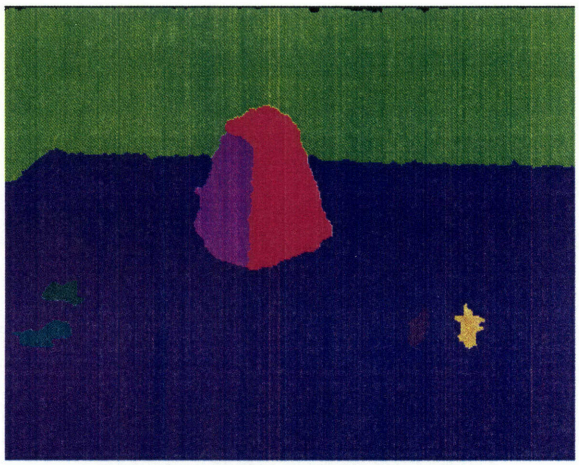
A scene image.

Result of color segmentation using quantized HSV space.

Figure 5-2: Images to demonstrate the input and the output of some of the steps of the recognition system.

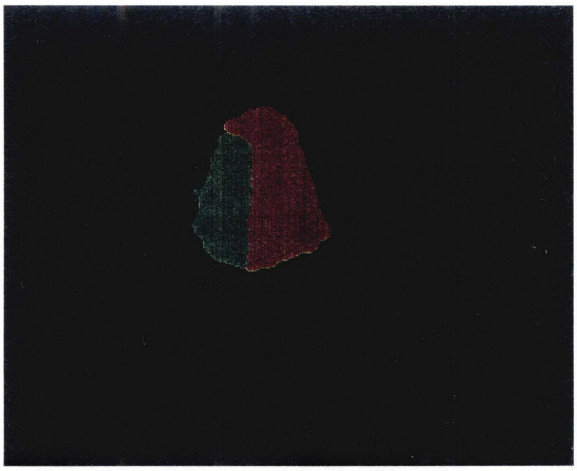
Result of removing small regions.

Result of dilating large region to fill the gaps.

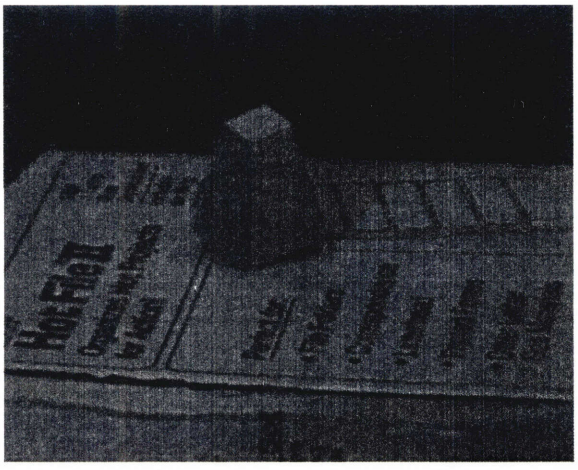


Result of merging neighboring regions with similar colors.

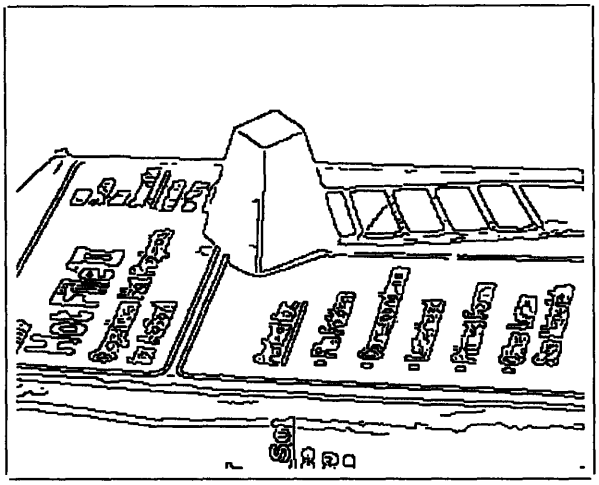
The scene image reconstructed from the segmented image.



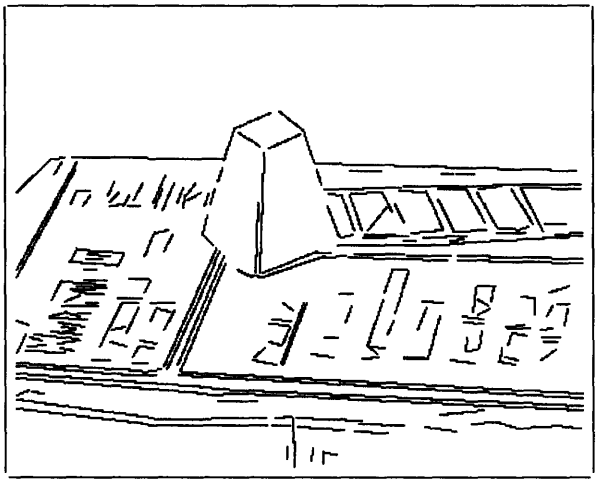
The regions selected by the color component.



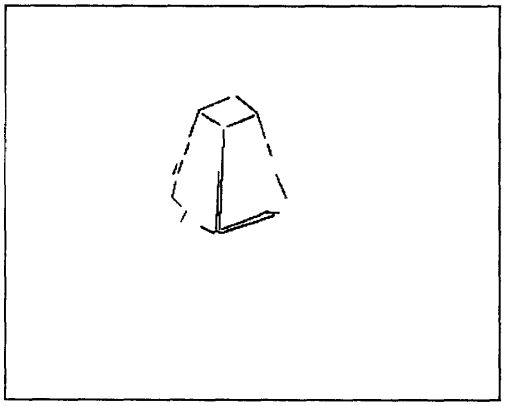
The color image converted to gray-level image.

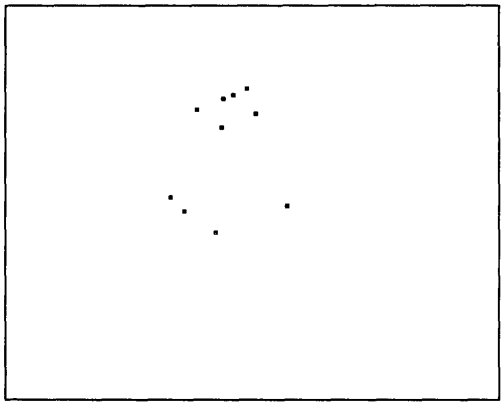


The edge map produced by Canny edge detector.



Line approximation of the edge map.





The line and point features selected by the color component.

Chapter 6

Experiments

We have presented in the previous chapters a recognition system which uses color information to guide the geometric recognition process. Color plays two roles in this system: (1) it is used to select the features from the image which might have come from a single object, and (2) it used to index into a color library to obtain hypotheses of what the objects in the image might be. The shape-based recognizer then takes the selected features and attempts to find matches between these features and the hypothesized object.

We constructed a model library containing the three objects shown in Figure 6.3. As described in Chapter 5, seven to eight images of the model objects were take. The library construction phase collects the color and geometry information on each individual library image. Given the correspondence of the geometric features across the entire set of images for a model, the library construction phase derives a representation of the model and stores the representation in the color and the geometry libraries.

We now test the recognition system to demonstrate the effectiveness of the use of color information. The original intention was to test how well color serves its selection and indexing roles by examining the response of the geometry component. The three planned sets of tests were:

1. Use color for selection and indexing.

number of perfect points	number of spurious points	image error bound	ideal number of match	matches returned by the system
8	0	0.2	40	40
8	3	0.2	40	71

Table 6.1: System behavior with perfect data.

2. Use color for selection only.

3. Eliminate color from the recognition process entirely.

We expected to find that with each successive removal of the roles played by color, the geometry system would run slower and produce more incorrect matches between the image features and model features. However, as the reader will see, our planned tests could not be carried out because of errors in the shape-based recognizer whose origins we have not ascertained. As a consequence, we will show the color component of the system performing the selection and indexing tasks and then show the reduction in the number of features from which any shaped-based recognition system can benefit.

6.1 Testing the Geometric Indexing Component

We tested the shape-based recognition component by using perfect data generated from one model object, the "p-g-cube," and then giving the recognizer the correct hypothesis of the model object. We allowed for numerical error by using 0.2 pixels as the uncertainty in the image coordinates of point features. As one would expect, the system finds all possible matches between the image data and the model features. We then added in spurious points into the perfect data so that the data given to the shape-based system contains 27.3% of spurious points. Using the same bounded error on image coordinates of points, the number of image subsets that matched any model features almost doubled. Table 6.1 lists our findings.

We next applied the shape-based system on the image given at the end of the previous chapter, that of the "p-g-cube." The color component was able to select the features in the region and produce the correct hypothesis for the object in the image.

The color component selected ten features for the shape-based system to find matches in the model object. Of the ten features, seven were from the object, and three were spurious. Of the points that came from the object, we compared the results of corner detection by hand with the corners automatically detected by the system. The result showed that the maximum error in the position of the corner locations is 2.5 pixels. Of the points that did come from the model object, we determined by hand that there were 18 subsets that should match some subsets of features from the model library. The number of possible subsets of five points with computable affine coordinates is 21 for the set coming from the object[1] and 251 for the entire set of points selected by the color component[2]. We tested the system by giving it the entire set of data and then eliminating the spurious data, each time varying the amount of bounded error in the image coordinates of the point features. We list below the relevant quantities:

1. Number of points in the the entire data set = 10.
2. Subsets from the entire data set with affine representation = 251.
3. Number of points from the model = 7.
4. Subsets from the model points with affine representation = 21.
5. Number of spurious points = 3.
6. Number of subsets that have matches in the model library = 18.
7. Approximate bounded error on the image coordinates = 2.5 pixels.

Assuming that a shape-based verification system randomly chooses a subset of five points from the entire data set to use for recognition, it has a $18/251 \approx 7.17\%$ chance of selecting a subset that comes from the model library. We subjected the shape-based system to the following tests: (1) using only data from the object, varying the amount of bounded error on the image position of points, and (2) using the entire

[1] 7 choose 5 is 21.

[2] 10 choose 5 is 252. However, there is one set where the affine coordinates cannot be defined because four of the five points are collinear and the fifth point is nearly collinear

Include spurious data?	bounded image error	matches found by the system	number of correct matches	% of correct matches
No.	2	5	5	100.0
	3	8	7	87.5
	4	13	9	69.2
	5	17	14	82.4
	10	20	17	85.0
	12	21	18	85.7
Yes.	2	70	5	7.1
	3	80	7	8.8
	4	121	9	7.4
	5	155	14	9.0
	10	238	17	7.1
	12	251	18	7.2

Table 6.2: Success rate of the system with data from the "p-g-cube" image.

data set, varying the the amount of bounded error on the image position of points. In both experiments, we used the percentage of correctly identified image sets over the number of all image sets identified by the system. Table 6.2 presents our findings.

As can be seen in the table, once spurious data is added, the shape-based system does not appear to perform significantly better than randomly choosing sets of points. We have not isolated the cause of this phenomenon. However, we speculate that a number of factors could be involved. We estimated the bounded error of image coordinates of points to be 2.5 pixels. However, our estimation is based on corner detection by hand on an image of the scene, which already contains the effect of perspective distortion. The error model used in this system and in [18] was developed for planar models only. It is unclear how this model will hold in the 3D case (see [1] for a treatment of the 3D error). Jacobs showed experimentally that the discrimination ability of a set of five points is quite weak. A set containing a larger number of points is much more desirable and should perform much better.

The overall system presented in this thesis lacks a verification system. As a result, we do not have an automated system to determine whether a match between the image features and model features is correct. Our system returns all possible

matchings between model and image features. Since arbitrary nonsingular linear transformation on 3D objects allows for too many degrees of freedom, the system returns many more matches than a rigid object can possibly produce under scaled orthographic projections. The large number of matches produced by the system makes it difficult to determine by hand the correct matches between the model and image features. This combined with the uncertainty of the effectiveness of the shape-based component lead us to the conclusion that we will only test the system on the color selection and indexing functions.

6.2 The Experiments

We would like to show how well the color component performs selection and indexing. Eight images, including the one appearing in the previous chapter, will be used to demonstrate the capabilities of the color component of this system.

Of the eight test images, the color component of our system was not able to segment two of the images. For example, Figure 6-2 shows an instance of such a failure. It should be noted that the "leaking" effect of the segmentation occurred not at the stage that color segmentation is done through quantizing the color space, but at the later stage of merging neighboring regions. Of two other instances, the conversion from the color image to the gray-level image did not produce enough intensity change between some neighboring color regions, and the Canny edge detector was not able to detect the boundaries between these regions.

In the remaining four images, the color component had a reasonable measure of success. The first image is given in the previous chapter. The color component was able to correctly segment the image into regions of roughly the same color, to group features, and to produce a hypothesis of the model in the image. Without the use of color segmentation, the "p-g-cube" image has 210 line segments in the image. The color selection chose 21 of the lines as relevant to the model under consideration. Of the 113 points in the image, the color selection step chose 10 as belonging to part of the model.

Another image tested on the system is shown in Figure 6-3. As in the previous case, the system was able to correctly segment the regions, select out the objects of interest, and produce a hypothesis for each region under consideration. The number of lines in the entire image is 137, and the number of points in the image is 69. The color selection process chose 21 lines for the region corresponding to the "p-g-cube," and 15 lines for the region corresponding to the "r-b-cube." It also chose 9 points for the "p-g-cube" region, and 9 points for the "r-b-cube." Note that part of the image in Figure 6-3 is set to black. That is because under the HSV representation of color, the hue component of the representation becomes unstable as the color comes close to any shade of gray, since small amount of noise could cause the H component to undergo a large change in angle. Since grouping pixels in dark regions in the HSV space color representation is not very stable, we have arbitrarily set a threshold of S = 0.15 below which a pixel is set to black.

We had implemented the color library to tolerate change in any of the color channels that are within two standard deviations of the mean. We tested the color component on an image that does not contain any known object, but has color regions that are similar to a model object (Figure 6-4). The color system correctly came to the conclusion that there was no known object in the scene. However, this behavior is very stable. Figure 6-5 shows that the color component missed the portion of image that came from the "p-g-cube." The reason for this miss is that the lighting condition under which the object is viewed is very different from the conditions under which the model images were taken. In particular, the "p-g-cube" in the image is in the shadow of the lamp shade, thus it is not well illuminated. The color system is sometimes too sensitive to color regions. Figure 6-6 shows such a case. The system correctly chose three regions that do belong to a model object, then chose a fourth one which does not belong to it. The orange region was selected as a region deserving further attention because it is near a region that is part of a model object, and it is close enough to the right color which should accompany the region from the model. In this case, the two standard deviations in the allowed color variation may have been too large.

6.3 Discussion

A few things are revealed through the experiments on the color component: (1) that color segmentation is hard, (2) that color constancy or a higher level component to compensate for lighting change is needed, (3) that color indexing can work if we can obtain a stable color description and that image color segmentation produces reasonable results. We have also seen that using the principle color as the projection vector and converting all RGB pixels to gray level for the purpose of edge detection is a viable method, but it still fails too often.

The color segmentation algorithm using quantized color space appears to work well, as long as the space is quantized into reasonably small buckets so that each bucket does not encompass two colors that the user wishes to distinguish. This would result in over-segmentation of a region—thus the need for a merge step to consolidate regions that are over-segmented. However, the merging step can cause "leaking" if the color regions between two true regions do not have a sharp transition in the color. This lack of a sharp transition could be caused by color aberration or by the blurring due to the fact that cameras do not usually follow the pin-hole image formation model. For the experiments above, the merging step can be made to work by changing its parameters. However, we have used a fixed set of parameters in the merging step for all experiments.

Part of the issue of using color for segmentation is what color representation to use. We have used the HSV representation. However, this choice may not be the best. The HSV representation is capable of factoring the intensity mostly into one channel, but it is also a space with a singularity in hue along the line that defines all shades of gray. Because the HSV space is non-uniformly quantized[3], given the some amount of error in the RGB values, the uneven quantization can produce a large fluctuation in the HSV representation when the color is dark, and a small fluctuation when the color is bright, even when the change in the RGB values in the two instances may be the

[3]The HSV representation is typically represented on a cone. There are just as many bins near the tip of the cone as there are at the base of the cone.

same. A possible alternative to the HSV representation may be to use the color ratio as done in [21], with the use of an intensity channel. As mentioned before, intensity information is needed to distinguish dark and bright objects of similar shades and hue.

To obtain edges from a color image, we have projected the RGB representation of each pixel onto the principle component in the color space. Even though in general this works well, it still fails to detect many obvious edges. Other methods for directly detecting edges in color space, or an adaptive method of converting color images to gray images is needed.

We have used color for the purpose of indexing into a library. Assuming that a stable representation of color is possible, we could do a lot more than using just two colors. Multiple color regions may be used to index into a library to produce a more reliable hypothesis of the presence of an object than the hypothesis obtained using only two colors. Color can also be used to give a partial ordering of geometric features, thus reducing the number of permutations that need to be stored in the library. We have not explored using color to order geometric features, but it should be clear the partial ordering will greatly reduce the size of the library that need to be stored.

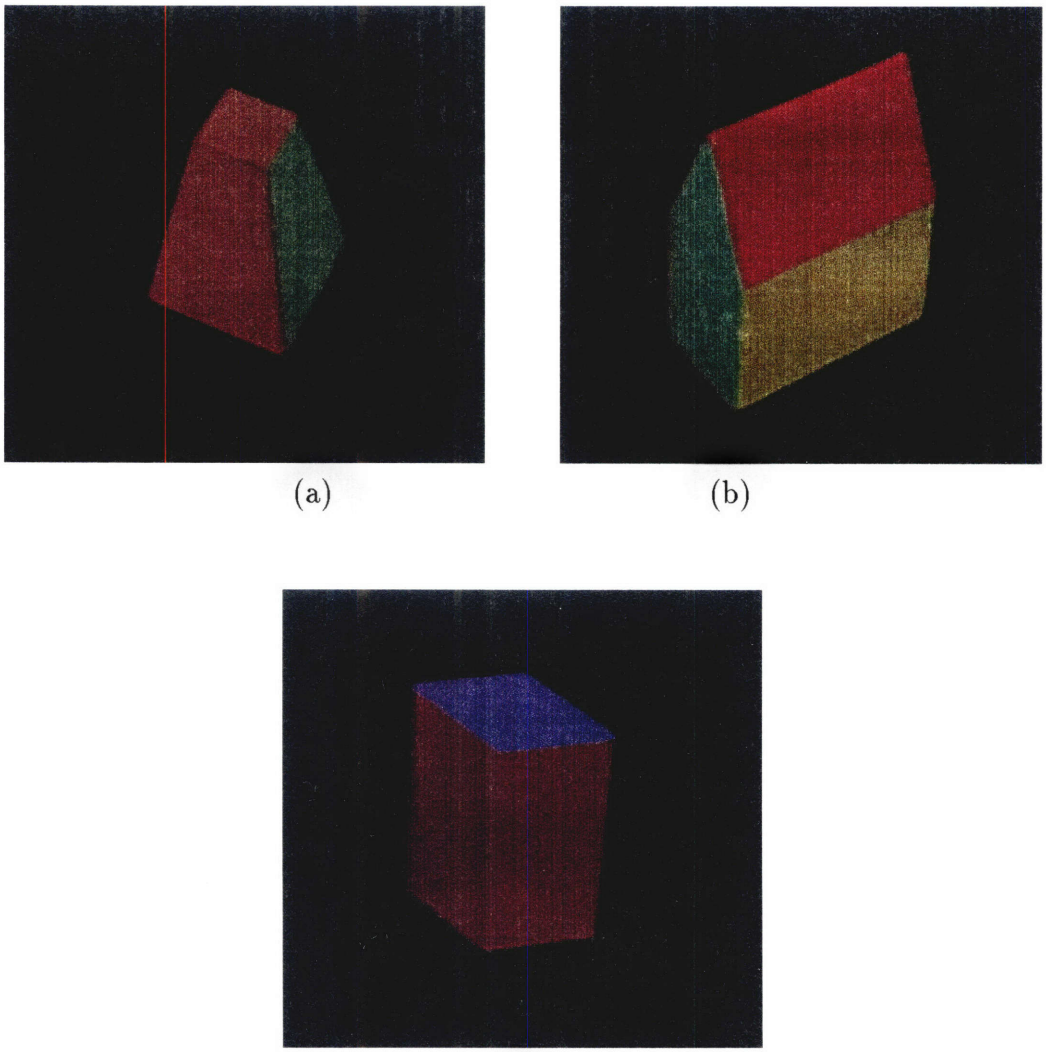
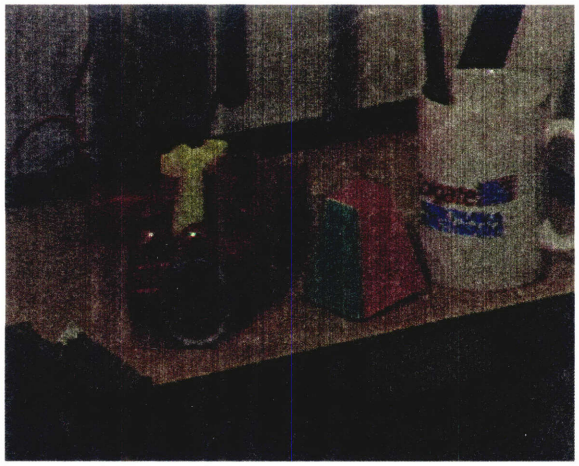
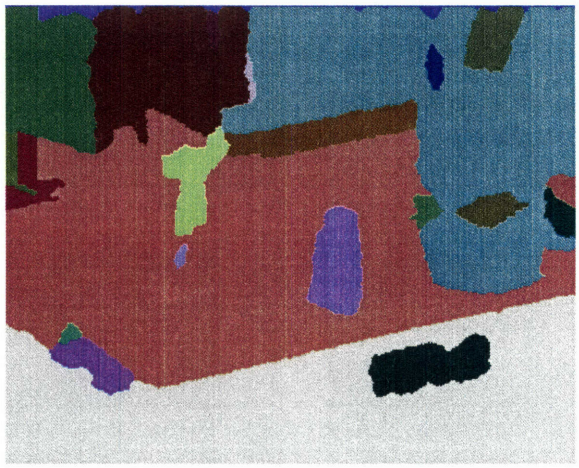


Figure 6-1: The three model objects: (a) p-g-cube, (b) house, and (c) r-b-cube.

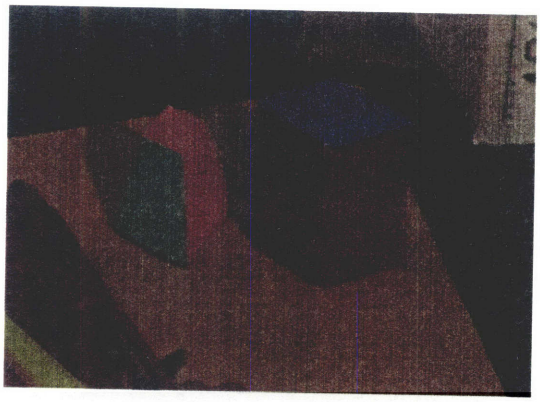


(a)

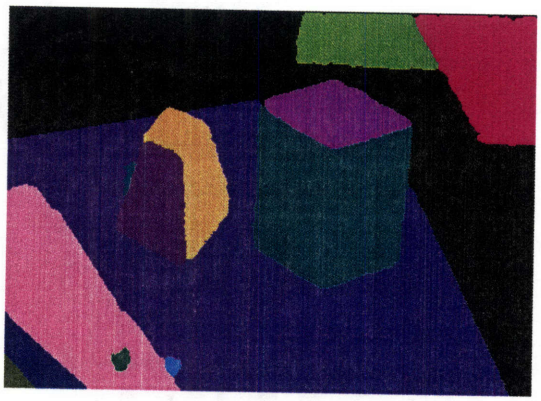


(b)

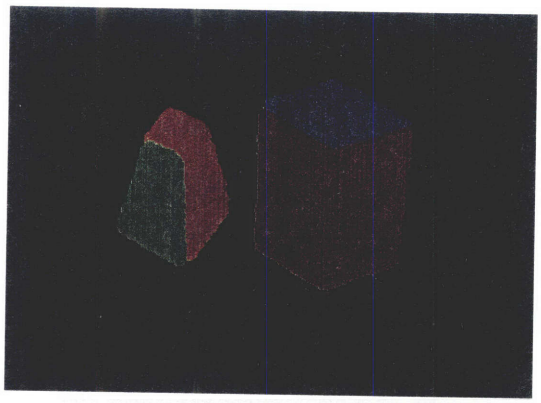
Figure 6-2: An example of the failure of the color segmentation algorithm. (a) is the original image, (b) is the result of color segmentation.



(a)



(b)



(c)

Figure 6-3: An image on which the color component operated successfully. (a) is the original image, (b) is the result of color segmentation, (c) is the result of color selection and indexing. (continued on the next page)

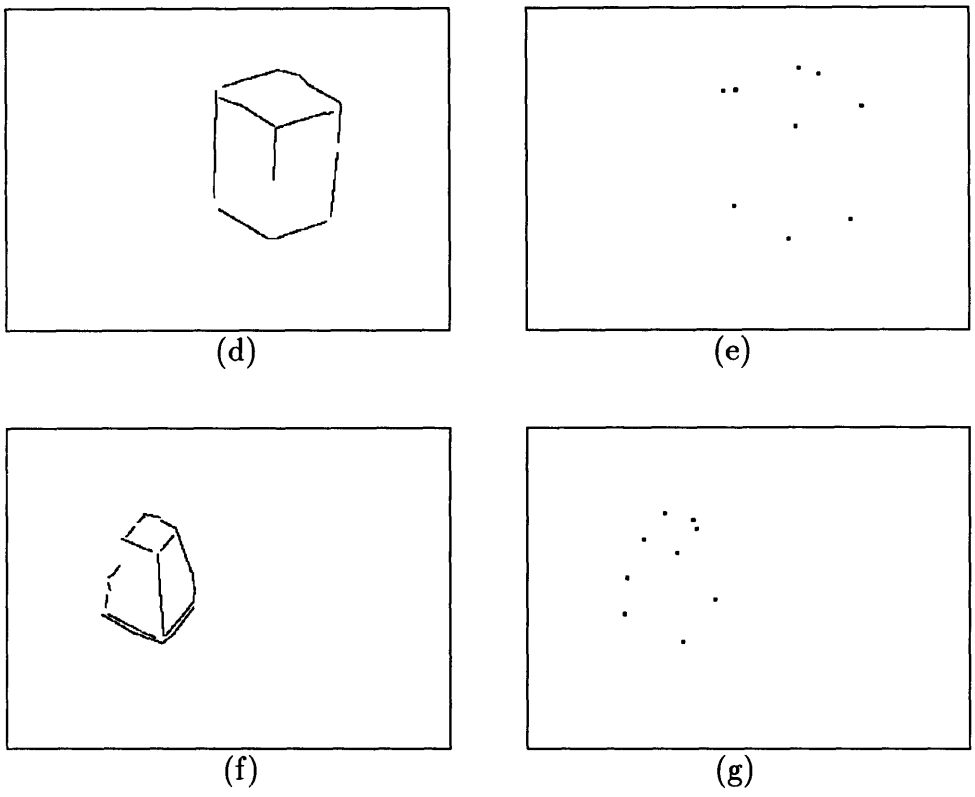
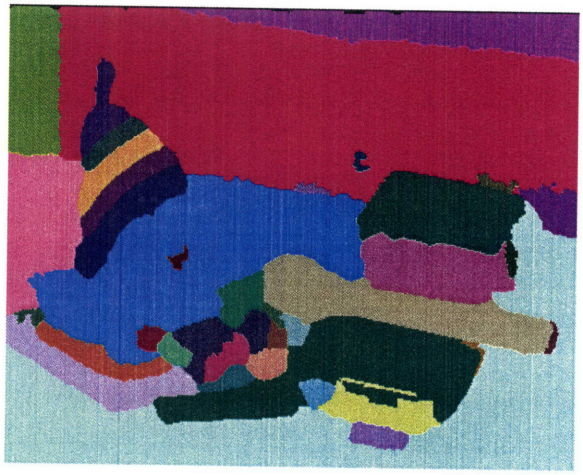


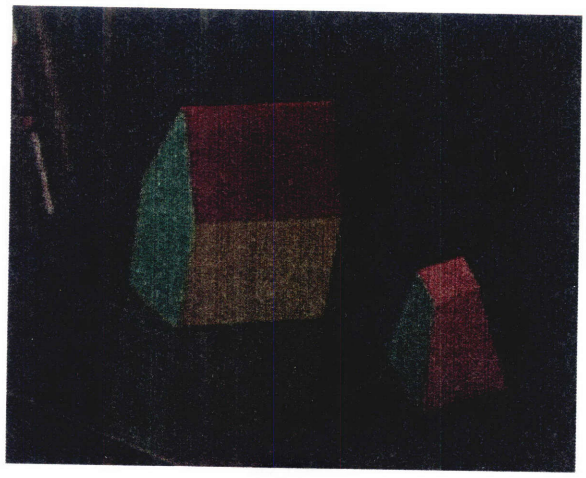
Figure 6-3 continued: (d) shows the first set of line features selected by the color component, and (e) the result of intersecting those lines. (f) shows the second set of line features selected by the color component, and (g) the result of intersecting those lines.

(a)

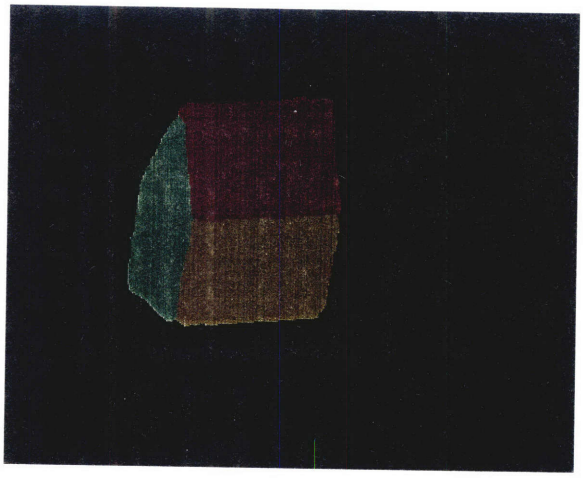


(b)

Figure 6-4: An image without any known model object. The color system made the correct hypothesis: no object present.

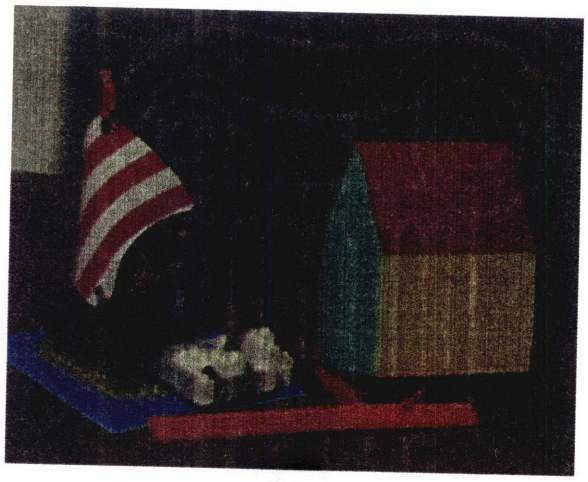


(a)

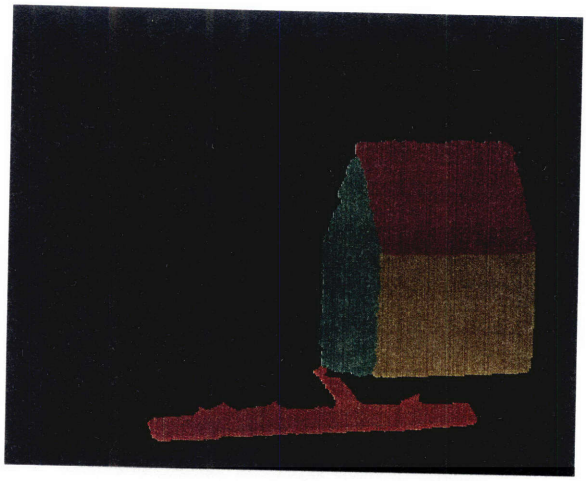


(b)

Figure 6-5: An image containing two model objects. (a) is the original image, (b) shows the regions selected by the color component for further processing. The color component failed to report the presence of the "p-g-cube", because its intensity is a not within the 2 standard deviations in color that are tolerated by the library.



(a)



(b)

Figure 6-6: An image on which the color component falsely reported the presence of an object. The color component took the bright orange to be a part of a model region because it is connected to another region which is from a model object, and the color of the bright orange is close to red, the neighbor of the yellow region.

Chapter 7

Conclusions

In this thesis, we have presented an object recognition system which pre-computes a library description of a set of objects. Our system employs two cues for recognition. The color cue selects parts of a given image that may come from a single object and then uses the color of these parts of the image to index into the color library to find models that could have given rise to the image color regions. Once the color component has a hypothesis of the model in the region, it feeds the set of image features and the set of possible models to the other component of the recognition system, the shape-based recognition system. The shape-based system ideally finds all consistent pairings image features and model features. This approach of using color as a filter for the geometry system is based on the fact that shape-based recognition alone suffers from a combinatorial explosion as the number of features is increased. It would help if another system could filter out irrelevant information before giving it to the shaped-based recognition system.

We have chosen to use the HSV representation of the color space on which to perform segmentation, selection, and indexing into the library. Our experiments on the color component show that if color segmentation can provide reasonable results consistently, and that objects are seen under white light conditions, color can be used to select features and to produce hypotheses on these features to aid the shape-based recognition system. Much work remains to develop a robust color segmentation algorithm and a stable description of color.

We chose the geometric indexing method proposed by [18] as the geometry component of the recognition system. This decision is based on the fact that Jacobs's representations of objects reduces the recognition space of the object to 1D manifolds, and can be used for indexing by simply looking up locations in the recognition space to see if an image feature set falls onto any of the manifolds representing model feature sets. As shown in Chapter 6, our success with this system is limited, particularly when error and spurious data are introduced into the system. The possible cause of this may be due to the effect of perspective distortion, and that the error bound used in the system was developed with the assumption of planar objects.

Although we made the decision to use both the color cue and the shape information, there is no reason to suggest that this approach would not work using other cues. In fact, the recognition system could become more robust if more cues are used. The issue with using more cues is how to integrate them. There is a clear relationship between the color component and the geometry component in our system—the color component provides the shape-based recognizer with a set of image data and a subset of the model library, and the geometry component produces the matching between the given sets. However, if another component is added, say texture, assuming that texture will be used in the same way that color is used with respect to geometry information, it is unclear how the texture and color components will interact between themselves.

There is much room for extension in the topic of incorporating multiple cues for object recognition. This thesis serves as the beginning of an exploration of building a fast recognition system, and using multiple cues from the scene to improve its performance.

Bibliography

[1] Tao Alter and David W. Jacobs. Error propagation in full 3d-from-2d object recognition. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*. Proceedings of the IEEE, June 1994. Also see MIT AI Memo 1476.

[2] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[3] T. M. Breuel. Indexing for visual recognition from a large model base. AI Memo 1108, Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1990.

[4] John F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8:34–43, 1986.

[5] David Clemens and David Jacobs. Space and time bounds on model indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1007–1018, 1991.

[6] Francois Ennesser and Gérard Medioni. Finding Waldo, or focus of attention using local color information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8), August 1995.

[7] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics: Principles and practices*. Addison-Wesley, 1990.

[8] David Forsyth et al. Invariant descriptors for 3-D object recognition and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):971–991, October 1991.

[9] W. Eric L. Grimson. The combinatorics of object recognition in cluttered environments using constrained search. *Artificial Intelligence*, 44(1–2):121–166, 1990.

[10] W. Eric L. Grimson. *Object Recognition by Computer:The role of geometric constraints.* The MIT Press, Cambridge, Massachusetts, 1990.

[11] W. Eric L. Grimson, Daniel P. Huttenlocher, and David W. Jacobs. Affine matching with bounded sensor error: A study of geometric hashing and alignment. AI Memo 1250, Massachusetts Institute of Technology Artificial Intelligence Laboratory, October 1991.

[12] W. Eric L. Grimson and Tomás Lozano-Pérez. Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research*, 3(3):3–35, 1984.

[13] W. Eric L. Grimson and Tomás Lozano-Pérez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):469–482, 1987.

[14] Robert M. Haralick and Linda G. Shapiro. *Computer And Robot Vision.* Addison-Wesley, 1992.

[15] Berthold K. P. Horn. *Robot Vision.* The MIT Press, 1986.

[16] S. L. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *Journal of the ACM*, 23:368–388, 1976.

[17] Daniel P. Huttenlocher and Peter C. Wayner. Finding convex edge groupings in an image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition.* Proceedings of the IEEE, 1991.

[18] David W. Jacobs. Recognizing 3d objects using 2d images. AI TR 1416, Massachusetts Institute of Technology Artificial Intelligence Laboratory, May 1993.

[19] Y. Lamdan and H. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proceedings of the International Conference on Computer Vision*, pages 238–249, Tampa, FL, 1988. Proceedings of the IEEE.

[20] JianChang Mao and Anil K. Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern Recognition*, 25(2):173–188, 1992.

[21] Kenji Nagao and Eric Grimson. Recognizing 3d object using photometric invariant. AI Memo AIM-1523, Massachusetts Institute of Technology Artificial Intelligence Laboratory, February 1995.

[22] Shree K. Nayar and Ruud M. Bolle. Reflectance ratio: A photometric invariant for object recognition. Technical Report RC 18500, IBM Research Center, Yorktown Heights, NY, November 1992.

[23] Wayne Niblack et al. The QBIC project: Querying images by content using color, texture, and shape. IBM Research Report RJ 9203 (81511), IBM Research Division, Almaden Research Center, San Jose, CA, February 1993.

[24] John A. Richards. *Remote Sensing Digital Image Analysis–An Introduction.* Springer-Verlag, 1986.

[25] Michael Swain and Dana Ballard. Color indexing. *International Journal of Computer Vision*, 7(1), 1991.

[26] Tanveer Syeda-Mahmood. Attentional selection in object recognition. AI TR 1420, Massachusetts Institute of Technology Artificial Intelligence Laboratory, February 1993.

[27] D. Thompson and J. L. Mundy. Three-dimensional model matching from an unconstrained viewpoint. In *Proceedings IEEE Conference on Robotics and Automation*, page 280. IEEE Computer Society Press, 1987.

[28] Shimon Ullman and Ronen Basri. Recognition by linear combinations of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):992–1006, October 1991.

[29] Richard Vistnes. Texture models and image measures for texture discrimination. *International Journal of Computer Vision*, 3:313–336, 1989.