A Probabilistic Approach to Geometric Hashing using Line Features

by
Frank Chee-Da Tsai

a dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

Department of Computer Science
New York University

September 1993

Approved: _____
Professor Jacob T. Schwartz
Research Advisor

Dedication

This dissertation is dedicated to Professor Jacob T. Schwartz for his invaulable advisory, to my parents for their endless love and to Buddha, whose wisdom cheers me through the dark patches of my life.

Acknowledgements

My most sincere gratitude goes to my research advisor Professor Jacob T. Schwartz for his generous guidance, without which this dissertation can not be possible. To whom, I also owe the understanding of identifying research directions of scientific interest.

I would also like to express my gratitude to Professor Wen-Hsiang Tsai of National Chiao-Tung University, Hsin-Chu, Taiwan, Republic of China. I took his course "Image Processing" when I was an undergraduate senior. This is my first taste of applying computer technology to the processing of images. After two-year R.O.T.C. military service upon graduation, I came here to the Courant Institute in 1987 to further my study. Professor Stéphane Mallat's "Computer Vision" course again intrigued my interest in the field of image analysis.

Thanks are also due to Professor Jaiwei Hong, Professor Robert Hummel, Professor Richard Wallace, Professor Haim Wolfson, Dr. Isidore Rigoutsos, Mr. Ronie Hecker and Mr. Jyh-Jong Liu for various kinds of helps and discussions.

I also thank the staffs in the Courant Robotics Laboratory, for the days we worked together, especially Dr. Xiaonan Tan for her encouragement.

Finally, I thank my parents, truly and sincerely, for their patience, support and constant encouragement throughout the work and my whole life.

Abstract

One of the most important goals of computer vision research is object recognition. Most current object recognition algorithms assume reliable image segmentation, which in practice is often not available. This research exploits the combination of the Hough method with the geometric hashing technique for model-based object recognition in seriously degraded intensity images.

We describe the analysis, design and implementation of a recognition system that can recognize, in a seriously degraded intensity image, multiple objects modeled by a collection of lines.

We first examine the factors affecting line extraction by the Hough transform and proposed various techniques to cope with them. Line features are then used as primitive features from which we compute the geometric invariants used by the geometric hashing technique. Various geometric transformations, including rigid, similarity, affine and projective transformations, are examined.

We then derive the "spread" of computed invariant over the hash space caused by "perturbation" of the lines giving rise to this invariant. This is the first of its kind for noise analysis on line features for geometric hashing. The result of the noise analysis is then used in a weighted voting scheme for the geometric hashing technique.

We have implemented the system described and carried out a series of experiments on polygonal objects modeled by lines, assuming affine approximations to perspective viewing transformations. Our experimental results show that the technique described is noise resistant and suitable in an environment containing many occlusions.

Contents

List of Figures

Chapter 1

Introduction

One of the most important goals of computer vision research is object recognition. This humanlike visual capability would enable machines to sense and analyze their environment and to take an appropriate action as desirable.

We consider intensity-image techniques. Range data is usually harder to obtain. There is also reason to believe that human vision emphasizes intensity images and that most practical applications of computer vision can be tackled without range information [42]. Specifically, we consider the problem of 2-D (or, flat 3-D) object recognition under various viewing transformations. We are interested in (1) large model bases (more than ten objects); (2) cluttered scene (low signal noise ratio); (3) high occlusion levels; (4) segmentation difficulties. The current approaches to image segmentation generally produce incomplete boundaries and extraneous edge indications. Therefore, any approach to object recognition has to cope with segmentation defects. To work in environments containing many occlusions, we impose minimal segmentation requirements — only positional information of edgels is assumed to be available.

We will describe the analysis, design and implementation of a recognition system that can recognize, in a seriously degraded intensity image, multiple objects which can be modeled as collections of lines.

1.1 Model-Based Object Recognition

Recognition can be achieved by establishing correspondences between many kinds of predicted and measured object properties, including shape, color, texture, connectivity, context, motion, or shading. Here, we will focus upon the problem of achieving spatial correspondence. This is often prerequisite to examining correspondences along the other dimensions.

Prior research [7,13] has indicated that a model-based approach to object recognition can be very effective in overcoming occlusion, complication and inadequate or erroneous low level processing. Most commercial vision systems are model-based ones, in which recognition involves matching an input image to a set of predefined models of objects. A key goal in this approach is to precompile a description of a known set of objects, then to use these object models to recognize in an image each instance of an object and to specify its position and orientation relative to the viewer.

1.1.1 Problem Definition

Object recognition can be conceptualized in various ways. A brief survey of the literature on this subject demonstrates this point [7,8,13]. Here, we adopt the definition given by Besl and Jain [7].

Their definition is motivated by the observation of human visual capabilities. Two main steps are involved. The first step is *learning*. When a new object is given, human visual system gathers information about that object from many different viewpoints. This process is usually referred to as *model formation*. The second step is *identification*.

The following definition of the *single-arbitrary-view model-based object recognition problem* is motivated by the above:

1. Given any collection of labeled solid objects, (a) each object can be examined as long as the object is not deformed; (b) labeled models can be created using information from this examination.

2. Given digitized sensor data corresponding to one particular, but arbitrary field of view of the real world, given any data stored previously during the model formation process, and given the list of distinguishable objects, the following questions must be answered:

- Does the object appear in the digitized sensor data ?
- If so, how many times does it occur ?
- For each occurrence, (1) determine the location in the sensor data (image), (2) determine the location (or translation parameters) of that object with respect to a known coordinate system (if possible with given sensor), and (3) determine the orientation (or rotation parameters) with respect to a known coordinate system.

This problem statement allows the possibility of using computers to solve it; at any rate the problem is clearly solvable by human beings.

1.1.2 Object Recognition Method

In most object-recognition systems, one can distinguish five sub-processes: image formation, feature extraction, model representation, scene analysis and hypotheses verification.

Image Formation

This process creates images via sensors, which can be sensitive to a variety of signals like visible spectrum light, X-rays, etc. As said, we will concentrate on intensity images.

Feature Extraction

This process acts on the sensor data and extracts relevant features. The geometric features we are interested in might in principle be lines, segments, curvature extrema, curvature discontinuities, conics and so on. But we will use lines exclusively.

Model Representation

This process converts models to the quantities which will be used to recognize them. This process is closely related to the feature-extracting process, since the same features are supposed to be extracted from a scene for matching.

Models based on geometric properties of an object's visible surfaces or silhouette are commonly used because they describe objects in terms of their constituent shape features. Although many other models of regions and images emphasizing gray-level properties (e.g.

texture and color) have been proposed, solving the problem of spatial correspondence is often a prerequisite to their use. Thus, recognition of objects in a scene is always apt to involve construction of a shape description of objects from sensed data and then matching the description to stored object models.

A shape representation scheme usually involves at least two components:

1. the primitive units, e.g. vertices (0-dimension), curves and edges (1-dimension), surfaces (2-dimension, e.g. plane surface, quadratic surface) and volumes (3-dimension, e.g. generalized cylinders);

2. the way those primitives are combined to form the object models.

Such a representation scheme must satisfy the following two criteria to be considered of good quality:

1. *Near-Uniqueness:* Ideally each object in the world should have a limited number of representations; otherwise, when image features are derived, there will be also many choices of configuration for each representation and thus increase the computational complexity to match configurations of image features to model representations.

2. *Continuity:* Similar objects should have similar representations and very different objects should have very different representations.

Scene Analysis

This process involves an algorithm for performing matching between models and scene descriptions. This process is most crucial in an object recognition system and is the focus of this thesis. It recovers the transformations the model objects undergo during the image-formation process. Sometimes a *quality measure* can be associated with each model instance detected; this measure can be used as a measure of belief for further decision making in an autonomous system. We will classify basic matching techniques and review them in chapter 2. The techniques we are interested in involve those allowing partial occlusion, viewing distortion and data perturbation.

In this dissertation, we will focus on using the geometric hashing technique for matching. The geometric hashing technique is very good as a *filter* capable of eliminating many candidate hypotheses as the identity of the objects [38].

Hypotheses Verification

This process evaluates the quality of surviving hypotheses and either accepts or rejects them. Usually an object recognition system projects the hypothesized models onto the scene and the fraction of the model accounted for by the available scene signals is computed. If this fraction is below a pre-set (usually obtained empirically) threshold, the hypothesis fails verification.

1.2 Difficulties to Be Faced

What makes the visual recognition task difficult ? Basically, there are four factors:

1. noise, e.g. sensor noise and bad lighting conditions;
2. clutter, e.g. industrial parts occluded by flakes generated in the milling process;
3. occlusion, e.g. industrial parts overlapping each other;
4. spurious data, e.g. presence of unfamiliar objects in the scene.

The amount of effort required to recognize and locate the objects increases when these four factors become serious. In dealing with these four factors, we have to consider three problems [13]:

1. What kind of "features" should and can be reliably be extracted from an image in order to adequately describe the spatial relations in a scene ?
2. What constitutes an effective representation of these features and their relationships for an object model ?
3. How should the correspondence between scene features and model features be matched in order to recover model instances in a complex scene ?

1.3 The Hashing Approach

In order to recognize 3-D objects in 2-D images, we must either perform a comparison in 2-D or 3-D. Since it is easier to project 3-D models onto a 2-D image than it is to

reconstruct 3-D shapes from 2-D scenes, it is logical, in model-based vision, to perform the comparisons in image space. If we simply project the 3-D models onto image space during the recognition process, we must guess at the projection parameters, and incur the computational expense of multiple projections at run-time. If we precompute many projections, then we potentially substitute many 2-D models for relatively fewer 3-D models.

A more effective method of performing object recognition was proposed by Schwartz and Sharir [33]. The technique exploits the idea of *hashing*. By appropriately selecting a hash function, *dictionary operations*[1] can be performed in $O(1)$ time on the average. By hashing transformation *invariants*, instead of raw data subject to a viewing transformation, the hash function "hashes" to a fixed "bucket" of the hash table before and after whatever viewing transformations are allowed. Thus object models can be represented by its constituent geometric features, some appropriate subset of which is hashed and pre-stored in the hash table in a redundant way. During recognition, the same hash operations are applied to a subset of scene features and candidate matching model features are retrieved from the hash table to hypothesize their correspondence.

This hashing technique trades space for time. We may even pre-compute many projections, substituting many 2-D models for relatively fewer 3-D models.

1.4 Overview of This Dissertation

In this dissertation, we consider highly degraded intensity images containing multiple objects. We emphasize use of four techniques:

- Use of an improved Hough transform to detect line features in a noisy image;
- Use of geometric invariants derived from line features under various viewing transformations;
- Use of the effect of line feature statistics on the perturbation analysis of invariants computed;
- Use of a Bayesian reasoning as the basis for line feature matching.

[1] consisting of "insert", "delete" and "member" operations [1].

1.4.1 Line Extraction

In noisy scenes, the locations of point features can be hard to detect. Line features are more robust and can be extracted by the Hough transform method with greater accuracy. Thus we choose lines as the primitive features to be used.

In chapter 3, we first briefly review the Hough transform technique for detecting line features. Then we point out several factors that adversely affect the performance of the method and propose several heuristics to cope with those factors to improve the performance.

A series of experiments relating to this point are presented.

1.4.2 Line Invariants

In chapter 4, we first examine the way in which coordinate changes act on various spaces of potential interest for recognition. This allows us to define a method of encoding a line using a combination of other lines in a way invariant under suitable geometric transformations. Potentially interesting transformations considered include rigid, similarity, affine and projective transformations.

1.4.3 Effect of Noise on Line Invariants

In chapter 5, we model the statistical behavior of line parameters detected by the Hough transform in a noisy image using a Gaussian random process with mild assumptions. We analyze the statistics of the computed invariants and show that these have a Gaussian distribution in a first order approximation.

Analytical formulae for various transformations including rigid, similarity, affine and projective transformations are given.

1.4.4 Invariant Matching with Weighted Voting

In chapter 6, we use the result of chapter 5 to formulate a Bayesian maximum likelihood pattern classification as the basis of weighted voting scheme for matching line features by Geometric Hashing.

We have implemented a system that makes use of these ideas to perform object recognition, assuming affine approximations to more general perspective viewing transformations.

Both synthesized and real images were used in experiments. Experimental results are given in chapter 7. It is seen that the technique is noise resistant and usable in environments containing many occlusions.

Chapter 2

Prior and Related Work

Numerous techniques have been proposed for object recognition. A brief survey and classification of those techniques is given below. These techniques are not independent of each other; most vision systems combine several of them.

We divide the analysis into three sections. The first section examines object recognition using various classical schemes, and the second two sections discuss the background and existing work in the field of geometric hashing. In this thesis, we study the Hough transform methods, discussed in section 2.1.3 and geometric hashing described in section 2.2. Our work directly builds upon the Bayesian weighted voting scheme of Rigoutsos, which we describe in section 2.3.

2.1 Basic Paradigms

2.1.1 Template Matching

Template matching involves matching an image to a stored representation and evaluating some fit function.

According to their flexibility, templates can be classified into four categories [14]:

- *Total templates* require an exact match between a scene and a template. Any displacement or orientation error of pattern in the scene will result in rejection.
- *Partial templates* move a template across the scene, computing cross-correlation factors. Points of maximum cross-correlation values are considered as locations where

the desired pattern appears; multiple matches against the scene is thus possible.

- *Piece templates* represent a pattern by its components. Usually the component templates are weighted by size and scored against a prototype list of expected features. This method is less sensitive to distortions of the pattern in the scene than the more limited techniques listed above. The subgraph matching technique described later can be viewed as an extension to this technique.

- *Flexible templates* are designed to handle the problems of scene deviations from prototypes. Starting with a good prototype of a known object, templates can be parametrically modified to obtain a better fit until no more improvement is obtained. (This technique has been successfully applied to the sorting of chromosome images.) A difficulty is that flexible approaches tend to be more time-expensive than rigid approaches.

Most of these template matching techniques apply only in two-dimensional cases and have little place in three-dimensional analysis, where perspective distortion comes into play. One can store a dense set of tessellations of possible views, but this results in enormous computational time in matching. For basic techniques of template matching, see [15].

2.1.2 Hypothesis-Prediction-Verification

The hypothesis-prediction-verification approach is among the most commonly-used techniques in vision systems. It combines both bottom-up and top-down paradigms.

Hypotheses are generated *bottom-up* among structures in a geometric hierarchy, from structures of edges to curves, from structures of curves to surfaces, and from structures of surfaces to objects. Predictions work *top-down* from object models to images. Once a hypothesis has been made, predictions and measurements provide new information for identification.

Many such schemes use geometric features such as arcs, lines and corners to construct model representations. These features are usually portions of the object's boundary.

In the HYPER system [4], both model and scene are represented in the same way by approximating the boundary with polygons. The ten longest segments are chosen as so-called *privileged segments* which are the focus features used to detect a prospective

correspondence (hypothesis) between object models and the scene. Using this correspondence, a transformation can be computed and nearby features are sought (prediction). The privileged segments, together with their nearby features, are combined to compute a new transformation. Then the verification step follows.

Verification is often accomplished using an alignment method. Usually there is a trade-off between the hypothesis generation stage and the verification stage. If hypotheses are generated in a quick-and-dirty manner, then the verification stage requires more effort. If we want the verification stage to be less pains-taking, then more reliable features have to be detected and more accurate hypotheses produced.

In alignment by Huttenlocher and Ullman [30,31], they consider affine approximations to more general perspective transformations, using alignments of triplets of points. Models are processed sequentially during recognition. For each model, an exhaustive enumeration of all the possible pairings of three non-collinear points of the model and the scene is exercised. Thus the alignment method heavily relies on verification. As a transformation is determined by the correspondence of model and scene features, the model is transformed and aligned to superimpose the image. Verification of the entire edge contour, rather than just a few local feature points, is then performed to reduce the false alarm rate. To cope with the high computational cost of the verification stage, a hierarchical scheme is used: Starting with a relatively simple and rapid check, they eliminate many false matches, and then conclude with a more accurate and slower check.

To sum up, the hypothesis-prediction-verification cycle relates scene images to object models and object models to scene images step by step with refinement in each step. Note also that any scheme using the "hypothesis-prediction-verification" paradigm can be tailored to a parallel implementation by using the overwhelming computing power to generate a great number of hypotheses concurrently and do verifications concurrently.

2.1.3 Transformation Accumulation

Transformation accumulation is also called pose clustering [50] or the generalized Hough transform [5], which is characterized by a "parallel" accumulation of low level pose evidences, followed by a clustering step which selects pose hypotheses with strong support from the set of evidences.

This method can be viewed as the inverse of template matching, which moves the model

template around all the positions of the scene and directly computes a value (usually cross-correlation) as a measure of matching. Transformation accumulation, instead of trying all possible positions of the model in the scene, computes which positions are consistent with model features and scene features. Moreover, its use of all of the evidences without regard to their order of arrival can be advantageous when there are occlusions.

The features used to generate pose hypotheses can be of high level or low level. If the level of features used are high enough, much less computational effort is required for the accumulation stage. However, the trade-off is that higher level features are usually more difficult to extract.

The major problem of this technique usually lies in ensuring that all visible hypotheses are considered within acceptable limits of storage and computational resources. However, as computer power continues to grow while its cost continues to drop, this problem has been growing steadily less significant.

In the generalized Hough transform [5], the transformation between a model and the scene is usually described by a set of transformation parameters. For example, four parameters are needed for the case of two-dimensional similarity transformations: two parameters for translation; one parameter for rotation; one parameter for scaling. Each transformation parameter is quantized in the so-called Hough space. The scene features vote for these parameters which are consistent with the pairings of these scene features and hypothesized model features. One problem of the generalized Hough transform is the size of the Hough space. In terms of two-dimensional cases, we face three-dimensional Hough space for rigid transformations; four-dimensional Hough space for similarity transformations; six-dimensional Hough space for affine transformations.

Tucker *et al.* [53] use local boundary features to constrain an object's position and orientation, which is then used as the basis for hypothesis generation. Their system takes advantage of the highly parallel processing power of the *Connection Machine* [26] to generate numerous transformation hypotheses concurrently and verify them concurrently. The number of processors required for each model is equal to the number of features of the model. Each scene feature participates, in parallel, independently in each processor to match model features for generating transformation hypotheses, which, after verified, are used for evidence accumulation for voting for the pose transformation of the object.

Thompson and Mundy [52] describe a system for locating objects in a relatively unconstrained environment. The availability of a three-dimensional surface model of a polyhedral object is assumed. The primitive feature used is the so-called *vertex-pair*, which consists of two vertices: one is characterized by its position coordinate; the other, in addition to position coordinate, includes two edges that define the vertex. This feature serves as the basis of computing the affine viewing transformation from the model to the scene. Through the voting in the transformation space, candidate transformations are selected.

A common critique about this paradigm lies in that as the scene is noisy, the accumulation of "evidences" contributed by random noises can possibly result in false alarms. Grimson and Huttenlocher [22] give a formal analysis of the likelihood of false positive responses of the generalized Hough transform for object recognition. However, we can use this paradigm as an early stage of processing (i.e., as a filter), followed by a scrutinized verification stage.

2.1.4 Consistency Checking and Constraint Propagation

Many model-based vision schemes are based on searching the set of possible interpretations, which is usually combinatorially large.

As in other areas of artificial intelligence, making use of large amounts of world knowledge can often lead not only to increased robustness but also to a reduction in the search pace that must be explored during the process of interpretation. It is usually possible to analyze a number of constraints or consistency conditions that must be satisfied to make a correct interpretation. Effective application of consistency checking or propagation of constraints during searching can often prune the search space greatly.

Lowe [41] emphasizes the importance of viewpoint consistency constraint, which requires that the locations of all object features in an image be consistent with the projection from a single viewpoint. The application of this constraint allows the spatial information in an image to be compared with prior knowledge of an object's shape to the full degree of available image resolution. Lowe also argues that viewpoint-consistency plays a central role in most instances of human visual recognition.

Grimson [20] extended his previous work RAF [23] to handle some classes of parameterized objects. He approaches the recognition problem as a searching problem using the

so-called *interpretation tree*. Since this search is inherently an exponential process, he analyzed a set of geometric constraints based on the local shape of parts of objects to prune large subtrees from consideration without having to explore them.

The relaxation labeling technique (e.g. [51,54]) is yet another example of this type. Many visual recognition problems can be viewed as constraint-satisfaction problems. For example, when labeling a block-world picture, a relaxation algorithm iteratively assigns values to mutually constrained objects using local information alone in such a way that ensures a consistent set of values for which no constraints are violated. The values assigned to the objects by relaxation algorithms can be discrete or probabilistic. In the former case, discrete levels are assigned to objects, while in the latter case, a level of certainty (or probability) is attached to each label. Relaxation algorithms can proceed in a parallel-iterative manner, propagating matching constraints in parallel.

2.1.5 Sub-Graph Matching

Both object models and scene images can be expressed by graphs of nodes and arcs. Nodes represent features (usually geometric structures) detected and arcs represent the geometric relations between these features, e.g. [11]. The task then reduces to find an *embedding* or *fitting* of a model in a description of the scene.

Since a model contains both local and relational features in the form of a graph, matching depends not only on the presence of particular boundary features but also on their interrelations (e.g. distance). The requirement for successful recognition is that a sufficient set of key local features have to be visible and in correct relative positions, which may allow for a specified amount of distortion. For example, Bolles and Cain [10] proposed the use of a hierarchy of local focus features for model representation. Once the local focus features have been detected, nearby features are predicted and searched for in the scene. If the best focus features are occluded, the second best focus features are used.

An advantage of this technique is its robustness to relative occlusion and distortion. Computational inefficiency can be a drawback, since graph-matching or subgraph isomorphism procedures have to be executed. Effective strategies to reduce the time complexity are therefore required. In [10], once an occurrence of the best focus feature is located, the system simply runs down the appropriate list of nearby features and a transformation is computed, followed by a verification step. Thus exploration of the whole relational graph

is avoided. Barrow and Tenenbaum [6] proposes use of a hierarchical graph-searching technique which decomposes the model into independent components.

2.1.6 Evidential Reasoning

Evidential reasoning is a method for combining information from different sources of evidence to update probabilistic expectations. The attractive feature of using evidential reasoning for computer vision is that it allows us to combine information of varying reliability from many sources, even though no particular item of evidence is by itself strong enough for recognizing a particular object.

For example, SCERPO by Lowe [40] is a search-based matching system. As to the search space of SCERPO, two major components are involved: (1) the space of possible viewpoints; (2) the space of selecting one model from the model database. Lowe tackles the first component by perceptual organization and suggests that the second component be treated by a technique of evidential reasoning. Each piece of evidence contributes to the presence of a certain object or objects, which quantitatively can be described by a probabilistic expectation. The combining of evidences thus can be quantitatively described by combining probabilistic expectations. The probability ranking is constantly updated to reflect new evidence found. For example, as soon as one object is recognized, it provides contextual information which updates the rankings and aids in the search process. Ranking and updating may take time; however, as the list of possible objects increases, cost of the evidential reasoning may be amortized and its use becomes more important.

To sum up, evidential reasoning may deserve serious consideration to be incorporated into a system under development. It shows promise for carrying out the objective of combining many sources of information, including color, texture, shape and prior knowledge in a flexible way to achieve recognition.

2.1.7 Miscellaneous

Dual Models

If partial visibility problem is to be attacked, a recognition system using global features alone can not be feasible. Only local features can be used in the matching procedure in order to cope with overlapping parts and occlusions. One is free to choose different local

features such as points, lines, curves, etc., or their combinations as the matching features, according to the nature of the objects in the model base. However, two factors of representation, *completeness* and *efficiency*, have to be taken into consideration. Completeness of representation suggests rich enough features in order to enable discrimination between similar objects; while efficiency suggests using some minimal characteristic information to match models again the scene (so that the inherent complexity of the matching algorithm will be small).

A main drawback of representation by local features can be its incompleteness, since objects usually can not be fully described by a set of local features alone. One way to cope with this problem is to use both global and local feature representations: Local features are used in the matching procedure; while the additional complete representation is used, after the matching process, for verification and further refinement of the object localization [4,10,19,31,36].

Dual Scenes

Object recognition algorithms usually fall into two categories – those using intensity images and those using range data. However, we may use both of them as long as it helps.

For example, Kishon [34] combines the use of both range and intensity data to extract 3-D curves from a scene. These curves will be of a much higher quality than if they were extracted from the range data alone. The combined information is also used to classify the 3-D curves as either shadow, occluding, fold or painted curves.

Abstracted Features

An abstracted feature is a feature which does not physically exist, but is inferred. For example, in preparing a model representation for polygonal objects, we may extend non-neighboring edges to obtain their intersection and use the contained angle as a feature. Some more advanced applications of using abstracted features include the *footprint* of concavity [37], the *Fourier Descriptor* used for silhouettes description of aircraft in [2] and the representation of turning angle as a function of arc length used in [3,19,49].

Normalization

There are various reasons for the use of normalization.

In some cases, it is used to preserve certain properties. For example, in the *probabilistic relaxation labeling* scheme, the accumulated contributions from neighboring points have to be normalized so that the updating rule results in a probability (i.e., a value in [0..1]).

In other cases, it can be viewed as a technique for removing some uncertain factors. For example, suppose we use Fourier Descriptor to describe the boundary of an object. In order for such representation insensitive to the variations as changes in size, rotation, translation and so on, we have to perform the normalization operations such that the contour has a "standard" size, orientation and starting point. By normalizing the Fourier component $F(1)$ to have unity magnitude, we remove the factor of scaling variation. Similarly, the (s-θ) graph (the representation of turning angle as a function of arc length) has to be shifted vertically by adding an offset to θ so that the reference point on the contour is somewhat a standard value, such as zero. Since the rotation of a contour in Cartesian space corresponds to a simple shift in the ordinate (θ) of the s-θ graph, such normalization process removes the factor of rotation [19].

Other good reasons to use normalization involve making some operations easier to perform. For example, in performing point set matching, Hong and Tan [27] normalize both point sets to their canonical forms first; then, their canonical forms are matched. After normalization, the matching between two canonical forms reduce to a simple rotation of one to match the other.

2.2 An Overview of Geometric Hashing

2.2.1 A Brief Description

The geometric hashing idea has its origins in work of Schwartz and Sharir [49]. Application of the geometric hashing idea for model-based visual recognition was described by Lamdan, Schwartz and Wolfson. This section outlines the method; a more complete description can be found in Lamdan's dissertation [36]; a parallel implementation can be found in [46].

The geometric hashing method proceeds in two stages: a preprocessing stage and a recognition stage. In the preprocessing stage, we construct a model representation by

computing and storing redundant, transformation-invariant model information in a hash table. During the subsequent recognition stage the same invariants are computed from features in a scene and used as indexing keys to retrieve from the hash table the possible matches with the model features. If a model's features scores sufficiently many *hits*, we hypothesize the existence of an instance of that model in the scene.

The Pre-processing Stage

Models are processed one by one. New models added to the model base can be processed and encoded into the hash table independently. For each model M and for every feasible basis b consisting of k points (k depends on the transformations the model objects undergo during formation of the class of images to be analyzed), we

(i) compute the invariants of all the remaining points in terms of the basis b;

(ii) use the computed invariants to index the hash table entries, in each of which we record a node (M, b).

Note that all feasible bases have to be used. In particular, all the permutations (up to $k!$) of the k inputs used to calculate the invariants have to be considered. The complexity of this stage is $O(m^{k+1})$ per model, where m is the number of points extracted from a model. However, since this stage is executed off-line, its complexity is of little significance.

The Recognition Stage

Given a scene with n feature points extracted, we

(i) choose a feasible set of k points as a basis b;

(ii) compute the invariants of all the remaining points in terms of this basis b;

(iii) use each computed invariant to index the hash table and hit all (M_i, b_j)'s that are stored in the entry retrieved;

(iv) histogram all (M_i, b_j)'s with the number of hits received;

(v) establish a hypothesis of the existence of an instance of model M_i in the scene if (M_i, b_j), for some j, peaks in the histogram with sufficiently many hits;

and repeat from step **(i)**, if all hypotheses established in step **(v)** fail verification.

The complexity of this stage is $O(n) + O(t)$ per probe, where n is the number of points extracted from the scene and t is the complexity of verifying an object instance.

2.2.2 Strengths and Weaknesses

At a glance, geometric hashing seems similar to the transformation accumulation techniques discussed in section 2.1.3. However, the similarity lies only in the use of "accumulating evidence" by voting. The techniques discussed in section 2.1.3 accumulate "pose" evidence while geometric hashing accumulates "feature correspondence" evidence. The former analysis always votes for parameters of transformations while the latter votes for (model identifier, basis set) pair, where transformation parameters can be computed when the correspondence of scene feature set and model basis set is hypothesized.

Strengths

Most of the methods discussed in section 2.1 are search-based: Model features are searched to match scene features and this search process goes through each model in the model base sequentially. For example, the interpretation tree technique by Grimson [23] has inherent exponential complexity by pairing each scene feature to each model feature combinatorially. Geometric hashing greatly accelerates search of the model base by using a hashing technique to select candidate model features to match scene features. This process is at worst sublinear in the size of the model base.

Like the transformation accumulation techniques, geometric hashing's voting scheme copes well with occlusion and with the fragility of existing image segmentation techniques. The accumulation of evidence without regard to order makes parallel implementation easy.

Weaknesses

Errors in feature extraction will commonly lead to perturbation of invariants and degradation of recognition performance, since perturbed invariant values are used to index the

hash table to retrieve candidate model features. Performance is similarly degraded by quantization noise introduced in constructing the hash table. Thus use of point features does not seem viable in a seriously degraded image, see [21].

The inherently non-uniform distribution of computed invariants in the hash table results in non-uniform hash bin occupancy for uniformly quantized hash table [47]. This degrades the index selectivity power of the hashing scheme used.

2.2.3 Geometric Hashing Systems

Since the introduction of the geometric hashing method by Wolfson and Lamdan, a number of subsequent applications and improvements have been developed at the Robotics Research Laboratory of New York University and other research laboratories.

Gavrila and Groen [18] apply the hashing technique for recognition of 3-D objects from single 2-D images. They determine limits of discriminability by experiments to generate viewpoint-centered models.

Gueziec and Ayache [24] address the problem of fast rigid matching of 3-D curves in medical images. They incorporate six differential invariants associated with the surface and introduce an original table, where they hash values for the six transformation parameters.

Hummel and Wolfson [29] discuss both the object matching and the curve matching by geometric hashing.

Flynn and Jain [17] use invariant feature and the hashing technique to generate hypotheses without resorting to a voting procedure. They conclude that this method is more efficient than a constrained search technique, for example, interpretation tree technique [23].

Recently, the work of Rigoutsos develops geometric hashing by incorporating a Bayesian model for weighted voting, which we describe in the next section.

2.3 The Bayesian Model of Rigoutsos

The thesis work of Rigoutsos [43] from 1992 considerably extends the capability of the geometric hashing scheme.

To cope with the problem caused by the positional uncertainty of point features, a Gaussian distribution is used to model the perturbation of the coordinate of the point. That

is, the measured values are assumed to be distributed according to a Gaussian distribution centered at the true values and having standard deviation σ. More precisely, let (x_i, y_i) be the "true" location of the i-th feature point in the scene. Let also (X_i, Y_i) be the continuous random variables denoting the coordinates of the i-th feature. The joint probability density function of X_i and Y_i is then given by:

$$f(X_i, Y_i) = \frac{1}{2\pi\sigma^2} \exp(-\frac{(X_i - x_i)^2 + (Y_i - y_i)^2}{2\sigma^2}).$$

Using this error model, he formulates a weighted voting scheme for the evidence accumulation in geometric hashing. He uses a weighted contribution to the model/basis hypothesis of an entry ζ in the hash space based on a scene point's hash ξ in the same space, using a formula of the form

$$\log[1 - c + A \exp(-\frac{1}{2}(\xi - \zeta)\Sigma^{-1}(\xi - \zeta)^t)],$$

where c and A are constants that depend on the scene density and Σ is the covariance matrix in hash space coordinates of the expected distribution of ξ based on the error model for (x, y), the feature point in the scene.

He shows that when weighted voting is used according to the above formula, the evidence accumulation can be interpreted as a Bayesian *a posteriori* classification scheme. He shows that the accumulations are related to

$$\log[Prob(H_k|E_1, \cdots, E_n)]$$

where the hypothesis H_k represents the proposition that a certain model/basis occurs in the scene and matches the chosen basis, and E_i's are pieces of evidence given by scene invariants.

The above formulation is implemented on a $8K$-processor Connection Machine (CM-2) and can recognize objects that have undergone a similarity transformation, from a library of 32 models. The models used for experiments are military aircraft and production automobiles. Features are extracted by using the Boie-Cox edge detector [9] and locating the points of high curvature. The features are the coordinate pairs of these points [44].

In this thesis, we also make use of a Bayesian *a posteriori* maximum likelihood recognition scheme based on geometric hashing. We use somewhat simplified versions of the formulae given by Rigoutsos. We make use of line features as opposed to point features.

However, we also assume a Gaussian perturbation model of the feature values, which in our case implies an independent perturbation of the (θ, r) variables. Since no reliable segmentation is assumed to be available, we extracted our line features by means of an improved Hough transform technique that operates on seriously degraded environments. Objects modeled by lines that have undergone affine transformations are used for experiments. A viewpoint consistency constraint is also applied to further filter out false alarms.

Chapter 3

Noise in the Hough Transform

This chapter overviews the effect of noise on the Hough transform, which is used to detect line features from an edge map.

We use a series of simulations to estimate the *reliability* and *accuracy* of the Hough technique. By reliability we mean its ability to detect lines, while coping with occlusions and additive noise; by accuracy we mean the deviation of detected line parameters from their true values. Although much has been published in this question and theoretical analyses have been given (see [22]), important questions on these two issues remains open and simulation remains a revealing technique.

In section 3.1, we briefly review the history and technique of the Hough transform. Section 3.2 describes the factors that affect the performance of the Hough transform and the way it can be improved. A series of simulations on images, covering a range of line lengths, line orientations and line positions under increasing noise level (both positive and negative noise) have been performed. Section 3.3 shows our experimental results, which are summarized in section 3.4.

3.1 The Hough Transform

The well-known Hough technique was introduced by Paul Hough in a US patent filed in 1962. Hough's initial application was analysis of bubble chamber photographs of particle tracks; such images contain a large amount of noise. Hough proposed to use amplifiers, delays, signal generators, and so on to perform what we now call the Hough transform in

an analog form.

A common digital implementation of the Hough transform applies the *normal parameterization* suggested by Duda and Hart [15] in the form

$$x \cos\theta + y \sin\theta = r,$$

where r is the perpendicular distance of the line to the origin and θ is the angle between a normal to the line and the positive x axis.

This normal parameterization has several advantages: r and θ vary uniformly as the line orientation and position change, and neither goes to infinity as the line becomes horizontal or vertical. It is easy to see that points on a particular line will all map to sinusoids that intersect in a common point in Hough space and the coordinate (θ, r) of that intersection point gives the parameters of the line.

In standard implementations of the Hough method, Hough space is quantized. Each (x_i, y_i) is mapped to a sampled, quantized sinusoid. The detailed algorithm is as follows:

1. Quantize the 2-D Hough space (i.e., parameter space) between 0° and 180° for θ and $-\frac{1}{\sqrt{2}}N$ to $+\frac{1}{\sqrt{2}}N$ for r ($N \times N$ is the image size).

2. Form an accumulator array $A[\theta][r]$, whose buckets are initialized to 0.

3. For each edgel (x, y) in the image, increment all buckets in the accumulator array along the appropriate sinusoid, i.e.,

   $$A[\theta][r] := A[\theta][r] + 1$$

   for θ and r satisfying $r = x \cos\theta + y \sin\theta$ within the limits of the quantization.

4. Locate, in the accumulator array, local maxima, which correspond to collinear edgels in the image (the values of the accumulator array provide a measure of the number of edgels on the line).

We note that due to the quantization of Hough space and noise in the image, sinusoids generated by points of the same line do not in general intersect precisely at a common point in the Hough space. Much literature has addressed this problem by suggesting additional processing (see [32] for a survey).

3.2 Various Hough Transform Improvements

Since the above straightforward implementation of the Hough transform needs improvement in highly degraded environments [22], we have investigated the factors affecting the performance of the technique and implemented and experimented on a series of variations of the Hough transform.

Factors that adversely affect the performance (reliability and accuracy) of the Hough transform include:

- Short lines inherently result in low peaks in Hough space $H(\theta, r)$, which prevents the detection of short lines relative to long lines.
- When $r = x\cos\theta + y\sin\theta$ is rounded to pick a particular bucket in $H(\theta, r)$ for a specific θ, fractional r information is lost and all subsequent computations are done on the rounded r values.
- θ is also sampled. If a line has an orientation θ', not exactly equal to any sampled θ, the points on the line, when mapped to Hough space, will not have identical r values. Instead, their r values will spread out near the real r value.
- Using the coordinate (θ, r) of the peak in Hough space as the line parameter limits the precision of the parameters detected, since Hough space is quantized.

To strengthen the low peaks in Hough space which result from short lines present in a source image (compared to the inherently high peaks which result from long lines), we apply *background subtraction* by removing the contribution made to Hough space by edgels along a long line after this long line has been detected. This method is attributed to Risse [48]. To implement the idea, a global maximum in Hough space, say bucket (θ, r), is located; then its corresponding line in the image is identified and all accumulator buckets which were incremented during processing edgels lying along this line are decremented. This algorithm differs from the algorithm described in the previous section in that instead of detecting local maxima in the accumulator array as candidate lines in the image, it detects lines one after one iteratively as follows:

while still more lines to be detected **do**
    $(\hat{\theta}, \hat{r})$:= globalMax(accumulator A);

for each edgel (x, y) on line $x\cos\hat{\theta} + y\sin\hat{\theta} = \hat{r}$ **do**
$A[\theta][r] := A[\theta][r] - 1$, for θ and r satisfying $r = x\cos\theta + y\sin\theta$
end for
end while

Starting with this algorithm as the backbone, we improve it with weighted voting to compensate the rounding error of r and achieve sub-bucket precision computing, as described below.

To compensate for the rounding error of r, we distribute the Hough vote-increment value (typically 1) over a region in Hough space. For example, suppose $r_{exact} = x\cos\theta + y\sin\theta$ (i.e., before rounding) and r_a and r_b are consecutive values of sampled r in the Hough space such that $r_a \leq r_{exact} < r_b$. Then we increment $H(r_a, \theta)$ by an amount proportional to $r_b - r_{exact}$ and increment $H(r_b, \theta)$ by an amount proportional to $r_{exact} - r_a$. That is, we distribute the increments linearly in two related buckets.

To compensate for the spreading of votes caused by the quantization of θ and r, we find the bucket (θ, r) receiving the maximum number of votes, using a small window (3 by 3 in our implementation) centered around this bucket and compute the center of mass over this small window to achieve sub-bucket precision in line parameter estimation.

Experiments on seriously degraded images show that this method is superior to other variations of the Hough transform in the sense that more true lines and fewer spurious lines are detected and estimated line parameters' accuracy is quite good.

In order to further reduce the number of spurious lines detected, we notice that the Hough transform detects lines in a very global way. More precisely, it counts the number of edgels which happen to lie on the same line, sparsely or densely, as the evidence of the existence of a line. This differs from human visual recognition, which exhibits grouping by proximity.

To further enhance our algorithm, we therefore use grouping by considering proximity as follows.

To detect n lines in an image, we

1. apply the algorithm described above to detect a pre-specified number, say $2n$, of lines;
2. for each line (θ_i, r_i) detected,

(a) go back to the image (edge map) and scan edgels (x, y)'s on the line with a 1-D sliding window of appropriate size (depending on the image resolution).

(b) if the number of edgels in the window is less than, say 1/3, of the window size, take (x, y) to be a noise edgel which happens to lie on the line (θ_i, r_i).

(c) re-accumulate the evidence support for the line without counting the noise edgels.

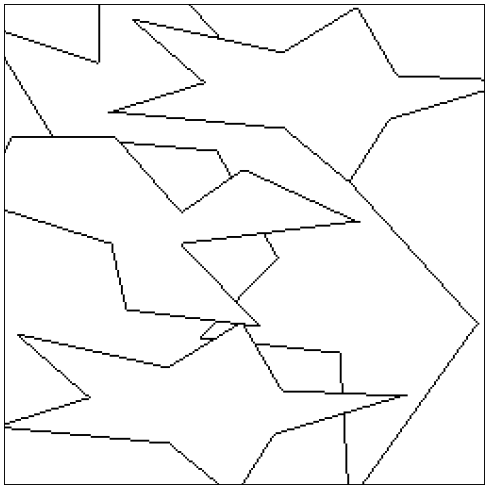
3. sort these lines by their new evidence support and select the top n from among them.

The reason we simply select the top n from among $2n$ lines detected by our previous algorithm is that our previous algorithm already has good performance, and we just want to further remove spurious lines.

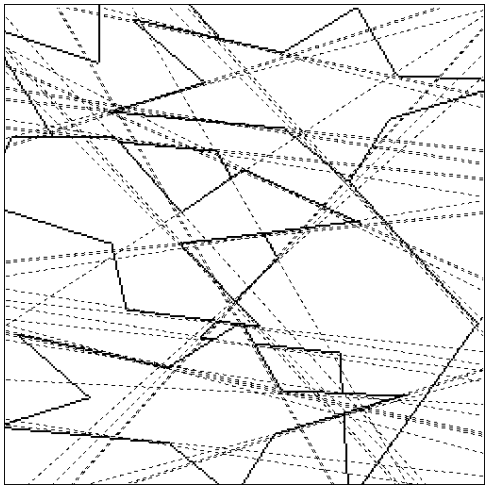
Figure 3.1 shows an example, comparing the result of the standard Hough technique and our improved Hough technique. Figure 3.1(a) shows a synthesized image without noise. It contains roughly 40 segments. Figure 3.1(b) shows the result of the standard Hough analysis. 50 lines are detected. 17 true lines are missing. Also many of the true lines are detected multiple times. Figure 3.1(c) shows the result by applying our improved Hough technique without enhancement of using proximity grouping. Also, 50 lines are detected. There is no missing true lines. Yet, since more lines are detected than are actually present in the source image, a few true lines are detected multiple times. Figure 3.1(d) shows the result by applying our improved Hough technique with proximity grouping enhancement. Figure 3.2(a) shows the same image as in Figure 3.1(a), yet with serious noise. Figure 3.2(b) to Figure 3.2(d) are the correspondences of Figure 3.1(b) to Figure 3.1(d) . The performance improvement over the standard Hough technique is much obvious.

3.3 Implementation and Measured Performance

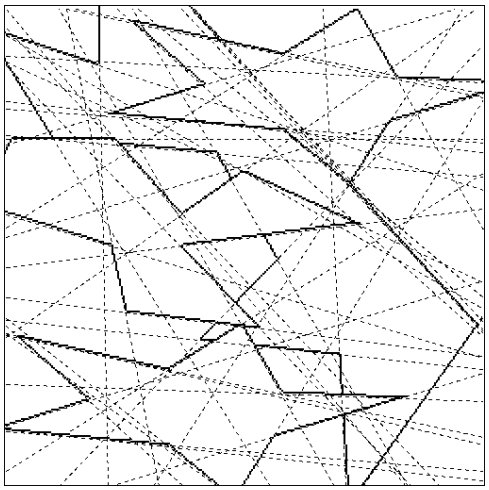
In our implementation, we use $\Delta\theta = 1$ (1 degree) and $\Delta r = 1$ (1 pixel) as the quantization values for θ and r in Hough space. This appears quite sufficient for our subsequent applications. Intuitively, smaller values of $\Delta\theta$ might give better precision but can result in the spreading of peaks (recall that image is digitized so that the edgels of a line will not lie



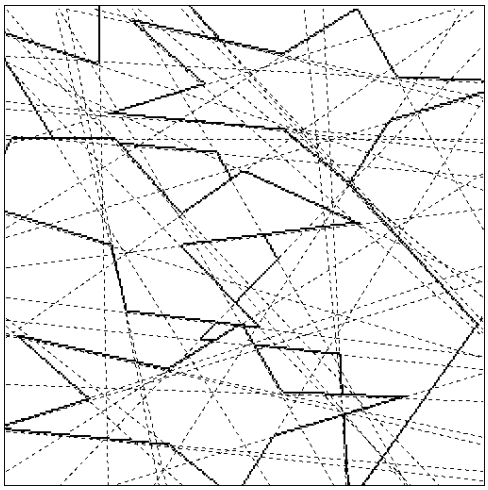
(a) A test image without noise. It contains roughly 40 segments.



(b) 50 lines are detected using the standard Hough transform.

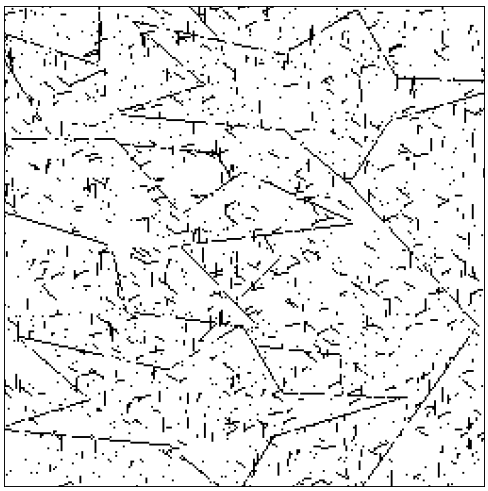


(c) 50 lines are detected using our improved Hough transform without proximity grouping.

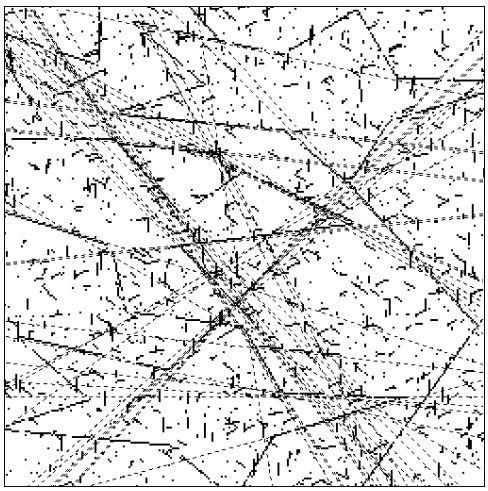


(d) 50 lines are detected using our improved Hough transform with proximity grouping.

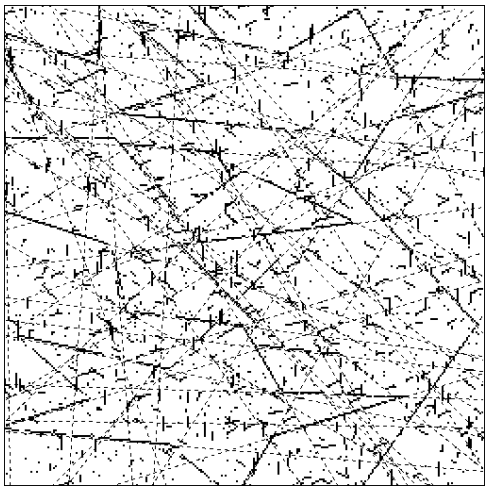
Figure 3.1: A comparison of the results of the standard Hough technique and our improved Hough technique, when applied to an image without noise.



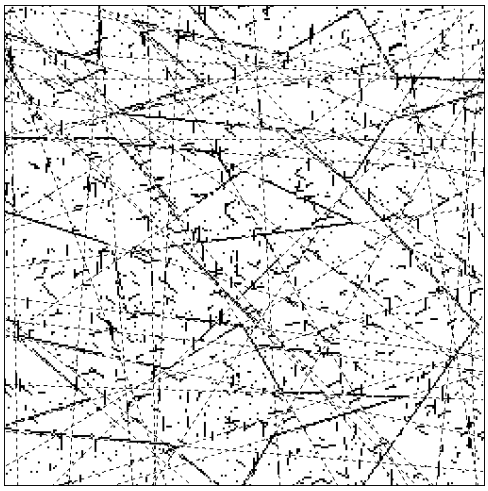
(a) A test image with serious noise. It contains roughly 40 segments.



(b) 50 lines are detected using the standard Hough transform.



(c) 50 lines are detected using our improved Hough transform without proximity grouping.



(d) 50 lines are detected using our improved Hough transform with proximity grouping.

Figure 3.2: A comparison of the results of the standard Hough technique and our improved Hough technique, when applied to a noisy image.

precisely on a digitized line except when the line has orientation of 0°, 45°, 90° or 135°). Similarly smaller values of Δr give better precision but result in the spreading of peaks.

We have performed a series of experiments on synthesized images containing polygonal objects, which are arbitrarily chosen from our model base (see Figure 3.3). (The same images are used in our later experiments.)

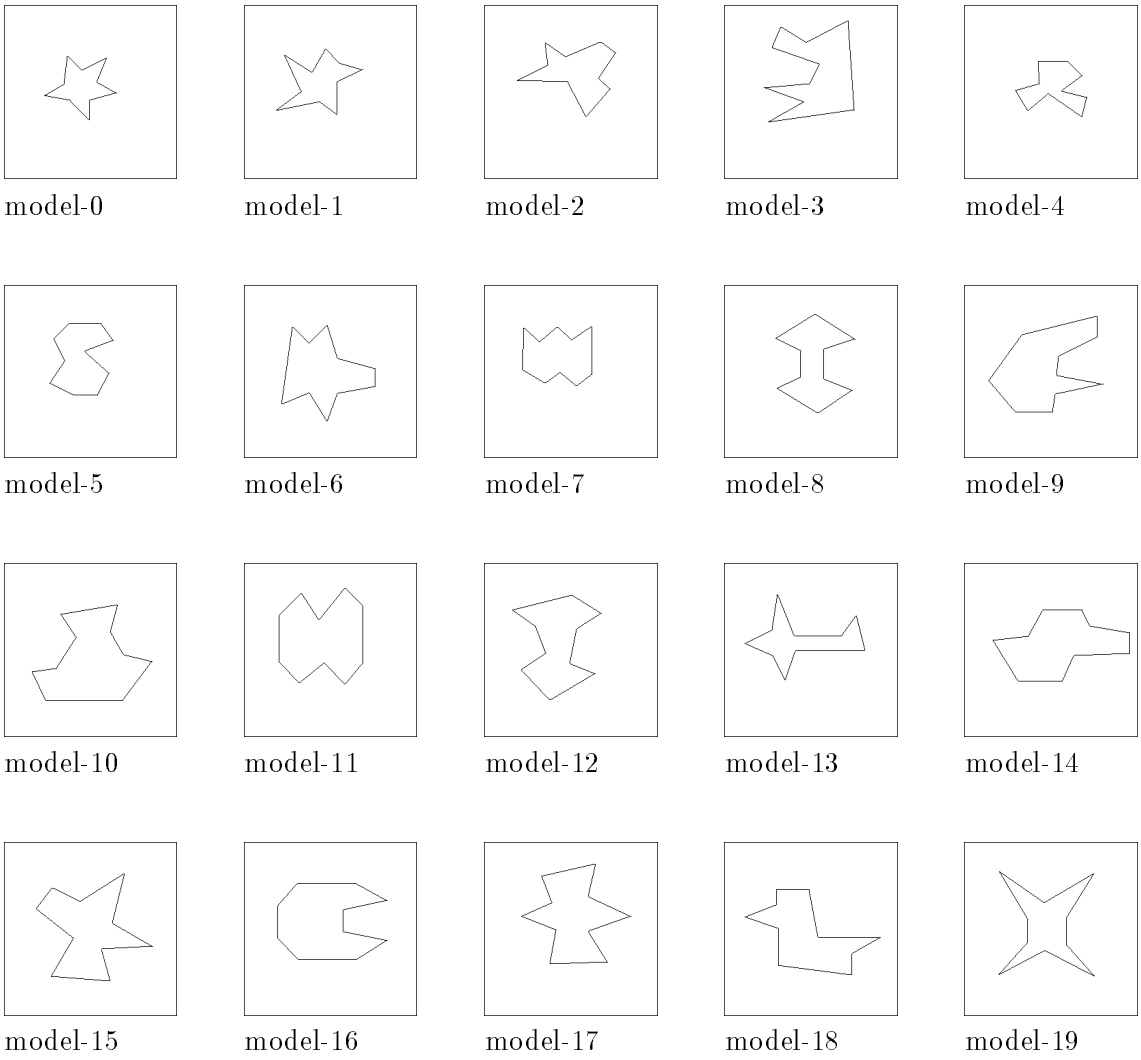


Figure 3.3: The twenty models used in our experiments. From left to right, top to bottom are model-0 to model-19.

Two kinds of noise are imposed in our experiments: positive and negative.

We generate negative noise by randomly erasing edgels from the object boundary. For example, if we want 10% of the boundary to be missing (occluded), we scan along the boundary of every object and erase the edgels whenever *ran* modulo 10 = 0, where *ran* is a random number; similarly, if we want 20% of the boundary to be missing, we do the same whenever *ran* modulo 5 = 0.

The positive noise we generate consists of random dot noise and segment noise. Random dot noise is an additive dot noise whose position in the image is randomly selected; segment noise is added by randomly selecting a position in the image and an orientation (from 0° to 360°) for the segment then putting a line segment of length varying randomly from 0 to 7 pixels.

Simulations consider different combinations of noise levels. The image size used in our experiments is 256 × 256. Figure 3.4 shows a sample scene (image-0) after application of various noise levels. Other test images are shown in Figure 3.5. The same noise levels were applied to them.

In the following, we show the result of experiments on ten images. The number of lines we select is 70. Since more lines are detected than are actually present in the source image, our experiments allow for spurious lines.

The following abbreviated table shows lines by the ranks of their votes, using "*" to represent lines actually present in the source image and "." to represent spurious lines. Standard deviations of the θ and r values and their covariances are also given; this data is of significance for design of the later recognition stage.

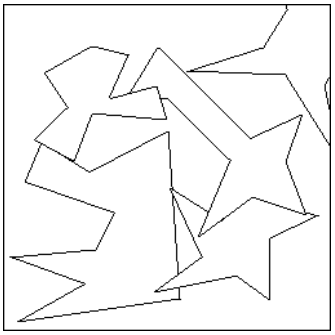
- **case 0**: no noise imposed on the image

```
image-0: ************************************************.****.*..***.........
image-1: ******************************.******************.**...*..*....**....
image-2: ********************************************************.*..**.*..*...
image-3: ********************************************************.....*.......*....
image-4: ****************.**********.****************.*******...*..**.........
image-5: ************************************************.***.*..........*..*.*.
image-6: ********************************************************.*.*..*....
image-7: ************************************************.****.........
image-8: ************************************************.**.*..*.....*...*.
image-9: ********************************************..***********.*.....*..*
```

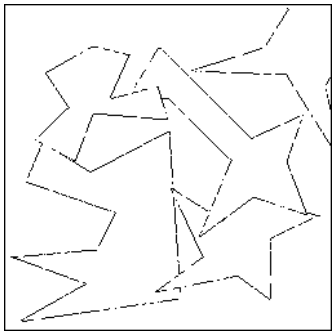
σ_θ = 1.398227 (degree) or 0.024404 (radian); σ_r = 1.672734 (pixel);

$\sigma_{\theta r}$ = 0.189464 (degree.pixel) or 0.003307 (radian.pixel).

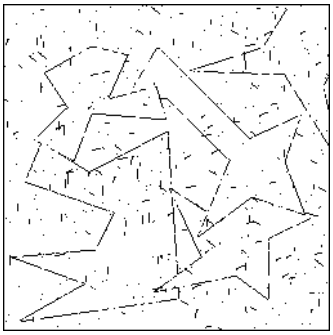
- **case 1**: 10% negative noise, no positive noise



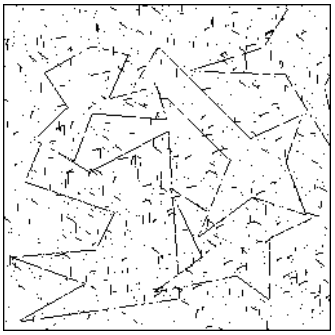
(0) Image-0 with no noise



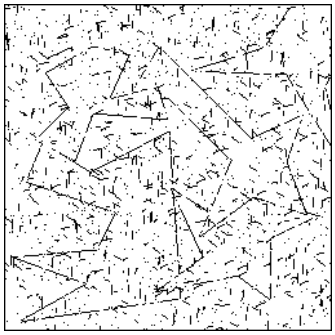
(1) Image-0 with 10% negative noise and no positive noise



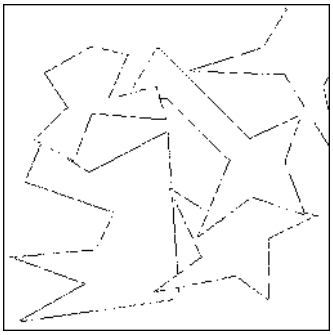
(2) Image-0 with 10% negative noise and 200 pieces of positive noise



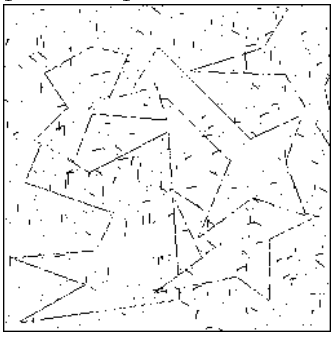
(3) Image-0 with 10% negative noise and 400 pieces of positive noise



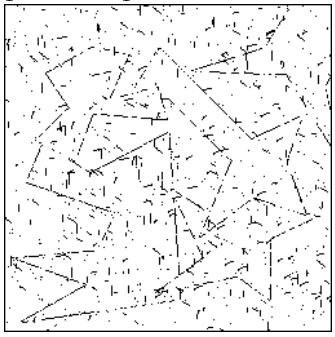
(4) Image-0 with 10% negative noise and 800 pieces of positive noise



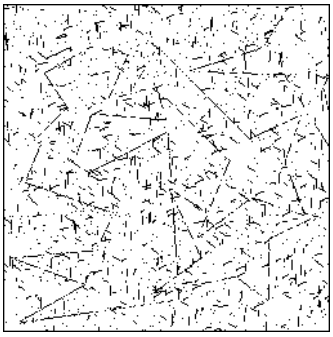
(5) Image-0 with 20% negative noise and no positive noise



(6) Image-0 with 20% negative noise and 200 pieces of positive noise



(7) Image-0 with 20% negative noise and 400 pieces of positive noise



(8) Image-0 with 20% negative noise and 800 pieces of positive noise

Figure 3.4: Reading from top to bottom, left to right, (0) to (8) are image-0 with different noise levels corresponding to cases 0 to 8.

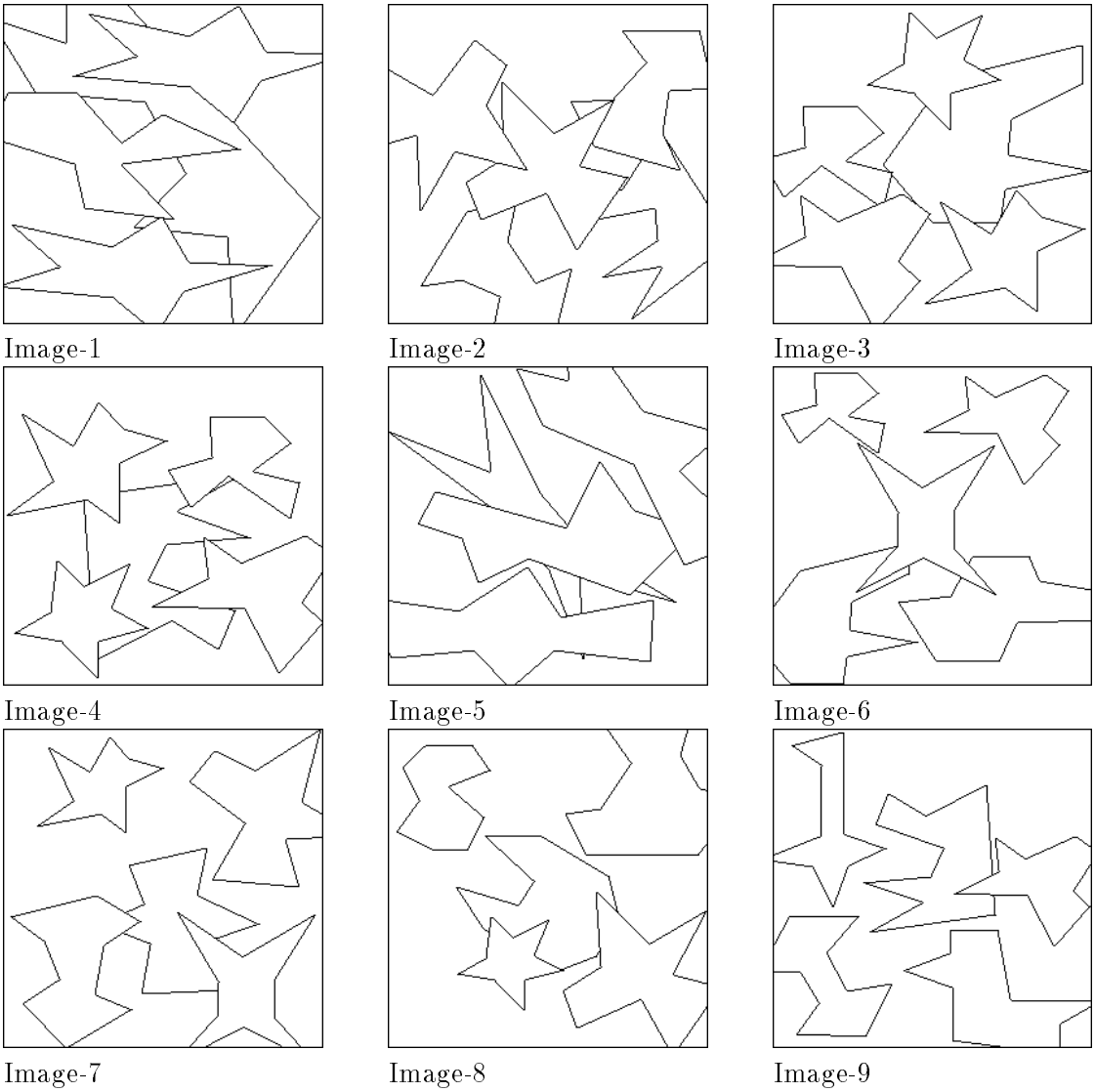
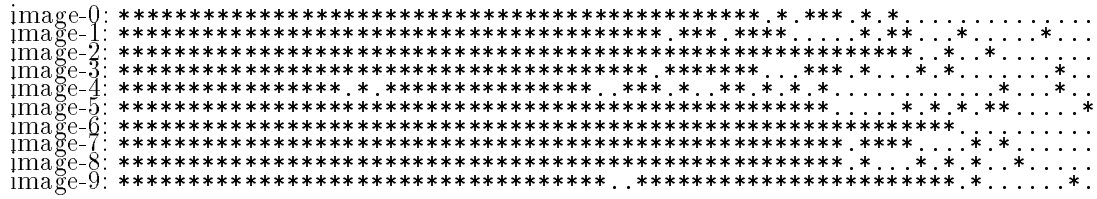


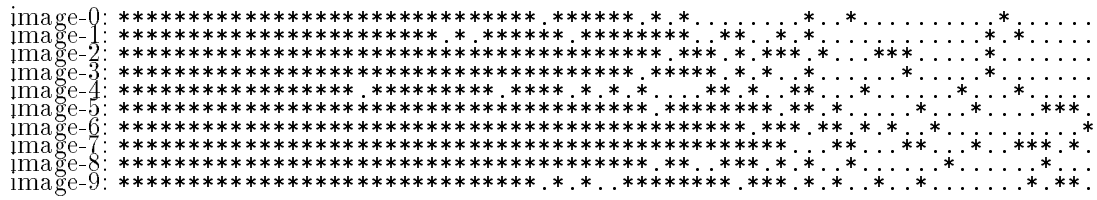
Figure 3.5: Image-1 to Image-9 before noise is imposed. The same noise levels as for Image-0 are applied during our experiments.



σ_θ = 1.496886 (degree) or 0.026126 (radian); σ_r = 1.803655 (pixel);

$\sigma_{\theta r}$ = 0.244846 (degree.pixel) or 0.004273 (radian.pixel).

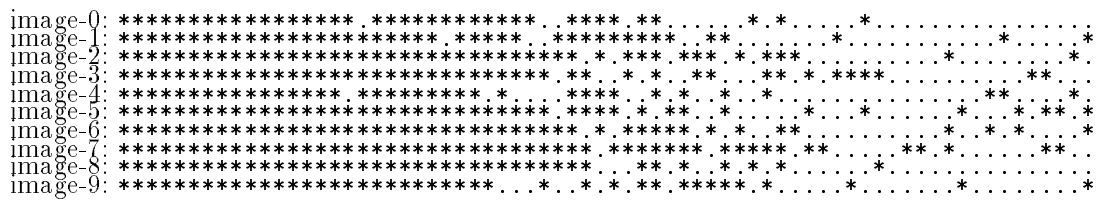
- **case 2**: 10% negative noise, along with 200 pieces of random dot noise and 200 pieces of segment noise



σ_θ = 1.656363 (degree) or 0.028909 (radian); σ_r = 2.059478 (pixel);

$\sigma_{\theta r}$ = 0.119053 (degree.pixel) or 0.002078 (radian.pixel).

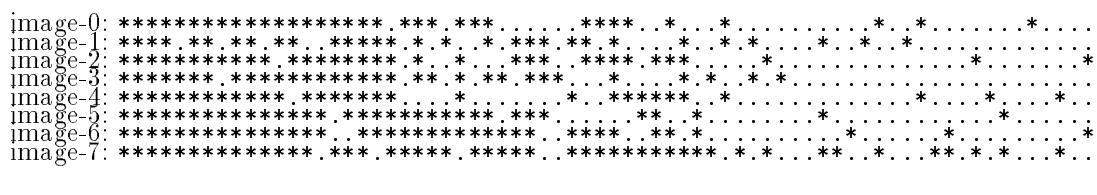
- **case 3**: 10% negative noise, along with 400 pieces of random dot noise and 400 pieces of segment noise



σ_θ = 1.712783 (degree) or 0.029894 (radian); σ_r = 2.017484 (pixel);

$\sigma_{\theta r}$ = 0.217823 (degree.pixel) or 0.003802 (radian.pixel).

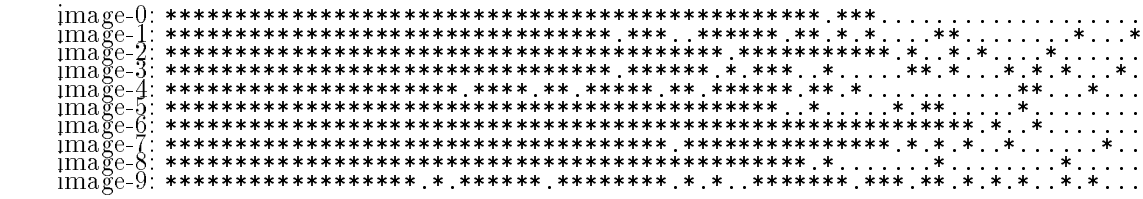
- **case 4**: 10% negative noise, along with 800 pieces of random dot noise and 800 pieces of segment noise



σ_θ = 1.938883 (degree) or 0.033840 (radian); σ_r = 2.519878 (pixel);

$\sigma_{\theta r}$ = -0.357589 (degree.pixel) or -0.006241 (radian.pixel).

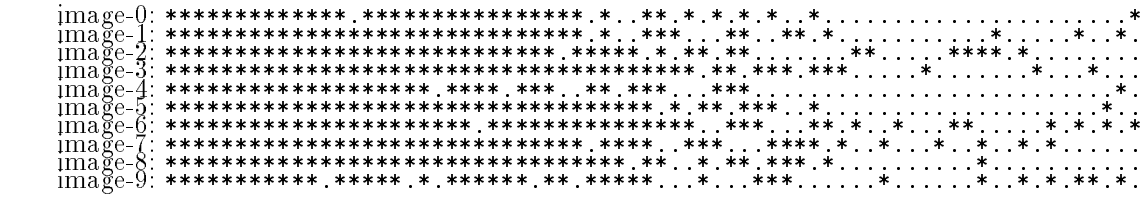
- **case 5**: 20% negative noise, along with no positive noise



σ_θ = 1.563745 (degree) or 0.027293 (radian); σ_r = 1.913004 (pixel);

$\sigma_{\theta r}$ = 0.219434 (degree.pixel) or 0.003830 (radian.pixel).

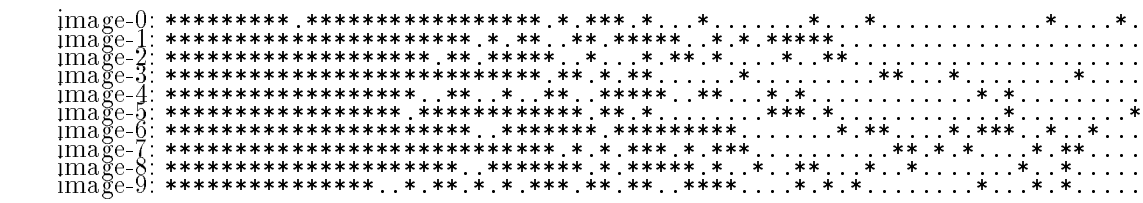
- **case 6**: 20% negative noise, along with 200 pieces of random dot noise and 200 pieces of segment noise



σ_θ = 1.754688 (degree) or 0.030625 (radian); σ_r = 2.144938 (pixel);

$\sigma_{\theta r}$ = 0.349338 (degree.pixel) or 0.006097 (radian.pixel).

- **case 7**: 20% negative noise, along with 400 pieces of random dot noise and 400 pieces of segment noise

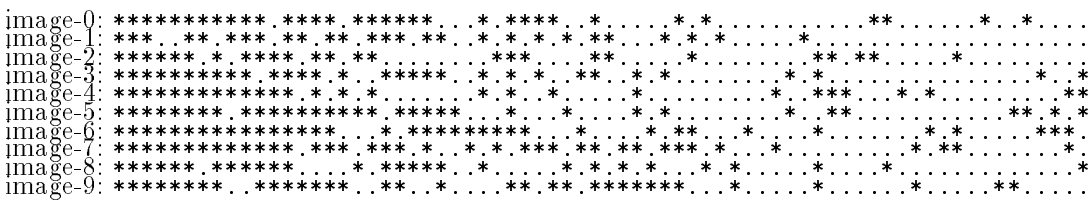


σ_θ = 1.880869 (degree) or 0.032827 (radian); σ_r = 2.397261 (pixel);

$\sigma_{\theta r}$ = -0.113044 (degree.pixel) or -0.001973 (radian.pixel).

- **case 8**: 20% negative noise, along with 800 pieces of random dot noise and 800 pieces

of segment noise

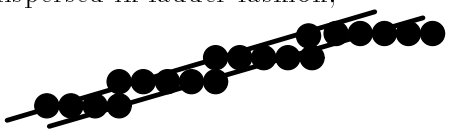


σ_θ = 2.168774 (degree) or 0.037852 (radian); σ_r = 2.503542 (pixel);

$\sigma_{\theta r}$ = -0.396545 (degree.pixel) or -0.006921 (radian.pixel).

3.4 Some Additional Observations Concerning the Accuracy of Hough Data

For the case when there is no noise imposed, we can see that true lines dominate the front of the list (those with high votes) and spurious lines are at the rear of the list. Note that some of the true lines are detected multiply. This can happen when a line is slanting, because its edgels become dispersed in ladder fashion,



and two parallel lines are often detected.

Negative noise is more devastating to the Hough transform than positive noise. Positive noise, unless very intense, does not make true lines disappear; Negative noise subtracts signals and can seriously affect the accuracy of the line parameters detected. This is especially true of short lines with a small slope. For example, given negative noise the edge map

can turn into something like

Indeed, our experiments showed that most of the undetected lines are slanting lines with small slopes.

While background subtraction works well to remove bias in favor of long line segments over short line segments, it in fact implicitly applies the principle of the-winner-takes-all. In particular, when a long line segment intersects a short line segment, their intersection pixel will be classified as belonging to the long line segment and *subtracted* from the image after the long line segment is detected. However, when these two line segments do not intersect each other even though their extended lines do, the perceptual grouping technique we use reduces this adverse effect.

In regard to accuracy, we observe that line length is the key role affecting the accuracy of Hough transform results. The Hough transform works well to detect long line segments even if the image is quite noisy. As line length decreases, noise affects the accuracy of the Hough transform even more seriously.

When the center of a segment is far away from the projection of the origin onto the extended line of the segment, the average error of r increases. This phenomenon can be rationalized as follows (see Figure 3.6): Since our algorithm loops through θ, we in fact consider a line through the origin with slope angle θ (i.e., with line parameter $(\theta + 90°, 0)$) and project all the image edgels onto this line. If bucket (θ, r) in Hough space is detected with n votes, roughly n edgels are projected onto the line at positions roughly r distant from the origin. With this viewpoint in mind, due to the quantization of images, if the center of a line segment, with line parameter (θ, r), is far away from the intersection point of line (θ, r) and line $(\theta + 90°, 0)$, the projections (onto line $(\theta + 90°, 0)$) of the points of that line segment disperse (on line $(\theta + 90°, 0)$) around the position r distant from the origin (except when the orientation of the line segment is 0°, 45°, 90° or 135°.)

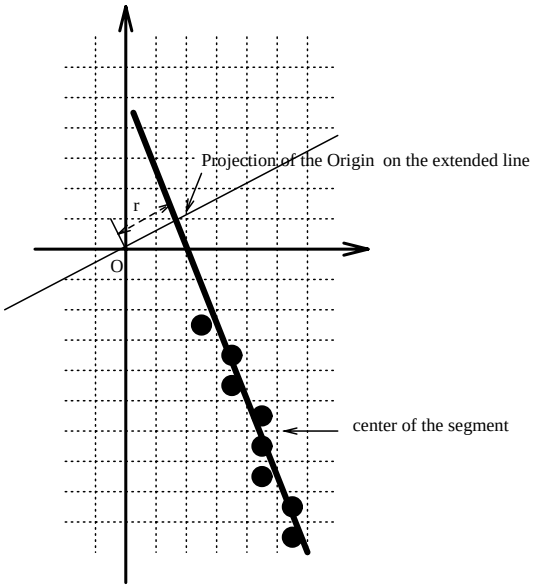


Figure 3.6: The thicker line is with parameter (θ, r). Projections of the pixels of this line onto line $(\theta + 90°, r)$ will disperse around the position r distant from the origin.

Chapter 4

Invariant Matching Using Line Features

The idea behind the geometric hashing method is to encode local geometric features in a manner which is invariant under the geometric transformation that model objects undergo during formation of the class of images being analyzed. This encoding can then be used in a hash function that makes possible fast retrieval of model features from the hash table, which can accordingly be viewed as an encoded model base. The technique can be applied directly to line features, without resorting to point features indirectly derived from lines, providing that we use some method of encoding line features in a way invariant under transformations considered.

Potentially relevant transformation groups include rigid transformations, similarity transformations, affine transformations and projective transformations, depending on the manner in which an image is formed. Each of these classes of transformations is a subgroup of the full group of projective transformations. Since a projective transformation is a collineation (referring to p. 89 of [35]), all these transformations preserve lines. This allows us to use line features as inputs to the recognition procedures.

4.1 Change of Coordinates in Various Spaces

Since we will represent line features by their normal parameterization (θ, r), we show in this section how the change of coordinates in image space by a linear transformation acts

to change of coordinates in (θ, r)-space. This is preliminary to the invariant line encodings exploited in its following sections.

4.1.1 Change of Coordinates in Image Space

It is often easiest to work in homogeneous coordinates, since this allows projective transformations to be represented by matrices.

A change of coordinates in homogeneous coordinates is given by

$$\mathbf{w}' = \mathbf{T}\mathbf{w},$$

where $\mathbf{w} = (w_1, w_2, w_3)^t$ is the homogeneous coordinate before transformation, $\mathbf{w}' = (w_1', w_2', w_3')^t$ is the homogeneous coordinate after transformation and $\mathbf{T}$ is a non-singular 3×3 matrix. Note that $\mathbf{T}$ and $\lambda\mathbf{T}$, for any $\lambda \neq 0$, define the same transformation, since we are dealing with homogeneous coordinates.

A point $(x, y)^t$ in image space is represented by a non-zero 3-D point $\mathbf{w} = (w_1, w_2, w_3)^t$ in homogeneous coordinate systems, such that

$$x = \frac{w_1}{w_3} \quad \text{and} \quad y = \frac{w_2}{w_3},$$

where $w_3 \neq 0$. This representation is not unique, since $\lambda\mathbf{w}$, for any $\lambda \neq 0$ is also a homogeneous representation of $(x, y)^t$.

Every non-singular 3×3 matrix defines a 2-D projective transformation of homogeneous coordinates. Various subgroups of the projective transformation group are defined by different restrictions on the form of the matrix $\mathbf{T}$.

4.1.2 Change of Coordinates in Line Parameter Space

A line in a 2-D plane can be represented by its *parameter vector* and is usually parameterized as

$$a_1 w_1 + a_2 w_2 + a_3 w_3 = 0$$

or

$$\mathbf{a}^t \mathbf{w} = 0,$$

where $\mathbf{a} = (a_1, a_2, a_3)^t$ is the parameter vector of the line (note that a_1 and a_2 are not both equal to 0) and $\mathbf{w} = (w_1, w_2, w_3)^t$ is a homogeneous coordinate of any point on the line.

This representation is not unique, since $\lambda\mathbf{a}$, for any $\lambda \neq 0$, is also a representation of the same line.

A transformation $\mathbf{T}$ in image space changes the coordinate of every point $\mathbf{w}$ on a line to $\mathbf{w}'$ by $\mathbf{w}' = \mathbf{T}\mathbf{w}$. Substituting $\mathbf{w} = \mathbf{T}^{-1}\mathbf{w}'$ in $\mathbf{a}^t\mathbf{w} = 0$, we get

$$\mathbf{a}^t\mathbf{T}^{-1}\mathbf{w}' = 0$$

and hence

$$((\mathbf{T}^{-1})^t\mathbf{a})^t\mathbf{w}' = 0,$$

or

$$\mathbf{a}'^t\mathbf{w}' = 0, \text{ where } \mathbf{a}' = (\mathbf{T}^{-1})^t\mathbf{a}.$$

This shows that the change of the coordinate of the point in 3-D parameter space is given by

$$\mathbf{a}' = \mathbf{P}\mathbf{a}.$$

where $\mathbf{P} = (\mathbf{T}^{-1})^t$.

4.1.3 Change of Coordinates in (θ, r) Space

Line features of an image are usually extracted by the Hough transform. A common implementation of the Hough transform applies the normal parameterization suggested by Duda and Hart[15], in the form

$$x\ \cos\theta + y\ \sin\theta = r,$$

where r is the perpendicular distance of the line to the origin and θ is the angle between a normal to the line and the positive x-axis.

This unique parameterization of lines relates to the preceding parameterization of lines. Let $\mathbf{F} : \mathbf{R}^2 \mapsto \mathbf{R}^3$ be a mapping such that

$$\mathbf{F}((\theta, r)^t) = (\cos\theta, \sin\theta, -r)^t$$

If we restrict the domain of θ to be in $[0, \pi)$, then $\mathbf{F}^{-1}$ exists. Define another mapping $\mathbf{G} : \mathbf{R}^3 \mapsto \mathbf{R}^2$ by $\mathbf{F}^{-1}$ as

$$\mathbf{G}((a_1, a_2, a_3)^t) = \begin{cases} \mathbf{F}^{-1}((\frac{a_1}{\sqrt{a_1^2+a_2^2}}, \frac{a_2}{\sqrt{a_1^2+a_2^2}}, \frac{a_3}{\sqrt{a_1^2+a_2^2}})^t) & \text{if } a_2 \geq 0 \\ \mathbf{F}^{-1}((\frac{-a_1}{\sqrt{a_1^2+a_2^2}}, \frac{-a_2}{\sqrt{a_1^2+a_2^2}}, \frac{-a_3}{\sqrt{a_1^2+a_2^2}})^t) & \text{otherwise} \end{cases}$$

where a_1 and a_2 are not both equal to 0. Then

$$\mathbf{G}((a_1,a_2,a_3)^t) = \begin{cases} (\tan^{-1}(\frac{a_2}{a_1}), \frac{-a_3}{\sqrt{a_1^2+a_2^2}})^t & \text{if } a_2 > 0 \\ (0, \frac{-a_3}{a_1})^t & \text{if } a_2 = 0 \\ (\tan^{-1}(\frac{-a_2}{-a_1}), \frac{a_3}{\sqrt{a_1^2+a_2^2}})^t & \text{if } a_2 < 0 \end{cases}$$

where the range of $\tan^{-1}$ is in $[0..\pi)$ and $\tan^{-1}(\infty) = \frac{\pi}{2}$.

Then $\mathbf{G}$ maps a point $\mathbf{a} = (a_1, a_2, a_3)^t$ in parameter space, where a_1 and a_2 are not both equal to 0, to $(\theta, r)^t = \mathbf{G}(\mathbf{a})$ in (θ, r)-space such that

$$a_1 w_1 + a_2 w_2 + a_3 w_3 = 0$$

and

$$\cos\theta\, w_1 + \sin\theta\, w_2 - r\, w_3 = 0$$

define the same line.

Let $(\theta, r)^t$ be the (θ, r)-parameter defining a line (in fact, the line $x\cos\theta + y\sin\theta = r$). Then $\mathbf{F}((\theta, r)^t) = \mathbf{a}$, where $\mathbf{a} = (\cos\theta, \sin\theta, -r)^t$, is a point in parameter space. A transformation $\mathbf{P}$ changes the coordinate of $\mathbf{a}$ to $\mathbf{a}' = \mathbf{P}\mathbf{a}$ in parameter space. Substituting $\mathbf{F}((\theta, r)^t)$ for $\mathbf{a}$ in $\mathbf{a}' = \mathbf{P}\mathbf{a}$, we get

$$\mathbf{a}' = \mathbf{P}\mathbf{F}((\theta, r)^t)$$

and hence $(\theta', r')^t = \mathbf{G}(\mathbf{a}') = \mathbf{G}(\mathbf{P}\mathbf{F}((\theta, r)^t))$ defines the same line as $\mathbf{a}'$ (or $\lambda\mathbf{a}'$, $\lambda \neq 0$).

Thus a transformation of a point in parameter space results in the transformation of $(\theta, r)^t$ in (θ, r)-space. The change of coordinate of $(\theta, r)^t$ in (θ, r)-space is given by

$$(\theta', r')^t = \mathbf{H}((\theta, r)^t) \tag{4.1}$$

where $\mathbf{H} = \mathbf{G} \circ \mathbf{P} \circ \mathbf{F}$.[1]

4.2 Encoding Lines by a Basis of Lines

To apply the geometric hashing technique to line features, we need to encode the features in a manner invariant under the transformation group considered. One way of encoding is

[1] We abused the notation by using $\mathbf{P}(\mathbf{v})$ to denote $\mathbf{P}\mathbf{v}$ (matrix $\mathbf{P}$ multiplies vector $\mathbf{v}$). We will continue to use $\mathbf{P}(\mathbf{v})$ or $\mathbf{P}\mathbf{v}$ interchangeably when no ambiguity occurs.

to find a canonical basis and a transformation (in the transformation group under consideration) that maps the basis to the canonical basis, then apply that transformation to the line being encoded. For example, let $\mathbf{T}$ be a projective transformation that maps a basis b to $\mathbf{T}(b)$. If $\mathbf{P}$ is the unique transformation such that $\mathbf{P}(b) = b_c$ and $\mathbf{P}'$ is the unique transformation such that $\mathbf{P}'(\mathbf{T}(b)) = b_c$, where b_c is the chosen canonical basis, then we have $\mathbf{P} = \mathbf{P}' \circ \mathbf{T}$ and any other line l and its correspondence $l' = \mathbf{T}(l)$ will be mapped to $\mathbf{P}(l)$ by $\mathbf{P}$ and $\mathbf{P}'(l')$ by $\mathbf{P}'$ respectively. Since $\mathbf{P}'(l') = \mathbf{P}'(\mathbf{T}(l)) = \mathbf{P}' \circ \mathbf{T}(l) = \mathbf{P}(l)$, this is an invariant encoding[25].

Various subgroups of the projective transformation group require different bases. We discuss them in the following section and give formulae for the corresponding invariants. Throughout the following discussion, we represent a line by its normal parameterization (θ, r) with θ in $[0, \pi)$ and r in $\mathbf{R}$. If the computed invariant (θ', r') has θ' not in $[0..\pi)$, we adjust it by adding π or $-\pi$ and adjust the value of r' by flipping its sign accordingly.

4.3 Line Invariants under Various Transformation Groups

4.3.1 Rigid Line Invariants

Any correspondence of two ordered pairs of non-parallel lines determines a rigid transformation up to a 180°-rotation ambiguity, and this ambiguity can be broken if we know the position of a third line during verification.

One way to encode a third line in terms of a pair of non-parallel lines in a rigid-invariant way is to find a transformation $\mathbf{P}$ which maps the first basis line to the x-axis and maps the second basis line to such that its intersection with the first basis line is the origin, and then apply $\mathbf{P}$ to the third line.

A rigid transformation $\mathbf{T}$ in image space has the form

$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{b} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi & b_1 \\ \sin\phi & \cos\phi & b_2 \\ 0 & 0 & 1 \end{pmatrix},$$

where $\mathbf{R}$ is the rotation matrix and $\mathbf{b}$, the translation vector. This corresponds to the

parameter-space transformation $\mathbf{P}$, defined by (c.f. section 4.1.2)

$$\mathbf{P} = (\mathbf{T}^{-1})^t = \begin{pmatrix} (\mathbf{R}^{-1})^t & 0 \\ -\mathbf{b}^t(\mathbf{R}^{-1})^t & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & 0 \\ -\mathbf{b}^t\mathbf{R} & 1 \end{pmatrix}, \tag{4.2}$$

so that

$$\mathbf{P}^{-1} = \mathbf{T}^t = \begin{pmatrix} \mathbf{R}^t & 0 \\ \mathbf{b}^t & 1 \end{pmatrix}.$$

Let a line be parameterized as $x\cos\theta + y\sin\theta - r = 0$ and let two additional basis lines be represented by their parameters $\mathbf{a}_1$ and $\mathbf{a}_2$, such that

$$\begin{aligned} \mathbf{a}_1 &= (\cos\theta_1, \sin\theta_1, -r_1)^t, \\ \mathbf{a}_2 &= (\cos\theta_2, \sin\theta_2, -r_2)^t, \end{aligned}$$

which are to be mapped by $\mathbf{P}$ to

$$\begin{aligned} \mathbf{e}_1 &= (\cos\frac{\pi}{2}, \sin\frac{\pi}{2}, 0)^t = (0,1,0)^t, \\ \mathbf{e}_2 &= (\cos\varphi, \sin\varphi, 0)^t = (C,S,0)^t. \end{aligned}$$

Note that φ is fixed, though unknown. In this way, $\mathbf{P}$ maps the first basis line to x-axis (or, $y = 0$) and the intersection of the two basis lines to the origin.

Since $ax + by + c = 0$ and $\lambda ax + \lambda by + \lambda c = 0$, $\lambda \neq 0$, represent the same line, we have

$$\begin{aligned} \lambda_1\mathbf{P}\mathbf{a}_1 &= \mathbf{e}_1, \\ \lambda_2\mathbf{P}\mathbf{a}_2 &= \mathbf{e}_2, \end{aligned}$$

where λ_1 and λ_2 are non-zero constants. Equivalently,

$$\mathbf{P}(\lambda_1\mathbf{a}_1, \lambda_2\mathbf{a}_2) = (\mathbf{e}_1, \mathbf{e}_2).$$

Then

$$\begin{aligned} (\lambda_1\mathbf{a}_1, \lambda_2\mathbf{a}_2) &= \mathbf{P}^{-1}(\mathbf{e}_1, \mathbf{e}_2) \\ &= \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ b_1 & b_2 & 1 \end{pmatrix} \begin{pmatrix} 0 & C \\ 1 & S \\ 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} \sin\phi & C\cos\phi + S\sin\phi \\ \cos\phi & -C\sin\phi + S\cos\phi \\ b_2 & C\,b_1 + S\,b_2 \end{pmatrix}. \end{aligned}$$

Thus

$$\lambda_1 = \frac{\sin\phi}{\cos\theta_1} = \frac{\cos\phi}{\sin\theta_1} = \frac{b_2}{-r_1}, \tag{4.3}$$

$$\lambda_2 = \frac{C\,\cos\phi + S\,\sin\phi}{\cos\theta_2} = \frac{-C\,\sin\phi + S\,\cos\phi}{\sin\theta_2} = \frac{C\,b_1 + S\,b_2}{-r_2}. \tag{4.4}$$

From Eq. (4.4), we obtain

$$C = \lambda_2 \cos(\theta_2 + \phi), \tag{4.5}$$

$$S = \lambda_2 \sin(\theta_2 + \phi), \tag{4.6}$$

$$b_1 = \frac{-\lambda_2 r_2 - S\,b_2}{C} \tag{4.7}$$

and from Eq. (4.3), we have $\sin\phi\,\sin\theta_1 = \cos\phi\,\cos\theta_1$ and thus $\phi = \frac{\pi}{2} - \theta_1$ or $\phi = -\frac{\pi}{2} - \theta_1$.

When $\phi = \frac{\pi}{2} - \theta_1$, we have $\lambda_1 = 1$ and $b_2 = -r_1$. Substituting them and Eq. (4.5), Eq. (4.6) in Eq. (4.7), we have $b_1 = -(r_2 - r_1\cos(\theta_1 - \theta_2))/\sin(\theta_1 - \theta_2)$. Thus, from Eq. (4.2) we obtain

$$\mathbf{P} = \begin{pmatrix} \sin\theta_1 & -\cos\theta_1 & 0 \\ \cos\theta_1 & \sin\theta_1 & 0 \\ \csc(\theta_1-\theta_2)(r_2\sin\theta_1 - r_1\sin\theta_2) & \csc(\theta_1-\theta_2)(-r_2\cos\theta_1 + r_1\cos\theta_2) & 1 \end{pmatrix}$$

When $\phi = -\frac{\pi}{2} - \theta_1$, we have $\lambda_1 = -1$ and $b_2 = r_1$. Substituting them and Eq. (4.5), Eq. (4.6) in Eq. (4.7), we have $b_1 = (r_2 - r_1\cos(\theta_1 - \theta_2))/\sin(\theta_1 - \theta_2)$. Thus, from Eq. (4.2) we obtain

$$\mathbf{P} = \begin{pmatrix} -\sin\theta_1 & \cos\theta_1 & 0 \\ -\cos\theta_1 & -\sin\theta_1 & 0 \\ \csc(\theta_1-\theta_2)(r_2\sin\theta_1 - r_1\sin\theta_2) & \csc(\theta_1-\theta_2)(-r_2\cos\theta_1 + r_1\cos\theta_2) & 1 \end{pmatrix}$$

From section 4.1.3, the invariant $(\theta', r')^t$ of a line $(\theta, r)^t$ can be obtained from the basis lines as follows

$$(\theta', r')^t = \mathbf{H}((\theta, r)^t) = \mathbf{G}(\mathbf{P}(\cos\theta, \sin\theta, -r)^t) = \mathbf{G}(\mathbf{a}')$$

where

$$\mathbf{a}' = \begin{pmatrix} -\sin(\theta - \theta_1) \\ \cos(\theta - \theta_1) \\ -r - r_2\csc(\theta_1 - \theta_2)\sin(\theta - \theta_1) + r_1\csc(\theta_1 - \theta_2)\sin(\theta - \theta_2) \end{pmatrix},$$

or

$$\mathbf{a}' = \begin{pmatrix} \sin(\theta - \theta_1) \\ -\cos(\theta - \theta_1) \\ -r - r_2 \csc(\theta_1 - \theta_2) \sin(\theta - \theta_1) + r_1 \csc(\theta_1 - \theta_2) \sin(\theta - \theta_2) \end{pmatrix},$$

depending upon which $\mathbf{P}$ is used, and hence

$$\begin{aligned} \theta' &= \tan^{-1}\left(\frac{\cos(\theta - \theta_1)}{-\sin(\theta - \theta_1)}\right) = \tan^{-1}\left(\frac{\sin(\theta - \theta_1 + \frac{\pi}{2})}{\cos(\theta - \theta_1 + \frac{\pi}{2})}\right) = \theta - \theta_1 + \frac{\pi}{2}, \\ r' &= r + r_2 \csc(\theta_1 - \theta_2) \sin(\theta - \theta_1) - r_1 \csc(\theta_1 - \theta_2) \sin(\theta - \theta_2), \end{aligned}$$

or

$$\begin{aligned} \theta' &= \tan^{-1}\left(\frac{-\cos(\theta - \theta_1)}{\sin(\theta - \theta_1)}\right) = \tan^{-1}\left(\frac{\sin(\theta - \theta_1 - \frac{\pi}{2})}{\cos(\theta - \theta_1 - \frac{\pi}{2})}\right) = \theta - \theta_1 - \frac{\pi}{2}, \\ r' &= r + r_2 \csc(\theta_1 - \theta_2) \sin(\theta - \theta_1) - r_1 \csc(\theta_1 - \theta_2) \sin(\theta - \theta_2). \end{aligned}$$

We may store each encoded invariant (θ', r') redundantly in two entries of the hash table, (θ', r') and $(\theta', -r')$, during preprocessing. Then we may hit a match with either (θ', r') or $(\theta', -r')$ as the computed scene invariant during recognition. (Note that with 180^o-rotation, the computed invariants should be (θ, r) and $(\theta + \pi, r)$. However, we would like to restrict θ to be in $[0..\pi)$, which makes the invariants to be (θ, r) and $(\theta, -r)$, where θ is in $[0..\pi)$.)

4.3.2 Similarity Line Invariants

To determine a similarity transformation uniquely, we need a correspondence of a pair of triplets of lines, not all of which are parallel to each other or intersect at a common point.

Without loss of generality, we may assume that the first basis line intersects the other two basis lines. One way to encode a fourth line in a similarity-invariant way is to find a transformation $\mathbf{P}$ which maps the first basis line to the x-axis, maps the second basis line to such that its intersection with the first basis line is the origin and maps the third basis line to such that its intersection with the first basis line has Euclidean coordinate $(1, 0)$, and then apply $\mathbf{P}$ to the fourth line.

A similarity transformation $\mathbf{T}$ in image space has the form

$$\mathbf{T} = \begin{pmatrix} s\,\mathbf{R} & \mathbf{b} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} s\,\cos\phi & -s\,\sin\phi & b_1 \\ s\,\sin\phi & s\,\cos\phi & b_2 \\ 0 & 0 & 1 \end{pmatrix},$$

where s is the scaling factor; $\mathbf{R}$, the rotation matrix; $\mathbf{b}$, the translation vector. This corresponds to the parameter-space transformation $\mathbf{P}$, defined by (c.f. section 4.1.2)

$$\mathbf{P} = (\mathbf{T}^{-1})^t = \begin{pmatrix} \frac{1}{s}(\mathbf{R}^{-1})^t & 0 \\ -\frac{1}{s}\mathbf{b}^t(\mathbf{R}^{-1})^t & 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{s}\mathbf{R} & 0 \\ -\frac{1}{s}\mathbf{b}^t\mathbf{R} & 1 \end{pmatrix}, \tag{4.8}$$

so that

$$\mathbf{P}^{-1} = \mathbf{T}^t = \begin{pmatrix} s\mathbf{R}^t & 0 \\ \mathbf{b}^t & 1 \end{pmatrix}.$$

Let a line be parameterized as $x\,\cos\theta + y\,\sin\theta - r = 0$ and let three additional basis lines, assuming the first line is not parallel to the others, be represented by their parameters $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$, such that

$$\begin{aligned} \mathbf{a}_1 &= (\cos\theta_1, \sin\theta_1, -r_1)^t, \\ \mathbf{a}_2 &= (\cos\theta_2, \sin\theta_2, -r_2)^t, \\ \mathbf{a}_3 &= (\cos\theta_3, \sin\theta_3, -r_3)^t, \end{aligned}$$

which are to be mapped by $\mathbf{P}$ to

$$\begin{aligned} \mathbf{e}_1 &= (\cos\frac{\pi}{2}, \sin\frac{\pi}{2}, 0)^t = (0, 1, 0)^t, \\ \mathbf{e}_2 &= (\cos\varphi_1, \sin\varphi_1, 0)^t = (C_1, S_1, 0)^t, \\ \mathbf{e}_3 &= (\cos\varphi_2, \sin\varphi_2, -\sin|\theta_1 - \theta_3|)^t = (C_2, S_2, -\sin|\theta_1 - \theta_3|)^t. \end{aligned}$$

Note that φ_1 and φ_2 are fixed, though unknown. Also note that since we fixed the third component of $\mathbf{e}_3$ to be $-\sin|\theta_1 - \theta_3| < 0$, φ_2 ranges within $(-\frac{\pi}{2}, \frac{\pi}{2})$ (or, $C_2 > 0$). In this way, $\mathbf{P}$ maps the first basis line to x-axis; the intersection of the first basis line and the second basis line to the origin; the intersection of the first basis line and the third basis line to $(1, 0)^t$.

Since $ax + by + c = 0$ and $\lambda ax + \lambda by + \lambda c = 0$, $\lambda \neq 0$, represent the same line, we have

$$\begin{aligned} \lambda_1 \mathbf{P}\mathbf{a}_1 &= \mathbf{e}_1, \\ \lambda_2 \mathbf{P}\mathbf{a}_2 &= \mathbf{e}_2, \\ \lambda_3 \mathbf{P}\mathbf{a}_3 &= \mathbf{e}_3, \end{aligned}$$

where λ_1, λ_2 and λ_3 are non-zero constants. Equivalently,

$$\mathbf{P}(\lambda_1 \mathbf{a}_1, \lambda_2 \mathbf{a}_2, \lambda_3 \mathbf{a}_3) = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3).$$

Then

$$\begin{aligned} (\lambda_1 \mathbf{a}_1, \lambda_2 \mathbf{a}_2, \lambda_3 \mathbf{a}_3) &= \mathbf{P}^{-1}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3) \\ &= \begin{pmatrix} s\cos\phi & s\sin\phi & 0 \\ -s\sin\phi & s\cos\phi & 0 \\ b_1 & b_2 & 1 \end{pmatrix} \begin{pmatrix} 0 & C_1 & C_2 \\ 1 & S_1 & S_2 \\ 0 & 0 & -\sin|\theta_1 - \theta_3| \end{pmatrix} \\ &= \begin{pmatrix} s\sin\phi & s\,C_1\cos\phi + s\,S_1\sin\phi & s\,C_2\cos\phi + s\,S_2\sin\phi \\ s\cos\phi & -s\,C_1\sin\phi + s\,S_1\cos\phi & -s\,C_2\sin\phi + s\,S_2\cos\phi \\ b_2 & C_1\,b_1 + S_1\,b_2 & C_2\,b_1 + S_2\,b_2 - \sin|\theta_1 - \theta_3| \end{pmatrix}. \end{aligned}$$

Thus

$$\begin{aligned} \lambda_1 &= \frac{s\sin\phi}{\cos\theta_1} \\ &= \frac{s\cos\phi}{\sin\theta_1} \\ &= \frac{b_2}{-r_1}, \end{aligned} \tag{4.9}$$

$$\begin{aligned} \lambda_2 &= \frac{s\,(C_1\cos\phi + S_1\sin\phi)}{\cos\theta_2} \\ &= \frac{s\,(-C_1\sin\phi + S_1\cos\phi)}{\sin\theta_2} \\ &= \frac{C_1\,b_1 + S_1\,b_2}{-r_2}, \end{aligned} \tag{4.10}$$

$$\begin{aligned} \lambda_3 &= \frac{s\,(C_2\cos\phi + S_2\sin\phi)}{\cos\theta_3} \\ &= \frac{s\,(-C_2\sin\phi + S_2\cos\phi)}{\sin\theta_3} \\ &= \frac{C_2\,b_1 + S_2\,b_2 - \sin|\theta_1 - \theta_3|}{-r_3}. \end{aligned} \tag{4.11}$$

From Eq. (4.9), we have $s \sin\phi \sin\theta_1 = s \cos\phi \cos\theta_1$ and thus $\phi = \frac{\pi}{2} - \theta_1$ or $\phi = -\frac{\pi}{2} - \theta_1$. From Eq. (4.10), we obtain

$$\begin{aligned} C_1 &= \frac{\lambda_2}{s}\cos(\theta_2 + \phi), \\ S_1 &= \frac{\lambda_2}{s}\sin(\theta_2 + \phi), \\ b_1 &= \frac{-\lambda_2 r_2 - S_1 b_2}{C_1}. \end{aligned} \tag{4.12}$$

From Eq. (4.11), we obtain

$$\begin{aligned} C_2 &= \frac{\lambda_3}{s}\cos(\theta_3 + \phi), \\ S_2 &= \frac{\lambda_3}{s}\sin(\theta_3 + \phi), \\ b_1 &= \frac{-\lambda_3 r_3 - S_2 b_2 + \sin|\theta_1 - \theta_3|}{C_2}, \end{aligned} \tag{4.13}$$

and since $C_2^2 + S_2^2 = 1$, we have $\lambda_3 = \pm s$.

When $\phi = \frac{\pi}{2} - \theta_1$, we have $\lambda_1 = s$ and $b_2 = -sr_1$; when $\phi = -\frac{\pi}{2} - \theta_1$, we have $\lambda_1 = -s$ and $b_2 = sr_1$. In both cases, b_1 in Eq. (4.12) and Eq. (4.13) must be consistent, we can solve to get

$$\lambda_3 = \frac{\sin(\theta_1 - \theta_2)\sin|\theta_1 - \theta_3|}{r_1\sin(\theta_2 - \theta_3) + r_2\sin(\theta_3 - \theta_1) + r_3\sin(\theta_1 - \theta_2)}.$$

Since we require $C_2 > 0$, we have $\lambda_3 = s$ when $\theta_1 < \theta_3$ and $\phi = -\frac{\pi}{2} - \theta_1$ or $\theta_1 > \theta_3$ and $\phi = \frac{\pi}{2} - \theta_1$; we have $\lambda_3 = -s$ when $\theta_1 < \theta_3$ and $\phi = \frac{\pi}{2} - \theta_1$ or $\theta_1 > \theta_3$ and $\phi = -\frac{\pi}{2} - \theta_1$.

To summarize, we have four cases as follows:

- **case 1** $\theta_1 < \theta_3$ (thus $\sin|\theta_1 - \theta_3| = -\sin(\theta_1 - \theta_3)$)
  - **case 1.1**

$$\begin{aligned} \phi &= \frac{\pi}{2} - \theta_1, \\ s &= \frac{\sin(\theta_1 - \theta_2)\sin(\theta_1 - \theta_3)}{r_1\sin(\theta_2 - \theta_3) + r_2\sin(\theta_3 - \theta_1) + r_3\sin(\theta_1 - \theta_2)}, \\ b_1 &= \frac{s\,(-r_2 + r_1\cos(\theta_1 - \theta_2))}{\sin(\theta_1 - \theta_2)}, \\ b_2 &= -sr_1; \end{aligned}$$

– **case 1.2**

$$\begin{aligned}
\phi &= -\frac{\pi}{2} - \theta_1, \\
s &= -\frac{\sin(\theta_1 - \theta_2)\sin(\theta_1 - \theta_3)}{r_1 \sin(\theta_2 - \theta_3) + r_2 \sin(\theta_3 - \theta_1) + r_3 \sin(\theta_1 - \theta_2)}, \\
b_1 &= -\frac{s\,(-r_2 + r_1 \cos(\theta_1 - \theta_2))}{\sin(\theta_1 - \theta_2)}, \\
b_2 &= sr_1;
\end{aligned}$$

- **case 2** $\theta_1 > \theta_3$ (thus $\sin|\theta_1 - \theta_3| = \sin(\theta_1 - \theta_3)$)

  – **case 2.1**

$$\begin{aligned}
\phi &= \frac{\pi}{2} - \theta_1, \\
s &= \frac{\sin(\theta_1 - \theta_2)\sin(\theta_1 - \theta_3)}{r_1 \sin(\theta_2 - \theta_3) + r_2 \sin(\theta_3 - \theta_1) + r_3 \sin(\theta_1 - \theta_2)}, \\
b_1 &= \frac{s\,(-r_2 + r_1 \cos(\theta_1 - \theta_2))}{\sin(\theta_1 - \theta_2)}, \\
b_2 &= -sr_1;
\end{aligned}$$

  – **case 2.2**

$$\begin{aligned}
\phi &= -\frac{\pi}{2} - \theta_1, \\
s &= -\frac{\sin(\theta_1 - \theta_2)\sin(\theta_1 - \theta_3)}{r_1 \sin(\theta_2 - \theta_3) + r_2 \sin(\theta_3 - \theta_1) + r_3 \sin(\theta_1 - \theta_2)}, \\
b_1 &= -\frac{s\,(-r_2 + r_1 \cos(\theta_1 - \theta_2))}{\sin(\theta_1 - \theta_2)}, \\
b_2 &= sr_1.
\end{aligned}$$

Note that **case 1.1** and **case 2.1** are identical and **case 1.2** and **case 2.2** are identical. The transformation $\mathbf{P}$ derived from **case 1.1** and **case 1.2** are again identical. We conclude that the above four cases result in

$$\mathbf{P} = \frac{1}{D}\begin{pmatrix} A\sin\theta_1 & -A\cos\theta_1 & 0 \\ A\cos\theta_1 & A\sin\theta_1 & 0 \\ (r_2\sin\theta_1 - r_1\sin\theta_2)\sin(\theta_1 - \theta_3) & -(r_2\cos\theta_1 - r_1\cos\theta_2)\sin(\theta_1 - \theta_3) & D \end{pmatrix}$$

where

$$\begin{aligned} A &= r_1 \sin(\theta_2 - \theta_3) + r_2 \sin(\theta_3 - \theta_1) + r_3 \sin(\theta_1 - \theta_2), \\ D &= \sin(\theta_1 - \theta_2) \sin(\theta_1 - \theta_3). \end{aligned}$$

From section 4.1.3, the invariant $(\theta', r')^t$ of a line $(\theta, r)^t$ can be obtained from the basis lines as follows

$$(\theta', r')^t = \mathbf{H}((\theta, r)^t) = \mathbf{G}(\mathbf{P}(\cos\theta, \sin\theta, -r)^t) = \mathbf{G}(\mathbf{a}')$$

where

$$\mathbf{a}' = \frac{1}{D} \begin{pmatrix} -A \sin(\theta - \theta_1) \\ A \cos(\theta - \theta_1) \\ (-r_2 \sin(\theta - \theta_1) + r_1 \sin(\theta - \theta_2) - r \sin(\theta_1 - \theta_2)) \sin(\theta_1 - \theta_3) \end{pmatrix}$$

and hence

$$\begin{aligned} \theta' &= \tan^{-1}\left(\frac{A \cos(\theta - \theta_1)}{-A \sin(\theta - \theta_1)}\right) = \tan^{-1}\left(\frac{\sin(\theta - \theta_1 + \frac{\pi}{2})}{\cos(\theta - \theta_1 + \frac{\pi}{2})}\right) = \theta - \theta_1 + \frac{\pi}{2}, \\ r' &= (r_2 \sin(\theta - \theta_1) - r_1 \sin(\theta - \theta_2) + r \sin(\theta_1 - \theta_2)) \sin(\theta_1 - \theta_3)/|A|. \end{aligned}$$

4.3.3 Affine Line Invariants

To determine an affine transformation uniquely, we need a correspondence of two ordered triplets of lines, which are not parallel to one another and do not intersect at a common point.

One way to encode a fourth line in an affine-invariant way is to find a transformation $\mathbf{P}$ which maps the three basis lines to a canonical basis, say $x = 0$, $y = 0$ and $x + y = 1$, and then apply $\mathbf{P}$ to the fourth line.

An affine transformation $\mathbf{T}$ in image space has the form

$$\mathbf{T} = \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{pmatrix},$$

where $\mathbf{A}$ is the skewing matrix; $\mathbf{b}$, the translation vector. This corresponds to the parameter-space transformation $\mathbf{P}$, defined by (c.f. section 4.1.2)

$$\mathbf{P} = (\mathbf{T}^{-1})^t = \begin{pmatrix} (\mathbf{A}^{-1})^t & 0 \\ -\mathbf{b}^t(\mathbf{A}^{-1})^t & 1 \end{pmatrix}, \tag{4.14}$$

so that

$$\mathbf{P}^{-1} = \mathbf{T}^t = \begin{pmatrix} \mathbf{A}^t & 0 \\ \mathbf{b}^t & 1 \end{pmatrix}.$$

Let a line be parameterized as $x\cos\theta + y\sin\theta - r = 0$ and let three additional basis lines be represented by their parameters $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$, such that

$$\begin{aligned} \mathbf{a}_1 &= (\cos\theta_1, \sin\theta_1, -r_1)^t, \\ \mathbf{a}_2 &= (\cos\theta_2, \sin\theta_2, -r_2)^t, \\ \mathbf{a}_3 &= (\cos\theta_3, \sin\theta_3, -r_3)^t, \end{aligned}$$

which are to be mapped by $\mathbf{P}$ to

$$\begin{aligned} \mathbf{e}_1 &= (\cos 0, \sin 0, 0)^t = (1,0,0)^t, \\ \mathbf{e}_2 &= (\cos\frac{\pi}{2}, \sin\frac{\pi}{2}, 0)^t = (0,1,0)^t, \\ \mathbf{e}_3 &= (\cos\frac{\pi}{4}, \sin\frac{\pi}{4}, \frac{-1}{\sqrt{2}})^t = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}})^t. \end{aligned}$$

In this way, $\mathbf{P}$ maps the first basis line to y-axis; the second basis line to x-axis; the third basis line to $x + y = 1$.

Since $ax + by + c = 0$ and $\lambda ax + \lambda by + \lambda c = 0$, $\lambda \neq 0$, represent the same line, we have

$$\begin{aligned} \lambda_1 \mathbf{P}\mathbf{a}_1 &= \mathbf{e}_1, \\ \lambda_2 \mathbf{P}\mathbf{a}_2 &= \mathbf{e}_2, \\ \lambda_3 \mathbf{P}\mathbf{a}_3 &= \mathbf{e}_3, \end{aligned}$$

where λ_1, λ_2 and λ_3 are non-zero constants. Equivalently,

$$\mathbf{P}(\lambda_1\mathbf{a}_1, \lambda_2\mathbf{a}_2, \lambda_3\mathbf{a}_3) = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3).$$

Then

$$\begin{aligned} (\lambda_1\mathbf{a}_1, \lambda_2\mathbf{a}_2, \lambda_3\mathbf{a}_3) &= \mathbf{P}^{-1}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3) \\ &= \begin{pmatrix} a_{11} & a_{21} & 0 \\ a_{12} & a_{22} & 0 \\ b_1 & b_2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \frac{1}{\sqrt{2}} \\ 0 & 1 & \frac{1}{\sqrt{2}} \\ 0 & 0 & -\frac{1}{\sqrt{2}} \end{pmatrix} \\ &= \begin{pmatrix} a_{11} & a_{21} & \frac{1}{\sqrt{2}}(a_{11} + a_{21}) \\ a_{12} & a_{22} & \frac{1}{\sqrt{2}}(a_{12} + a_{22}) \\ b_1 & b_2 & \frac{1}{\sqrt{2}}(b_1 + b_2 - 1) \end{pmatrix}. \end{aligned}$$

Thus

$$\lambda_1 = \frac{a_{11}}{\cos\theta_1} = \frac{a_{12}}{\sin\theta_1} = \frac{b_1}{-r_1}, \tag{4.15}$$

$$\lambda_2 = \frac{a_{21}}{\cos\theta_2} = \frac{a_{22}}{\sin\theta_2} = \frac{b_2}{-r_2}, \tag{4.16}$$

$$\lambda_3 = \frac{\frac{1}{\sqrt{2}}(a_{11}+a_{21})}{\cos\theta_3} = \frac{\frac{1}{\sqrt{2}}(a_{12}+a_{22})}{\sin\theta_3} = \frac{\frac{1}{\sqrt{2}}(b_1+b_2-1)}{-r_3}. \tag{4.17}$$

From Eq. (4.15), we obtain

$$a_{11} = \lambda_1 \cos\theta_1, \quad a_{12} = \lambda_1 \sin\theta_1 \quad \text{and} \quad b_1 = -\lambda_1 r_1. \tag{4.18}$$

From Eq. (4.16), we obtain

$$a_{21} = \lambda_2 \cos\theta_2, \quad a_{22} = \lambda_2 \sin\theta_2 \quad \text{and} \quad b_2 = -\lambda_2 r_2. \tag{4.19}$$

From Eq. (4.17), we obtain

$$a_{11}+a_{21} = \lambda_3\sqrt{2}\cos\theta_3, \quad a_{12}+a_{22} = \lambda_3\sqrt{2}\sin\theta_3 \quad \text{and} \quad b_1+b_2-1 = -\lambda_3\sqrt{2}\, r_3. \tag{4.20}$$

Substituting Eq. (4.18) and Eq. (4.19) into Eq. (4.20), we have

$$\begin{aligned} \lambda_1\cos\theta_1 + \lambda_2\cos\theta_2 - \lambda_3\sqrt{2}\cos\theta_3 &= 0, \\ \lambda_1\sin\theta_1 + \lambda_2\sin\theta_2 - \lambda_3\sqrt{2}\sin\theta_3 &= 0, \\ \lambda_1 r_1 + \lambda_2 r_2 - \lambda_3\sqrt{2}\, r_3 &= -1, \end{aligned}$$

and hence

$$\begin{aligned} \lambda_1 &= -\sin(\theta_2-\theta_3)/(r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2)), \\ \lambda_2 &= -\sin(\theta_3-\theta_1)/(r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2)). \end{aligned}$$

Substituting λ_1 and λ_2 back to Eq. (4.18) and Eq. (4.19), we immediately obtain,

$$\begin{aligned} a_{11} &= -\cos\theta_1\sin(\theta_2-\theta_3)/(r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2)), \\ a_{12} &= -\sin\theta_1\sin(\theta_2-\theta_3)/(r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2)), \\ a_{21} &= -\cos\theta_2\sin(\theta_3-\theta_1)/(r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2)), \\ a_{22} &= -\sin\theta_2\sin(\theta_3-\theta_1)/(r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2)), \\ b_1 &= r_1\sin(\theta_2-\theta_3)/(r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2)), \\ b_2 &= r_2\sin(\theta_3-\theta_1)/(r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2)). \end{aligned}$$

Thus the transformation $\mathbf{P}$ in parameter space can be obtained from Eq. (4.14) as

$$\mathbf{P} = \begin{pmatrix} A\csc(\theta_1-\theta_2)\csc(\theta_2-\theta_3)\sin\theta_2 & -A\csc(\theta_1-\theta_2)\csc(\theta_2-\theta_3)\cos\theta_2 & 0 \\ A\csc(\theta_1-\theta_2)\csc(\theta_1-\theta_3)\sin\theta_1 & -A\csc(\theta_1-\theta_2)\csc(\theta_1-\theta_3)\cos\theta_1 & 0 \\ \csc(\theta_1-\theta_2)(r_2\sin\theta_1 - r_1\sin\theta_2) & -\csc(\theta_1-\theta_2)(r_2\cos\theta_1 - r_1\cos\theta_2) & 1 \end{pmatrix}_{3\times 3}$$

where

$$A = r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2).$$

From section 4.1.3, the invariant $(\theta', r')^t$ of a line $(\theta, r)^t$ can be obtained from the basis lines as follows:

$$(\theta', r')^t = \mathbf{H}((\theta, r)^t) = \mathbf{G}(\mathbf{P}(\cos\theta, \sin\theta, -r)^t) = \mathbf{G}(\mathbf{a}'),$$

where

$$\mathbf{a}' = \begin{pmatrix} -\csc(\theta_1-\theta_2)\csc(\theta_2-\theta_3)\sin(\theta-\theta_2) \\ (r_3\sin(\theta_1-\theta_2) + r_2\sin(\theta_3-\theta_1) + r_1\sin(\theta_2-\theta_3)) \\ \\ -\csc(\theta_1-\theta_2)\csc(\theta_1-\theta_3)\sin(\theta-\theta_1) \\ (r_3\sin(\theta_1-\theta_2) + r_2\sin(\theta_3-\theta_1) + r_1\sin(\theta_2-\theta_3)) \\ \\ r_1\csc(\theta_1-\theta_2)\sin(\theta-\theta_2) - r_2\csc(\theta_1-\theta_2)\sin(\theta-\theta_1) - r \end{pmatrix}_{3\times 1}$$

and hence

$$\begin{aligned} \theta' &= \tan^{-1}(A\csc(\theta_1-\theta_3)\sin(\theta_1-\theta)\csc(\theta_1-\theta_2)/ \\ &\qquad A\csc(\theta_2-\theta_3)\sin(\theta_2-\theta)\csc(\theta_1-\theta_2)), \qquad (4.21) \\ r' &= \frac{1}{\tau}(r_1\csc(\theta_1-\theta_2)\sin(\theta_2-\theta) - r_2\csc(\theta_1-\theta_2)\sin(\theta_1-\theta) + r), \qquad (4.22) \end{aligned}$$

where

$$\begin{aligned} \tau &= \sqrt{\csc^2(\theta_1-\theta_3)\sin^2(\theta_1-\theta) + \csc^2(\theta_2-\theta_3)\sin^2(\theta_2-\theta)}\,|A\csc(\theta_1-\theta_2)|, \\ A &= r_1\sin(\theta_2-\theta_3) + r_2\sin(\theta_3-\theta_1) + r_3\sin(\theta_1-\theta_2). \end{aligned}$$

4.3.4 Projective Line Invariants

Although a projective transformation is fractional linear, one can well treat it as a linear transformation by using homogeneous coordinates.

To determine a projective transformation uniquely, we need a correspondence of two ordered quadruplets of lines and for each quadruplet, no three lines are parallel to one another or intersect at a common point (thus no three points are collinear in parameter space).

One way to encode a fifth line in a projective-invariant way is to find a transformation $\mathbf{P}$ which maps the four basis lines to a canonical basis, say $x = 0$, $y = 0$, $x = 1$ and $y = 1$, and then apply $\mathbf{P}$ to the fifth line.

A projective transformation $\mathbf{T}$ in image space has the form

$$\mathbf{T} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}.$$

This corresponds to the parameter-space transformation $\mathbf{P}$, defined by (c.f. section 4.1.2)

$$\mathbf{P} = (\mathbf{T}^{-1})^t = \frac{1}{D} \begin{pmatrix} a_{22}a_{33} - a_{23}a_{32} & a_{23}a_{31} - a_{21}a_{33} & a_{21}a_{32} - a_{22}a_{31} \\ a_{13}a_{32} - a_{12}a_{33} & a_{11}a_{33} - a_{13}a_{31} & a_{12}a_{31} - a_{11}a_{32} \\ a_{12}a_{23} - a_{13}a_{22} & a_{13}a_{21} - a_{11}a_{23} & a_{11}a_{22} - a_{12}a_{21} \end{pmatrix}, \tag{4.23}$$

where

$$D = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} - a_{13}a_{22}a_{31},$$

so that

$$\mathbf{P}^{-1} = \mathbf{T}^t = \begin{pmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{pmatrix}.$$

Let a line be parameterized as $x \cos\theta + y \sin\theta - r = 0$ and let four additional basis lines be represented by their parameters $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{a}_3$ and $\mathbf{a}_4$, such that

$$\begin{aligned} \mathbf{a}_1 &= (\cos\theta_1, \sin\theta_1, -r_1)^t, \\ \mathbf{a}_2 &= (\cos\theta_2, \sin\theta_2, -r_2)^t, \\ \mathbf{a}_3 &= (\cos\theta_3, \sin\theta_3, -r_3)^t, \\ \mathbf{a}_4 &= (\cos\theta_4, \sin\theta_4, -r_4)^t, \end{aligned}$$

which are to be mapped by $\mathbf{P}$ to

$$\begin{aligned}
\mathbf{e}_1 &= (\cos 0, \sin 0, 0)^t = (1,0,0)^t, \\
\mathbf{e}_2 &= (\cos\frac{\pi}{2}, \sin\frac{\pi}{2}, 0)^t = (0,1,0)^t, \\
\mathbf{e}_3 &= (\cos 0, \sin 0, -1)^t = (1,0,-1)^t, \\
\mathbf{e}_4 &= (\cos\frac{\pi}{2}, \sin\frac{\pi}{2}, -1)^t = (0,1,-1)^t.
\end{aligned}$$

In this way, $\mathbf{P}$ maps the first basis line to y-axis; the second basis line to x-axis; the third basis line to $x = 1$; the fourth basis line to $y = 1$.

Since $ax + by + c = 0$ and $\lambda ax + \lambda by + \lambda c = 0$, $\lambda \neq 0$, represent the same line, we have

$$\begin{aligned}
\lambda_1 \mathbf{P}\mathbf{a}_1 &= \mathbf{e}_1, \\
\lambda_2 \mathbf{P}\mathbf{a}_2 &= \mathbf{e}_2, \\
\lambda_3 \mathbf{P}\mathbf{a}_3 &= \mathbf{e}_3, \\
\lambda_4 \mathbf{P}\mathbf{a}_4 &= \mathbf{e}_4,
\end{aligned}$$

where λ_1, λ_2, λ_3 and λ_4 are constants. Equivalently,

$$\mathbf{P}(\lambda_1\mathbf{a}_1, \lambda_2\mathbf{a}_2, \lambda_3\mathbf{a}_3, \lambda_4\mathbf{a}_4) = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4).$$

Then

$$\begin{aligned}
(\lambda_1\mathbf{a}_1, \lambda_2\mathbf{a}_2, \lambda_3\mathbf{a}_3, \lambda_4\mathbf{a}_4) &= \mathbf{P}^{-1}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4) \\
&= \begin{pmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & -1 \end{pmatrix} \\
&= \begin{pmatrix} a_{11} & a_{21} & a_{11}-a_{31} & a_{21}-a_{31} \\ a_{12} & a_{22} & a_{12}-a_{32} & a_{22}-a_{32} \\ a_{13} & a_{23} & a_{13}-a_{33} & a_{23}-a_{33} \end{pmatrix}.
\end{aligned}$$

We have

$$\lambda_1 = \frac{a_{11}}{\cos\theta_1} = \frac{a_{12}}{\sin\theta_1} = \frac{a_{13}}{-r_1}, \tag{4.24}$$

$$\lambda_2 = \frac{a_{21}}{\cos\theta_2} = \frac{a_{22}}{\sin\theta_2} = \frac{a_{23}}{-r_2}, \tag{4.25}$$

$$\lambda_3 = \frac{a_{11}-a_{31}}{\cos\theta_3} = \frac{a_{12}-a_{32}}{\sin\theta_3} = \frac{a_{13}-a_{33}}{-r_3}, \tag{4.26}$$

$$\lambda_4 = \frac{a_{21}-a_{31}}{\cos\theta_4} = \frac{a_{22}-a_{32}}{\sin\theta_4} = \frac{a_{23}-a_{33}}{-r_4}. \tag{4.27}$$

From Eq. (4.24), we obtain

$$a_{11} = \lambda_1 \cos\theta_1, \quad a_{12} = \lambda_1 \sin\theta_1 \quad \text{and} \quad a_{13} = -\lambda_1 r_1. \tag{4.28}$$

From Eq. (4.25), we obtain

$$a_{21} = \lambda_2 \cos\theta_2, \quad a_{22} = \lambda_2 \sin\theta_2 \quad \text{and} \quad a_{23} = -\lambda_2 r_2. \tag{4.29}$$

From Eq. (4.26), we obtain

$$a_{31} = a_{11} - \lambda_3 \cos\theta_3, \quad a_{32} = a_{12} - \lambda_3 \sin\theta_3 \quad \text{and} \quad a_{33} = a_{13} + \lambda_3 r_3. \tag{4.30}$$

From Eq. (4.27), we obtain

$$a_{31} = a_{21} - \lambda_4 \cos\theta_4, \quad a_{32} = a_{22} - \lambda_4 \sin\theta_4 \quad \text{and} \quad a_{33} = a_{23} + \lambda_4 r_4. \tag{4.31}$$

Since Eq. (4.30) and Eq. (4.31) must be consistent, we have by substituting Eq. (4.28) in Eq. (4.30) and Eq. (4.29) in Eq. (4.31),

$$\begin{aligned}
\lambda_1 \cos\theta_1 - \lambda_2 \cos\theta_2 - \lambda_3 \cos\theta_3 &= -\lambda_4 \cos\theta_4, \\
\lambda_1 \sin\theta_1 - \lambda_2 \sin\theta_2 - \lambda_3 \sin\theta_3 &= -\lambda_4 \sin\theta_4, \\
\lambda_1 r_1 - \lambda_2 r_2 - \lambda_3 r_3 &= -\lambda_4 r_4,
\end{aligned}$$

and hence

$$\lambda_1 = \frac{C_1}{C_4}\lambda_4, \quad \lambda_2 = \frac{C_2}{C_4}\lambda_4 \quad \text{and} \quad \lambda_3 = \frac{C_3}{C_4}\lambda_4,$$

where

$$\begin{aligned}
C_1 &= -r_4 \sin(\theta_2 - \theta_3) + r_3 \sin(\theta_2 - \theta_4) - r_2 \sin(\theta_3 - \theta_4), \\
C_2 &= -r_4 \sin(\theta_1 - \theta_3) + r_3 \sin(\theta_1 - \theta_4) - r_1 \sin(\theta_3 - \theta_4), \\
C_3 &= r_4 \sin(\theta_1 - \theta_2) - r_2 \sin(\theta_1 - \theta_4) + r_1 \sin(\theta_2 - \theta_4), \\
C_4 &= r_1 \sin(\theta_2 - \theta_3) + r_2 \sin(\theta_3 - \theta_1) + r_3 \sin(\theta_1 - \theta_2).
\end{aligned}$$

Thus

$$a_{11} = \cos\theta_1 \frac{C_1}{C_4}\lambda_4, \; a_{12} = \sin\theta_1 \frac{C_1}{C_4}\lambda_4 \; \text{and} \; a_{13} = -r_1 \frac{C_1}{C_4}\lambda_4,$$

$$a_{21} = \cos\theta_2 \frac{C_2}{C_4}\lambda_4, \; a_{22} = \sin\theta_2 \frac{C_2}{C_4}\lambda_4, \; \text{and} \; a_{23} = -r_2 \frac{C_2}{C_4}\lambda_4,$$

$a_{31} = (\cos\theta_1 \frac{C_1}{C_4} - \cos\theta_3 \frac{C_3}{C_4})\lambda_4$, $a_{32} = (\sin\theta_1 \frac{C_1}{C_4} - \sin\theta_3 \frac{C_3}{C_4})\lambda_4$ and $a_{33} = (-r_1 \frac{C_1}{C_4} + r_3 \frac{C_3}{C_4})\lambda_4$.
The transformation $\mathbf{P}$ in parameter space can be obtained from Eq. (4.23) as

$$\mathbf{P} = -1/C_1C_2C_3\lambda_4 \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{pmatrix}$$

where

$$\begin{aligned}
p_{11} &= C_2(C_1 r_2 \sin\theta_1 - C_1 r_1 \sin\theta_2 + C_3 r_3 \sin\theta_2 - C_3 r_2 \sin\theta_3), \\
p_{12} &= -C_2(C_1 r_2 \cos\theta_1 - C_1 r_1 \cos\theta_2 + C_3 r_3 \cos\theta_2 - C_3 r_2 \cos\theta_3), \\
p_{13} &= C_2(C_1 \sin(\theta_1 - \theta_2) + C_3 \sin(\theta_2 - \theta_3)), \\
p_{21} &= C_1C_3(r_1 \sin\theta_3 - r_3 \sin\theta_1), \\
p_{22} &= -C_1C_3(r_1 \cos\theta_3 - r_3 \cos\theta_1), \\
p_{23} &= -C_1C_3 \sin(\theta_1 - \theta_3), \\
p_{31} &= C_1C_2(r_1 \sin\theta_2 - r_2 \sin\theta_1), \\
p_{32} &= -C_1C_2(r_1 \cos\theta_2 - r_2 \cos\theta_1), \\
p_{33} &= -C_1C_2 \sin(\theta_1 - \theta_2).
\end{aligned}$$

From section 4.1.3, the invariant $(\theta', r')^t$ of a line $(\theta, r)^t$ can be obtained from the basis lines as follows

$$(\theta', r')^t = \mathbf{H}((\theta, r)^t) = \mathbf{G}(\mathbf{P}(\cos\theta, \sin\theta, -r)^t) = \mathbf{G}(\mathbf{a}')$$

where

$$\mathbf{a}' = -1/C_1C_2C_3\lambda_4 \begin{pmatrix} ABC \\ DEF \\ GCF \end{pmatrix}_{3\times 1}$$

where

$$\begin{aligned}
A &= (r_3 \sin(\theta_1 - \theta_2) - r_2 \sin(\theta_1 - \theta_3) + r_1 \sin(\theta_2 - \theta_3)), \\
B &= (r_4 \sin(\theta - \theta_2) - r_2 \sin(\theta - \theta_4) + r \sin(\theta_2 - \theta_4)), \\
C &= (r_4 \sin(\theta_1 - \theta_3) - r_3 \sin(\theta_1 - \theta_4) + r_1 \sin(\theta_3 - \theta_4)),
\end{aligned}$$

$$
\begin{aligned}
D &= (-r_3 \sin(\theta - \theta_1) + r_1 \sin(\theta - \theta_3) - r \sin(\theta_1 - \theta_3)), \\
E &= (r_4 \sin(\theta_1 - \theta_2) - r_2 \sin(\theta_1 - \theta_4) + r_1 \sin(\theta_2 - \theta_4)), \\
F &= (r_4 \sin(\theta_2 - \theta_3) - r_3 \sin(\theta_2 - \theta_4) + r_2 \sin(\theta_3 - \theta_4)), \\
G &= (r_2 \sin(\theta - \theta_1) - r_1 \sin(\theta - \theta_2) + r \sin(\theta_1 - \theta_2)),
\end{aligned}
$$

and hence

$$
\begin{aligned}
\theta' &= \tan^{-1}(DEF/ABC), \\
r' &= \frac{-GCF}{\sqrt{(ABC)^2 + (DEF)^2}}.
\end{aligned}
$$

Chapter 5

The Effect of Noise on the Invariants

In an environment affected by serious noise and occlusion, line features are usually detected using the Hough transform. However, detected line parameters (θ, r)'s always deviate slightly from their true values, since in the physical process of image acquisition the positions of the endpoints of a segment are usually randomly perturbed and occlusion also affects the process of the Hough transform. As long as a segment is not too short, this induces only slight perturbation of the line parameters of the segment. Thus it is reasonable for us to assume that detected line parameters differ from the "true" values by a small perturbation having a Gaussian distribution.

In the following, we derive the "spread" of the computed invariant over hash space from the "perturbation" of the lines which give rise to this invariant.

5.1 A Noise Model for Line Parameters

We make the following mild assumptions: The measured line parameters (θ, r)'s of a set of lines

1. are statistically independent;
2. are distributed according to a Gaussian distribution centered at the true value of the parameter;

3. have a fixed covariance $\Sigma = \begin{pmatrix} \sigma_\theta^2 & \sigma_{\theta r} \\ \sigma_{\theta r} & \sigma_r^2 \end{pmatrix}$.

More precisely, let (θ, r) be the "true" value of the parameters of a line and $(\Delta\Theta, \Delta R)$ be the stochastic variable denoting the perturbation of (θ, r). The joint probability density function of $\Delta\Theta$ and ΔR is then given by

$$p(\Delta\Theta, \Delta R) = \frac{1}{2\pi\sqrt{|\Sigma|}} \exp[-\frac{1}{2}(\Delta\Theta, \Delta R)\Sigma^{-1}(\Delta\Theta, \Delta R)^t]$$

and is centered around (θ, r).

5.2 The Spread Function of the Invariants

In the following, we derive, from the perturbations of the basis lines and the line being encoded, a closed-form formula of the perturbation of the invariants considered in the preceding chapter.

We first introduce a lemma and a corollary derived from the lemma.

Lemma: Let $X_1, X_2, ..., X_n$ have a multivariate Gaussian distribution with vector $\mathbf{u}$ of means and positive definite covariance matrix $\mathbf{\Sigma}$. Then the moment-generating function of the multivariate Gaussian *p.d.f.* is given by

$$\exp[\mathbf{v}^t\mathbf{u} + \frac{\mathbf{v}^t\mathbf{\Sigma}\mathbf{v}}{2}], \text{ for all real vectors of } \mathbf{v}.$$

Corollary: Let $\mathbf{Y}^t = (Y_1, Y_2)$ such that $\mathbf{Y} = \mathbf{C}\mathbf{X}$, where $\mathbf{X}^t = (X_1, X_2, ..., X_n)$ and $\mathbf{C} = \begin{pmatrix} c_{11} & ... & c_{1n} \\ c_{21} & ... & c_{2n} \end{pmatrix}$, a real $2 \times n$ matrix. Then the random variable $\mathbf{Y}$ is $N(\mathbf{C}\mathbf{u}, \mathbf{C}\mathbf{\Sigma}\mathbf{C}^t)$.

Proof. The moment-generating function of the distribution $\mathbf{Y}$ is given by

$$M(\mathbf{v}) = E(e^{\mathbf{v}^t\mathbf{Y}}) = E(e^{\mathbf{v}^t\mathbf{C}\mathbf{X}}) = E(e^{(\mathbf{C}^t\mathbf{v})^t\mathbf{X}})$$

Applying the above theorem, we have

$$\begin{aligned} M(\mathbf{v}) &= \exp[(\mathbf{C}^t\mathbf{v})^t\mathbf{u} + \frac{(\mathbf{C}^t\mathbf{v})^t\mathbf{\Sigma}(\mathbf{C}^t\mathbf{v})}{2}] \\ &= \exp[\mathbf{v}^t(\mathbf{C}\mathbf{u}) + \frac{\mathbf{v}^t(\mathbf{C}\mathbf{\Sigma}\mathbf{C}^t)\mathbf{v}}{2}] \end{aligned}$$

Thus the random variable $\mathbf{Y}$ is $N(\mathbf{C}\mathbf{u}, \mathbf{C}\mathbf{\Sigma}\mathbf{C}^t)$. *Q.E.D.*

Now, we go back to consider the perturbation of the invariant from the perturbation of lines. In the preceding chapter, we have given formulae of the invariants (θ', r')'s for various transformations. These invariants are functions of the line being encoded, $(\theta, r)^t$ and the basis lines, $(\theta_i, r_i)^t_{i=1,\ldots,k}$:

$$(\theta', r')^t = f((\theta, r)^t, (\theta_1, r_1)^t, \cdots, (\theta_k, r_k)^t).$$

Equivalently we may rewrite the above equation as follows:

$$\theta' = f'((\theta, r)^t, (\theta_1, r_1)^t, \cdots, (\theta_k, r_k)^t), \quad (5.1)$$

$$r' = f''((\theta, r)^t, (\theta_1, r_1)^t, \cdots, (\theta_k, r_k)^t). \quad (5.2)$$

The exact forms of Eq. (5.1) and Eq. (5.2), together with the value of k (i.e., the number of basis lines needed), depend on the viewing transformation under consideration and are given in section 4.3.1, 4.3.2, 4.3.3 and 4.3.4 respectively.

Introducing perturbations of the line parameters $(\theta, r)^t$ and $(\theta_i, r_i)^t_{i=1,\ldots,k}$ results in a perturbation of the computed invariant $(\theta', r')^t$, having the following form:

$$\theta' + \delta\theta' = f'((\theta + \delta\theta, r + \delta r)^t, (\theta_1 + \delta\theta_1, r_1 + \delta r_1)^t, \cdots, (\theta_k + \delta\theta_k, r_k + \delta r_k)^t), \quad (5.3)$$

$$r' + \delta r' = f''((\theta + \delta\theta, r + \delta r)^t, (\theta_1 + \delta\theta_1, r_1 + \delta r_1)^t, \cdots, (\theta_k + \delta\theta_k, r_k + \delta r_k)^t). \quad (5.4)$$

Expanding Eq. (5.3) and Eq. (5.4) in Maclaurin series and ignoring second and higher order perturbation terms and then subtracting Eq. (5.1) and Eq. (5.2) from them, we have

$$\begin{aligned}\delta\theta' &= \frac{\partial\theta'}{\partial\theta}\delta\theta + \frac{\partial\theta'}{\partial r}\delta r + \sum_{i=1}^{k}\left(\frac{\partial\theta'}{\partial\theta_i}\delta\theta_i + \frac{\partial\theta'}{\partial r_i}\delta r_i\right) \\ &= c_{11}\delta\theta + c_{12}\delta r + c_{13}\delta\theta_1 + c_{14}\delta r_1 + \ldots + c_{1,2k+1}\delta\theta_k + c_{1,2k+2}\delta r_k, \quad (5.5) \\ \delta r' &= \frac{\partial r'}{\partial\theta}\delta\theta + \frac{\partial r'}{\partial r}\delta r + \sum_{i=1}^{k}\left(\frac{\partial r'}{\partial\theta_i}\delta\theta_i + \frac{\partial r'}{\partial r_i}\delta r_i\right) \\ &= c_{21}\delta\theta + c_{22}\delta r + c_{23}\delta\theta_1 + c_{24}\delta r_1 + \ldots + c_{2,2k+1}\delta\theta_k + c_{2,2k+2}\delta r_k, \quad (5.6)\end{aligned}$$

where $c_{11} = \partial\theta'/\partial\theta$, $c_{12} = \partial\theta'/\partial r$, $c_{21} = \partial r'/\partial\theta$, $c_{22} = \partial r'/\partial r$, $c_{1,i} = \partial\theta'/\partial\theta_{i-2}$, $c_{2,i} = \partial r'/\partial\theta_{i-2}$, for $i = 3, 5, \ldots, 2k+1$, and $c_{1,i} = \partial\theta'/\partial r_{i-2}$, $c_{2,i} = \partial r'/\partial r_{i-2}$, for $i = 4, 6, \ldots, 2k+2$.

Let $(\Delta\Theta, \Delta R)$, $(\Delta\Theta_i, \Delta R_i)_{i=1,\ldots,k}$ and $(\Delta\Theta', \Delta R')$ be stochastic variables denoting the perturbations of (θ, r), $(\theta_i, r_i)_{i=1,\ldots,k}$ and (θ', r') respectively. Eq. (5.5) and Eq. (5.6) can

be rewritten as

$$\begin{aligned}\Delta\Theta^{'} &= c_{11}\Delta\Theta + c_{12}\Delta R + c_{13}\Delta\Theta_1 + c_{14}\Delta R_1 + \ldots + c_{1,2k+1}\Delta\Theta_k + c_{1,2k+2}\Delta R_k,\\ \Delta R^{'} &= c_{21}\Delta\Theta + c_{22}\Delta R + c_{23}\Delta\Theta_1 + c_{24}\Delta R_1 + \ldots + c_{2,2k+1}\Delta\Theta_k + c_{2,2k+2}\Delta R_k.\end{aligned}$$

Since $p(\Delta\Theta, \Delta R)$ and $p(\Delta\Theta_i, \Delta R_i)_{i=1,\ldots,k}$ are Gaussian and independent, we have that $p.d.f.$ $p(\Delta\Theta, \Delta R, \Delta\Theta_1, \Delta R_1, \ldots, \Delta\Theta_k, \Delta R_k)$ is also Gaussian. More precisely,

$$\begin{aligned}& p(\Delta\Theta, \Delta R, \Delta\Theta_1, \Delta R_1, \ldots, \Delta\Theta_k, \Delta R_k)\\ = \;& p(\Delta\Theta, \Delta R)\, p(\Delta\Theta_1, \Delta R_1)\cdots p(\Delta\Theta_k, \Delta R_k)\\ = \;& (\frac{1}{(2\pi)^{k+1}\sqrt{|\hat{\mathbf{\Sigma}}|}})\exp[-\frac{1}{2}\mathbf{V}^t\hat{\mathbf{\Sigma}}^{-1}\mathbf{V}]\end{aligned}$$

where

$$\begin{aligned}\mathbf{V} &= (\Delta\Theta, \Delta R, \Delta\Theta_1, \Delta R_1, \ldots, \Delta\Theta_k, \Delta R_k)^t\\ \hat{\mathbf{\Sigma}} &= \begin{pmatrix} \sigma_\theta^2 & \sigma_{\theta r} & \ldots & \ldots & 0 & 0\\ \sigma_{\theta r} & \sigma_r^2 & & & 0 & 0\\ \vdots & & \ddots & & & \vdots\\ \vdots & & & \ddots & & \vdots\\ 0 & 0 & & & \sigma_\theta^2 & \sigma_{\theta r}\\ 0 & 0 & \ldots & \ldots & \sigma_{\theta r} & \sigma_r^2 \end{pmatrix}_{(2k+2)\times(2k+2).}\end{aligned}$$

Since

$$(\Delta\Theta', \Delta R')^t = \begin{pmatrix} c_{11} & c_{12} & \ldots & c_{1,2k+1} & c_{1,2k+2}\\ c_{21} & c_{22} & \ldots & c_{2,2k+1} & c_{2,2k+2}\end{pmatrix}\mathbf{V},$$

from the above corollary, we can show that $(\Delta\Theta', \Delta R')$ also has a Gaussian distribution having $p.d.f.$

$$p(\Delta\Theta', \Delta R') = \frac{1}{2\pi\sqrt{|\mathbf{\Sigma}'|}}\exp[-\frac{1}{2}(\Delta\Theta', \Delta R')\mathbf{\Sigma}'^{-1}(\Delta\Theta', \Delta R')^t] \tag{5.7}$$

and covariance matrix

$$\mathbf{\Sigma}' = \begin{pmatrix} \sigma_{\theta'}^2 & \sigma_{\theta' r'}\\ \sigma_{\theta' r'} & \sigma_{r'}^2\end{pmatrix},$$

where

$$\begin{aligned}
\sigma_{\theta'}^2 &= (c_{11}^2 + c_{13}^2 + \cdots + c_{1,2k+1}^2)\sigma_\theta^2 + (c_{12}^2 + c_{14}^2 + \cdots + c_{1,2k+2}^2)\sigma_r^2 + \\
&\quad 2(c_{11}c_{12} + c_{13}c_{14} + \cdots + c_{1,2k+1}c_{1,2k+2})\sigma_{\theta r}, \\
\sigma_{r'}^2 &= (c_{21}^2 + c_{23}^2 + \cdots + c_{2,2k+1}^2)\sigma_\theta^2 + (c_{22}^2 + c_{24}^2 + \cdots + c_{2,2k+2}^2)\sigma_r^2 + \\
&\quad 2(c_{21}c_{22} + c_{23}c_{24} + \cdots + c_{2,2k+1}c_{2,2k+2})\sigma_{\theta r}, \\
\sigma_{\theta' r'} &= (c_{11}c_{21} + c_{13}c_{23} + \cdots + c_{1,2k+1}c_{2,2k+1})\sigma_\theta^2 + \\
&\quad (c_{12}c_{22} + c_{14}c_{24} + \cdots + c_{1,2k+2}c_{2,2k+2})\sigma_r^2 + \\
&\quad (c_{11}c_{22} + c_{13}c_{24} + \cdots + c_{1,2k+1}c_{2,2k+2} + c_{21}c_{12} + c_{23}c_{14} + \cdots + c_{2,2k+1}c_{1,2k+2})\sigma_{\theta r}.
\end{aligned}$$

Recall that in computing the invariant $(\theta', r')^t$, we may adjust the value of θ' by adding π or $-\pi$ with the sign of r' flipped accordingly such that θ' is in $[0..\pi)$. If this is the case, then the covariance matrix $\mathbf{\Sigma}'$ must be adjusted by flipping the sign of $\sigma_{\theta' r'}$.

5.3 Spread Functions under Various Transformation Groups

In the following subsections, we specialize the general result derived in the preceding section to the various transformation groups considered earlier.

5.3.1 Rigid-Invariant Spread Function

From section 4.3.1, we have two sets of invariants (due to 180^o-rotation ambiguity) for rigid transformations as follows:

$$\begin{aligned}
\theta' &= \theta - \theta_1 + \frac{\pi}{2}, \\
r' &= r + \csc(\theta_1 - \theta_2)(r_2 \sin(\theta - \theta_1) - r_1 \sin(\theta - \theta_2)),
\end{aligned}$$

and

$$\begin{aligned}
\theta' &= \theta - \theta_1 - \frac{\pi}{2}, \\
r' &= r + \csc(\theta_1 - \theta_2)(r_2 \sin(\theta - \theta_1) - r_1 \sin(\theta - \theta_2)),
\end{aligned}$$

where (θ, r) is the parameter of the line being encoded and $(\theta_i, r_i)_{i=1,2}$ are the parameters of the basis lines.

Let $(\Delta\Theta, \Delta R)$, $(\Delta\Theta_i, \Delta R_i)_{i=1,2}$ and $(\Delta\Theta^{'}, \Delta R^{'})$ be stochastic variables denoting respectively the perturbations of (θ, r), $(\theta_i, r_i)_{i=1,2}$ and the computed invariant $(\theta^{'}, r^{'})$. From section 5.2, we have

$$\begin{aligned}
\Delta\Theta^{'} &= c_{11}\Delta\Theta + c_{12}\Delta R + c_{13}\Delta\Theta_1 + c_{14}\Delta R_1 + c_{15}\Delta\Theta_2 + c_{16}\Delta R_2, \\
\Delta R^{'} &= c_{21}\Delta\Theta + c_{22}\Delta R + c_{23}\Delta\Theta_1 + c_{24}\Delta R_1 + c_{25}\Delta\Theta_2 + c_{26}\Delta R_2.
\end{aligned}$$

Computing $c_{i,j}$'s, we obtain,

$$\begin{aligned}
&c_{11} = 1, && c_{21} = \csc(\theta_1 - \theta_2)(r_2\cos(\theta - \theta_1) - r_1\cos(\theta - \theta_2)), \\
&c_{12} = 0, && c_{22} = 1, \\
&c_{13} = -1, && c_{23} = -\sin(\theta - \theta_2)\csc^2(\theta_1 - \theta_2)(r_2 - r_1\cos(\theta_1 - \theta_2)), \\
&c_{14} = 0, && c_{24} = -\sin(\theta - \theta_2)\csc(\theta_1 - \theta_2), \\
&c_{15} = 0, && c_{25} = -\sin(\theta - \theta_1)\csc^2(\theta_1 - \theta_2)(r_1 - r_2\cos(\theta_1 - \theta_2)), \\
&c_{16} = 0, && c_{26} = \sin(\theta - \theta_1)\csc(\theta_1 - \theta_2).
\end{aligned}$$

Thus we conclude that $(\Delta\Theta', \Delta R')$ has a Gaussian distribution with $p.d.f$

$$p(\Delta\Theta', \Delta R') = \frac{1}{2\pi\sqrt{|\mathbf{\Sigma}|}}\exp[-\frac{1}{2}(\Delta\Theta', \Delta R')\mathbf{\Sigma}^{-1}(\Delta\Theta', \Delta R')^t]$$

and covariance matrix

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma^2_{\theta'} & \sigma_{\theta' r'} \\ \sigma_{\theta' r'} & \sigma^2_{r'} \end{pmatrix},$$

where

$$\begin{aligned}
\sigma^2_{\theta'} &= 2\sigma^2_\theta, \\
\sigma^2_{r'} &= (c^2_{21} + c^2_{23} + c^2_{25})\sigma^2_\theta + (1 + c^2_{24} + c^2_{26})\sigma^2_r + 2(c_{21} + c_{23}c_{24} + c_{25}c_{26})\sigma_{\theta r}, \\
\sigma_{\theta' r'} &= (c_{21} - c_{23})\sigma^2_\theta + (c_{22} - c_{24})\sigma_{\theta r}.
\end{aligned}$$

5.3.2 Similarity-Invariant Spread Function

From section 4.3.2, we have the following formula for the invariant for similarity transformations:

$$\begin{aligned}
\theta' &= = \theta - \theta_1 + \frac{\pi}{2}, \\
r' &= \sin(\theta_1 - \theta_3)\, B/|A|
\end{aligned}$$

and

$$\begin{aligned} A &= r_1 \sin(\theta_2 - \theta_3) + r_2 \sin(\theta_3 - \theta_1) + r_3 \sin(\theta_1 - \theta_2), \\ B &= r_2 \sin(\theta - \theta_1) - r_1 \sin(\theta - \theta_2) + r \sin(\theta_1 - \theta_2), \end{aligned}$$

where (θ, r) is the parameter of the line being encoded and $(\theta_i, r_i)_{i=1,2,3}$ are the parameters of the basis lines.

Let $(\Delta\Theta, \Delta R)$, $(\Delta\Theta_i, \Delta R_i)_{i=1,2,3}$ and $(\Delta\Theta^{'}, \Delta R^{'})$ be stochastic variables denoting respectively the perturbations of (θ, r), $(\theta_i, r_i)_{i=1,2,3}$ and the computed invariant $(\theta^{'}, r^{'})$. From section 5.2, we have

$$\begin{aligned} \Delta\Theta^{'} &= c_{11}\Delta\Theta + c_{12}\Delta R + c_{13}\Delta\Theta_1 + c_{14}\Delta R_1 + c_{15}\Delta\Theta_2 + c_{16}\Delta R_2 + c_{17}\Delta\Theta_3 + c_{18}\Delta R_3, \\ \Delta R^{'} &= c_{21}\Delta\Theta + c_{22}\Delta R + c_{23}\Delta\Theta_1 + c_{24}\Delta R_1 + c_{25}\Delta\Theta_2 + c_{26}\Delta R_2 + c_{27}\Delta\Theta_3 + c_{28}\Delta R_3. \end{aligned}$$

Computing $c_{i,j}$'s, we obtain,

$$\begin{aligned} &c_{11} = 1, && c_{21} = r_2 \cos(\theta - \theta_1) - r_1 \cos(\theta - \theta_2)/|A|, \\ &c_{12} = 0, && c_{22} = \sin(\theta_1 - \theta_2)/|A|, \\ &c_{13} = -1, && c_{23} = [(r_2 \cos(\theta_1 - \theta_3) - r_3 \cos(\theta_1 - \theta_2))B/A - \\ & && \quad r_2 \cos(\theta - \theta_1) + r \cos(\theta_1 - \theta_2)]/|A|, \\ &c_{14} = 0, && c_{24} = [-\sin(\theta_2 - \theta_3)B/A - \sin(\theta - \theta_2)]/|A|, \\ &c_{15} = 0, && c_{25} = [(r_3 \cos(\theta_1 - \theta_2) - r_1 \cos(\theta_2 - \theta_3))B/A + \\ & && \quad r_1 \cos(\theta - \theta_2) - r \cos(\theta_1 - \theta_2)]/|A|, \\ &c_{16} = 0, && c_{26} = [\sin(\theta_1 - \theta_3)B/A + \sin(\theta - \theta_1)]/|A|, \\ &c_{17} = 0, && c_{27} = (r_1 \cos(\theta_2 - \theta_3) - r_2 \cos(\theta_1 - \theta_3))B/(A|A|), \\ &c_{18} = 0, && c_{28} = -\sin(\theta_1 - \theta_2)B/(A|A|). \end{aligned}$$

Thus we conclude that $(\Delta\Theta', \Delta R')$ has a Gaussian distribution with $p.d.f$

$$p(\Delta\Theta', \Delta R') = \frac{1}{2\pi\sqrt{|\mathbf{\Sigma}|}} \exp[-\frac{1}{2}(\Delta\Theta', \Delta R')\mathbf{\Sigma}^{-1}(\Delta\Theta', \Delta R')^t]$$

and covariance matrix

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma^2_{\theta'} & \sigma_{\theta' r'} \\ \sigma_{\theta' r'} & \sigma^2_{r'} \end{pmatrix},$$

where

$$\begin{aligned}
\sigma^2_{\theta'} &= 2\sigma^2_\theta, \\
\sigma^2_{r'} &= (c^2_{21}+c^2_{23}+c^2_{25}+c^2_{27})\sigma^2_\theta + (c^2_{22}+c^2_{24}+c^2_{26}+c^2_{28})\sigma^2_r + \\
&\quad 2(c_{21}c_{22}+c_{23}c_{24}+c_{25}c_{26}+c_{27}c_{28})\sigma_{\theta r}, \\
\sigma_{\theta' r'} &= (c_{21}-c_{23})\sigma^2_\theta + (c_{22}-c_{24})\sigma_{\theta r}.
\end{aligned}$$

5.3.3 Affine-Invariant Spread Function

From section 4.3.3, the invariant for affine transformations has the following form:

$$\begin{aligned}
\theta' &= \tan^{-1}(A\,\csc(\theta_1-\theta_3)\sin(\theta_1-\theta)\csc(\theta_1-\theta_2)/ \\
&\qquad A\,\csc(\theta_2-\theta_3)\sin(\theta_2-\theta)\csc(\theta_1-\theta_2)), \\
r' &= C/D
\end{aligned}$$

and

$$\begin{aligned}
A &= r_3\sin(\theta_1-\theta_2)+r_2\sin(\theta_3-\theta_1)+r_1\sin(\theta_2-\theta_3), \\
B &= \csc^2(\theta_1-\theta_3)\sin^2(\theta_1-\theta_4)+\csc^2(\theta_2-\theta_3)\sin^2(\theta_2-\theta_4), \\
C &= r_1\csc(\theta_1-\theta_2)\sin(\theta_2-\theta)-r_2\csc(\theta_1-\theta_2)\sin(\theta_1-\theta)+r, \\
D &= \sqrt{A^2B\,\csc^2(\theta_1-\theta_2)},
\end{aligned}$$

where (θ, r) is the parameter of the line being encoded and $(\theta_i, r_i)_{i=1,2,3}$ are the parameters of the basis lines.

Let $(\Delta\Theta, \Delta R)$, $(\Delta\Theta_i, \Delta R_i)_{i=1,2,3}$ and $(\Delta\Theta', \Delta R')$ be stochastic variables denoting respectively the perturbations of (θ, r), $(\theta_i, r_i)_{i=1,2,3}$ and the computed invariant (θ', r'). From section 5.2, we have

$$\begin{aligned}
\Delta\Theta' &= c_{11}\Delta\Theta + c_{12}\Delta R + c_{13}\Delta\Theta_1 + c_{14}\Delta R_1 + c_{15}\Delta\Theta_2 + c_{16}\Delta R_2 + c_{17}\Delta\Theta_3 + c_{18}\Delta R_3, \\
\Delta R' &= c_{21}\Delta\Theta + c_{22}\Delta R + c_{23}\Delta\Theta_1 + c_{24}\Delta R_1 + c_{25}\Delta\Theta_2 + c_{26}\Delta R_2 + c_{27}\Delta\Theta_3 + c_{28}\Delta R_3.
\end{aligned}$$

Computing $c_{i,j}$'s and writing $E = \csc(\theta_1-\theta_3)\csc(\theta_2-\theta_3)/B$, we obtain,

$$\begin{aligned}
c_{11} &= \sin(\theta_1-\theta_2)E, \\
c_{12} &= 0,
\end{aligned}$$

$$
\begin{aligned}
c_{13} &= -\csc(\theta_1-\theta_3)\sin(\theta-\theta_2)\sin(\theta-\theta_3)E,\\
c_{14} &= 0,\\
c_{15} &= \csc(\theta_2-\theta_3)\sin(\theta-\theta_1)\sin(\theta-\theta_3)E,\\
c_{16} &= 0,\\
c_{17} &= -\csc(\theta_1-\theta_3)\csc(\theta_2-\theta_3)\sin(\theta_1-\theta_2)\sin(\theta-\theta_1)\sin(\theta-\theta_2)E,\\
c_{18} &= 0,\\
c_{21} &= [-(\cos(\theta-\theta_1)\csc^2(\theta_1-\theta_3)\sin(\theta-\theta_1)+\\
&\quad \cos(\theta-\theta_2)\csc^2(\theta_2-\theta_3)\sin(\theta-\theta_2))C/B+\\
&\quad \csc(\theta_1-\theta_2)(r_2\cos(\theta-\theta_1)-r_1\cos(\theta-\theta_2))]/D,\\
c_{22} &= 1/D,\\
c_{23} &= [-\csc(\theta_1-\theta_2)(r_2\cos(\theta-\theta_1)+\cot(\theta_1-\theta_2)(r_2\sin(\theta-\theta_1)-r_1\sin(\theta-\theta_2)))-\\
&\quad C((r_3\cos(\theta_1-\theta_2)-r_2\cos(\theta_1-\theta_3))/A-\\
&\quad \csc^2(\theta_1-\theta_3)\sin(\theta-\theta_1)(\cos(\theta-\theta_1)+\cot(\theta_1-\theta_3)\sin(\theta-\theta_1))/B)-\\
&\quad \cot(\theta_1-\theta_2)]/D,\\
c_{24} &= [-\sin(\theta_2-\theta_3)AC-\csc(\theta_1-\theta_2)\sin(\theta-\theta_2)]/D,\\
c_{25} &= [\csc(\theta_1-\theta_2)(r_1\cos(\theta-\theta_2)+\cot(\theta_1-\theta_2)(r_2\sin(\theta-\theta_1)-r_1\sin(\theta-\theta_2)))-\\
&\quad C((r_1\cos(\theta_2-\theta_3)-r_3\cos(\theta_1-\theta_2))/A-\\
&\quad \csc^2(\theta_2-\theta_3)\sin(\theta-\theta_2)(\cos(\theta-\theta_2)+\cot(\theta_2-\theta_3)\sin(\theta-\theta_2))/B)+\\
&\quad \cot(\theta_1-\theta_2)]/D,\\
c_{26} &= [\sin(\theta_1-\theta_3)C/A+\csc(\theta_1-\theta_2)\sin(\theta-\theta_1)]/D,\\
c_{27} &= [-C((r_2\cos(\theta_1-\theta_3)-r_1\cos(\theta_2-\theta_3))/A+\\
&\quad (\cot(\theta_1-\theta_3)\csc^2(\theta_1-\theta_3)\sin^2(\theta-\theta_1)+\\
&\quad \cot(\theta_2-\theta_3)\csc^2(\theta_2-\theta_3)\sin^2(\theta-\theta_2))/B)]/D,\\
c_{28} &= [-\sin(\theta_1-\theta_2)C/A]/D.
\end{aligned}
$$

Thus we conclude that $(\Delta\Theta', \Delta R')$ has a Gaussian distribution with $p.d.f$

$$p(\Delta\Theta', \Delta R') = \frac{1}{2\pi\sqrt{|\mathbf{\Sigma}|}}\exp[-\frac{1}{2}(\Delta\Theta', \Delta R')\mathbf{\Sigma}^{-1}(\Delta\Theta', \Delta R')^t] \qquad (5.8)$$

and covariance matrix

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma^2_{\theta'} & \sigma_{\theta' r'} \\ \sigma_{\theta' r'} & \sigma^2_{r'} \end{pmatrix},$$

where

$$\begin{aligned}
\sigma^2_{\theta'} &= (c_{11}^2 + c_{13}^2 + c_{15}^2 + c_{17}^2)\sigma^2_\theta, \\
\sigma^2_{r'} &= (c_{21}^2 + c_{23}^2 + c_{25}^2 + c_{27}^2)\sigma^2_\theta + (c_{22}^2 + c_{24}^2 + c_{26}^2 + c_{28}^2)\sigma^2_r + \\
&\quad 2(c_{21}c_{22} + c_{23}c_{24} + c_{25}c_{26} + c_{27}c_{28})\sigma_{\theta r}, \\
\sigma_{\theta' r'} &= (c_{11}c_{21} + c_{13}c_{23} + c_{15}c_{25} + c_{17}c_{27})\sigma^2_\theta + (c_{11}c_{22} + c_{13}c_{24} + c_{15}c_{26} + c_{17}c_{28})\sigma_{\theta r}.
\end{aligned}$$

5.3.4 Projective-Invariant Spread Function

From section 4.3.4, the invariant for projective transformations is as follows:

$$\begin{aligned}
\theta' &= \tan^{-1}(DEF/ABC), \\
r' &= \frac{-GCF}{\sqrt{(ABC)^2 + (DEF)^2}}
\end{aligned}$$

and

$$\begin{aligned}
A &= (r_3 \sin(\theta_1 - \theta_2) - r_2 \sin(\theta_1 - \theta_3) + r_1 \sin(\theta_2 - \theta_3)), \\
B &= (r_4 \sin(\theta - \theta_2) - r_2 \sin(\theta - \theta_4) + r \sin(\theta_2 - \theta_4)), \\
C &= (r_4 \sin(\theta_1 - \theta_3) - r_3 \sin(\theta_1 - \theta_4) + r_1 \sin(\theta_3 - \theta_4)), \\
D &= (-r_3 \sin(\theta - \theta_1) + r_1 \sin(\theta - \theta_3) - r \sin(\theta_1 - \theta_3)), \\
E &= (r_4 \sin(\theta_1 - \theta_2) - r_2 \sin(\theta_1 - \theta_4) + r_1 \sin(\theta_2 - \theta_4)), \\
F &= (r_4 \sin(\theta_2 - \theta_3) - r_3 \sin(\theta_2 - \theta_4) + r_2 \sin(\theta_3 - \theta_4)), \\
G &= (r_2 \sin(\theta - \theta_1) - r_1 \sin(\theta - \theta_2) + r \sin(\theta_1 - \theta_2)),
\end{aligned}$$

where (θ, r) is the parameter of the line being encoded and $(\theta_i, r_i)_{i=1,2,3,4}$ are the parameters of the basis lines.

Let $(\Delta\Theta, \Delta R)$, $(\Delta\Theta_i, \Delta R_i)_{i=1,2,3,4}$ and $(\Delta\Theta', \Delta R')$ be stochastic variables denoting respectively the perturbations of (θ, r), $(\theta_i, r_i)_{i=1,2,3,4}$ and the computed invariant (θ', r'). From section 5.2, we have

$$\Delta\Theta' = c_{11}\Delta\Theta + c_{12}\Delta R + c_{13}\Delta\Theta_1 + c_{14}\Delta R_1 + c_{15}\Delta\Theta_2 +$$

$$
\begin{aligned}
& c_{16}\Delta R_2 + c_{17}\Delta\Theta_3 + c_{18}\Delta R_3 + c_{19}\Delta\Theta_4 + c_{1,10}\Delta R_4, \\
\Delta R' \;=\; & c_{21}\Delta\Theta + c_{22}\Delta R + c_{23}\Delta\Theta_1 + c_{24}\Delta R_1 + c_{25}\Delta\Theta_2 + \\
& c_{26}\Delta R_2 + c_{27}\Delta\Theta_3 + c_{28}\Delta R_3 + c_{29}\Delta\Theta_4 + c_{2,10}\Delta R_4.
\end{aligned}
$$

Computing $c_{i,j}$'s, we obtain,

$$
\begin{aligned}
c_{11} &= [(-r_4\cos(\theta-\theta_2)+r_2\cos(\theta-\theta_4))I/B + \\
&\quad (-r_3\cos(\theta-\theta_1)+r_1\cos(\theta-\theta_3))EF]/HK, \\
c_{12} &= [\sin(\theta_2-\theta_4)I/B - \sin(\theta_1-\theta_3)EF]/HK, \\
c_{13} &= [-(r_4\cos(\theta_1-\theta_3)-r_3\cos(\theta_1-\theta_4))I/C + \\
&\quad (r_4\cos(\theta_1-\theta_2)-r_2\cos(\theta_1-\theta_4))DF - \\
&\quad (r_3\cos(\theta_1-\theta_2)-r_2\cos(\theta_1-\theta_3))I/A + \\
&\quad (r_3\cos(\theta-\theta_1)-r\cos(\theta_1-\theta_3))EF]/HK, \\
c_{14} &= [-\sin(\theta_3-\theta_4)I/C + \sin(\theta_2-\theta_4)DF - \\
&\quad \sin(\theta_2-\theta_3)I/A + \sin(\theta-\theta_3)EF]/HK, \\
c_{15} &= [-(-r_4\cos(\theta_2-\theta_3)+r_3\cos(\theta_2-\theta_4))DE + \\
&\quad (-r_4\cos(\theta_1-\theta_2)+r_1\cos(\theta_2-\theta_4))DF - \\
&\quad (-r_4\cos(\theta-\theta_2)+r\cos(\theta_2-\theta_4))I/B - \\
&\quad (-r_3\cos(\theta_1-\theta_2)+r_1\cos(\theta_2-\theta_3))I/A]/HK, \\
c_{16} &= [\sin(\theta_3-\theta_4)DE - \sin(\theta_1-\theta_4)DF + \\
&\quad \sin(\theta-\theta_4)I/B + \sin(\theta_1-\theta_3)I/A]/HK, \\
c_{17} &= [-(r_4\cos(\theta_2-\theta_3)-r_2\cos(\theta_3-\theta_4))DE + \\
&\quad (r_4\cos(\theta_1-\theta_3)-r_1\cos(\theta_3-\theta_4))I/C - \\
&\quad (r_2\cos(\theta_1-\theta_3)-r_1\cos(\theta_2-\theta_3))I/A + \\
&\quad (-r_1\cos(\theta-\theta_3)+r\cos(\theta_1-\theta_3))EF]/HK, \\
c_{18} &= [-\sin(\theta_2-\theta_4)DE + \sin(\theta_1-\theta_4)I/C - \\
&\quad \sin(\theta_1-\theta_2)I/A - \sin(\theta-\theta_1)EF]/HK, \\
c_{19} &= [-(-r_3\cos(\theta_2-\theta_4)+r_2\cos(\theta_3-\theta_4))DE +
\end{aligned}
$$

$$
\begin{aligned}
& (-r_3\cos(\theta_1-\theta_4)+r_1\cos(\theta_3-\theta_4))I/C+\\
& (r_2\cos(\theta_1-\theta_4)-r_1\cos(\theta_2-\theta_4))DF-\\
& (r_2\cos(\theta-\theta_4)-r\cos(\theta_2-\theta_4))I/B]/HK,\\
c_{1,10} &= [\sin(\theta_2-\theta_3)DE-\sin(\theta_1-\theta_3)I/C+\\
& \sin(\theta_1-\theta_2)DF-\sin(\theta-\theta_2)I/B]/HK,\\
c_{21} &= [(r_4\cos(\theta-\theta_2)-r_2\cos(\theta-\theta_4))HAC+\\
& (-r_3\cos(\theta-\theta_1)+r_1\cos(\theta-\theta_3))IEF]J/L^3+\\
& [-r_2\cos(\theta-\theta_1)+r_1\cos(\theta-\theta_2)]CF/L,\\
c_{22} &= [\sin(\theta_2-\theta_4)HAC-\sin(\theta_1-\theta_3)IEF]J/L^3-\\
& \sin(\theta_1-\theta_2)CF/L,\\
c_{23} &= [(r_4\cos(\theta_1-\theta_3)-r_3\cos(\theta_1-\theta_4))HAB+\\
& (r_3\cos(\theta_1-\theta_2)-r_2\cos(\theta_1-\theta_3))HBC+\\
& (r_4\cos(\theta_1-\theta_2)-r_2\cos(\theta_1-\theta_4))IDF+\\
& (r_3\cos(\theta-\theta_1)+r\cos(\theta_1-\theta_3))IEF]J/L^3-\\
& [r_4\cos(\theta_1-\theta_3)-r_3\cos(\theta_1-\theta_4)]GF/L+\\
& [r_2\cos(\theta-\theta_1)-r\cos(\theta_1-\theta_2)]CF/L,\\
c_{24} &= [\sin(\theta_3-\theta_4)HAB+\sin(\theta_2-\theta_3)HBC+\\
& \sin(\theta_2-\theta_4)IDF+\sin(\theta-\theta_3)IEF]J/L^3-\\
& \sin(\theta_3-\theta_4)GF/L+\sin(\theta-\theta_2)CF/L,\\
c_{25} &= [(-r_4\cos(\theta-\theta_2)+r\cos(\theta_2-\theta_4))HAC+\\
& (-r_3\cos(\theta_1-\theta_2)+r_1\cos(\theta_2-\theta_3))HBC+\\
& (r_4\cos(\theta_2-\theta_3)-r_3\cos(\theta_2-\theta_4))IDE-\\
& (r_4\cos(\theta_1-\theta_2)-r_1\cos(\theta_2-\theta_4))IDF]J/L^3-\\
& [r_4\cos(\theta_2-\theta_3)-r_3\cos(\theta_2-\theta_4)]GC/L+\\
& [-r_1\cos(\theta-\theta_2)+r\cos(\theta_1-\theta_2)]CF/L,\\
c_{26} &= [-\sin(\theta-\theta_4)HAC-\sin(\theta_1-\theta_3)HBC+\\
& \sin(\theta_3-\theta_4)IDE-\sin(\theta_1-\theta_4)IDF]J/L^3-\\
& \sin(\theta_3-\theta_4)GC/L-\sin(\theta-\theta_1)CF/L,
\end{aligned}
$$

$$
\begin{aligned}
c_{27} &= [-(r_4\cos(\theta_1-\theta_3)-r_1\cos(\theta_3-\theta_4))HAB + \\
&\quad (r_2\cos(\theta_1-\theta_3)-r_1\cos(\theta_2-\theta_3))HBC - \\
&\quad (r_4\cos(\theta_2-\theta_3)-r_2\cos(\theta_3-\theta_4))IDE + \\
&\quad (-r_1\cos(\theta-\theta_3)+r\cos(\theta_1-\theta_3))IEF]J/L^3 - \\
&\quad [-r_4\cos(\theta_2-\theta_3)+r_2\cos(\theta_3-\theta_4)]GC/L - \\
&\quad [-r_4\cos(\theta_1-\theta_3)+r_1\cos(\theta_3-\theta_4)]GF/L, \\
c_{28} &= [-\sin(\theta_1-\theta_4)HAB+\sin(\theta_1-\theta_2)HBC - \\
&\quad \sin(\theta_2-\theta_4)IDE-\sin(\theta-\theta_1)IEF]J/L^3 + \\
&\quad \sin(\theta_2-\theta_4)GC/L+\sin(\theta_1-\theta_4)GF/L, \\
c_{29} &= [(r_3\cos(\theta_1-\theta_4)-r_1\cos(\theta_3-\theta_4))HAB + \\
&\quad (r_2\cos(\theta-\theta_4)-r\cos(\theta_2-\theta_4))HAC + \\
&\quad (r_3\cos(\theta_2-\theta_4)-r_2\cos(\theta_3-\theta_4))IDE + \\
&\quad (r_2\cos(\theta_1-\theta_4)-r_1\cos(\theta_2-\theta_4))IDF]J/L^3 - \\
&\quad [r_3\cos(\theta_2-\theta_4)-r_2\cos(\theta_3-\theta_4)]GC/L - \\
&\quad [r_3\cos(\theta_1-\theta_4)-r_1\cos(\theta_3-\theta_4)]GF/L, \\
c_{2,10} &= [\sin(\theta_1-\theta_3)HAB+\sin(\theta-\theta_2)HAC + \\
&\quad \sin(\theta_2-\theta_3)IDE+\sin(\theta_1-\theta_2)IDF]J/L^3 - \\
&\quad \sin(\theta_2-\theta_3)GC/L-\sin(\theta_1-\theta_3)GF/L.
\end{aligned}
$$

and

$$H = ABC, \quad I = DEF, \quad J = GCF, \quad K = (1+I^2/H^2), \quad L = \sqrt{H^2+I^2}.$$

Thus we conclude that $(\Delta\Theta', \Delta R')$ has a Gaussian distribution with $p.d.f$

$$p(\Delta\Theta', \Delta R') = \frac{1}{2\pi\sqrt{|\mathbf{\Sigma}|}}\exp[-\frac{1}{2}(\Delta\Theta', \Delta R')\mathbf{\Sigma}^{-1}(\Delta\Theta', \Delta R')^t]$$

and covariance matrix

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma^2_{\theta'} & \sigma_{\theta' r'} \\ \sigma_{\theta' r'} & \sigma^2_{r'} \end{pmatrix},$$

where

$$\begin{aligned}
\sigma^2_{\theta'} &= (c^2_{11} + c^2_{13} + \cdots + c^2_{19})\sigma^2_\theta + (c^2_{12} + c^2_{14} + \cdots + c^2_{1,10})\sigma^2_r + \\
&\quad 2(c_{11}c_{12} + c_{13}c_{14} + \cdots + c_{19}c_{1,10})\sigma_{\theta r}, \\
\sigma^2_{r'} &= (c^2_{21} + c^2_{23} + \cdots + c^2_{29})\sigma^2_\theta + (c^2_{22} + c^2_{24} + \cdots + c^2_{2,10})\sigma^2_r + \\
&\quad 2(c_{21}c_{22} + c_{23}c_{24} + \cdots + c_{29}c_{2,10})\sigma_{\theta r}, \\
\sigma_{\theta' r'} &= (c_{11}c_{21} + c_{13}c_{23} + \cdots + c_{19}c_{29})\sigma^2_\theta + (c_{12}c_{22} + c_{14}c_{24} + \cdots + c_{1,10}c_{2,10})\sigma^2_r + \\
&\quad (c_{11}c_{22} + c_{13}c_{24} + \cdots + c_{19}c_{2,10} + c_{21}c_{12} + c_{23}c_{14} + \cdots + c_{29}c_{1,10})\sigma_{\theta r}.
\end{aligned}$$

Chapter 6

Bayesian Line Invariant Matching

Our recognition scheme builds upon the geometric hashing method, as reviewed in chapter 2, which also notes some weaknesses of the geometric hashing technique. Even when we use an improved Hough transform to detect line features in a degraded image, we still face the problem dealing with image noise and with the resulting problem of noise in the invariants used for hashing.

To minimize these noise effects, we need to derive a theoretically justified scheme for weighted voting among hash bins. We show how a weighted voting scheme can be used to cope with the problem. This is the aim of the probabilistic procedure which we now go on to describe.

6.1 A Measure of Matching between Scene and Model Invariants

In the geometric hashing method described in chapter 2, during recognition a computed scene invariant is used to index the geometric hash table and tally one vote for the entry retrieved. However, considering the presence of noise, a scene invariant can not exactly hit the hash table entry where its matching model invariant is supposed to reside. Gavrila and Groen [18] assume a bounded circular region around each hash bin and tally equal vote for all the bins within the region centered around the bin being hashed. However, our analysis of the noise effect on computed invariants in the preceding chapter shows that the perturbation of an invariant has an approximately Gaussian distribution with a

non-zero correlation coefficient. This implies that the region of hash space that would need to be accessed is an ellipse, instead of a circle, centered around the "true" value of the invariant. Moreover, this perturbation also depends on the basis and the line being encoded, which means that different invariants should be associated with different size, shape and orientation of hash space regions for error tolerance.

The nodes in hash space that more "closely match" the value of the invariant should get a higher weighted score than those which are far away, in a manner accurately representing the behavior of the noise which affects these invariants. Let inv_s be a scene invariant and let inv_{M_i} be a model invariant computed from a basis B_{M_i} and a line l_j of M_i. Bayes' theorem [16] allows us to precisely determine the probability that inv_s results from the presence of a certain $[M_i, B_{M_i}, l_j]$:

$$P([M_i, B_{M_i}, l_j]|inv_s) \quad = \quad P([M_i, B_{M_i}, l_j]) \, \frac{p(inv_s|[M_i, B_{M_i}, l_j])}{p(inv_s)}$$

where

- $P([M_i, B_{M_i}, l_j])$ is the *a priori* probability that M_i is embedded in the scene and B_{M_i} is selected to encode l_j. We assume that all the models are equally likely to appear and have the same number of features, which is a simplification that could be easily generalized. Thus $P([M_i, B_{M_i}, l_j])$ is the same for all [model, basis, line] combinations. (Note that this may not be quite true if the basis selection procedure favors long bases instead of selecting bases randomly.)

- $p(inv_s|[M_i, B_{M_i}, l_j])$ is the conditional $p.d.f.$ of observing inv_s given that M_i appears in the scene with B_{M_i} being selected and l_j being encoded. It consists of two parts: the spike induced by $[M_i, B_{M_i}, l_j]$ and a background distribution.

- $p(inv_s)$ is the $p.d.f.$ of observing inv_s, regardless of which $[M_i, B_{M_i}, l_j]$ induces it.

Assume that no two invariants of the same model and with a fixed basis locate near each other in hash space (i.e., model features are distinct and separate; If more than one model feature lands in approximately the same location, then the features should be coalesced into a single feature), inv_s can be close to at most one model invariant. Thus, given inv_s, the supporting evidence that M_i appears in the scene and the basis B_{M_i} is selected is then

$$P([M_i, B_{M_i}])|inv_s) \quad \approx \quad \max_j P([M_i, B_{M_i}, l_j]|inv_s)$$

$$\propto \quad \frac{\max_j \; p(inv_s|[M_i, B_{M_i}, l_j])}{p(inv_s)}. \tag{6.1}$$

6.2 Evidence Synthesis by Bayesian Reasoning

In the recognition stage, a certain scene basis B_s is probed and the invariants of all the remaining features with respect to this basis are computed. Each invariant provides various degrees of support for the existence of $[M_i, B_{M_i}]$ combinations, assuming B_s corresponds to B_{M_i}. If for every $[M_i, B_{M_i}]$ combination we synthesize support from all the invariants and hypothesize the one with maximum support, we in fact exercise a maximum likelihood approach to object recognition.

In order to synthesize support coming from different invariants, we have the following formula, based on a probability derivation using Bayes' theorem [12] :

$$\begin{aligned}
& \frac{P([M_i, B_{M_i}]|inv_{s_1}, \ldots, inv_{s_n})}{P([M_i, B_{M_i}])} \\
= \quad & \frac{p(inv_{s_1}, \ldots, inv_{s_n}|[M_i, B_{M_i}])}{p(inv_{s_1}, \ldots, inv_{s_n})} \\
= \quad & \frac{p(inv_{s_1}|[M_i, B_{M_i}]) \ldots p(inv_{s_n}|[M_i, B_{M_i}])}{p(inv_{s_1}, \ldots, inv_{s_n})} \qquad \text{(assuming conditional independence)} \\
= \quad & \frac{1}{p(inv_{s_1}, \ldots, inv_{s_n})} \frac{P([M_i, B_{M_i}]|inv_{s_1})\, p(inv_{s_1})}{P([M_i, B_{M_i}])} \cdots \frac{P([M_i, B_{M_i}]|inv_{s_n})\, p(inv_{s_n})}{P([M_i, B_{M_i}])} \\
= \quad & \frac{p(inv_{s_1}) \ldots p(inv_{s_n})}{p(inv_{s_1}, \ldots, inv_{s_n})} \prod_{k=1}^{n} \frac{P([M_i, B_{M_i}]|inv_{s_k})}{P([M_i, B_{M_i}])} \\
\propto \quad & \frac{p(inv_{s_1}) \ldots p(inv_{s_n})}{p(inv_{s_1}, \ldots, inv_{s_n})} \prod_{k=1}^{n} \frac{\max_j p(inv_{s_k}|[M_i, B_{M_i}, l_j]}{p(inv_{s_k})\, P([M_i, B_{M_i}])} \qquad \text{(by Eq. (6.1))} \\
= \quad & \frac{1}{p(inv_{s_1}, \ldots, inv_{s_n})} \prod_{k=1}^{n} \frac{\max_j p(inv_{s_k}|[M_i, B_{M_i}, l_j]}{P([M_i, B_{M_i}])}.
\end{aligned}$$

Thus

$$P([M_i, B_{M_i}]|inv_{s_1}, \ldots, inv_{s_n}) \propto \prod_{k=1}^{n} \max_j \; p(inv_{s_k}|[M_i, B_{M_i}, l_j]), \tag{6.2}$$

since $p(inv_{s_1}, \ldots, inv_{s_n})$ is independent of all the hypotheses and all $P([M_i, B_{M_i}])$'s are identical.

In the above derivation, we have assumed conditional independence among the invariants, i.e.,

$$p(inv_{s_1}, \ldots, inv_{s_n}|[M_i, B_{M_i}]) = \prod_{k=1}^{n} p(inv_{s_k}|[M_i, B_{M_i}]),$$

which means that the expected probability distribution of a scene invariant inv_{s_k} is independent of the other scene invariants. The intuition for this conditional independence to hold is that if the given inv_{s_k} does not belong to the embedded model M_i, it has nothing to do with $[M_i, B_{M_i}]$ and also other scene invariants; if the given inv_{s_k} belongs to the embedded model M_i, since geometric hashing applies transformation-invariant information, inv_{s_k} is destined to appear, given the hypothesis that B_{M_i} is selected. Thus inv_{s_k} is not affected by other invariants. A more formal argument is given in [28].

In computing $P([M_i, B_{M_i}]|inv_{s_1}, \ldots, inv_{s_n})$, we do not need to compute absolute probabilities but only relative probabilities. Since the logarithm function is monotonic, we can apply the logarithm to both sides of Eq. (6.2):

$$\log P([M_i, B_{M_i}]|inv_{s_1}, \ldots, inv_{s_n}) = \log C + \sum_{k=1}^{n} \log \max_j p(inv_{s_k}|[M_i, B_{M_i}, l_j]), \qquad (6.3)$$

turning products into sums.

6.3 The Algorithm

The pre-processing stage parallels our description of the preprocessing stage of the geometric hashing method reviewed in section 2.2. For a specific transformation group considered, we use the formulae given in section 4.3 to compute the invariants. We also compute the spread functions of these invariants using Eq. (5.7). Both pieces of information, together with the identity of [model, basis, line], are stored in a node in the hash entry indexed by each invariant. Understanding that the spread function is Gaussian, we store its covariance matrix as a representation of the function. We note that each node records information about a particular [model, basis, line] combination necessary for our recognition stage. The hash table is so arranged that the position of each entry in fact represents a quantized coordinate (θ, r) of hash space. In this way, range query can be easily performed.

In the recognition stage, lines are detected by the Hough transform and a subset of these lines are chosen as a basis for associating invariants with all remaining lines. Eq. (6.3)

is used to accumulate support from all these invariants ($\log C$ term, which is common to all the hypotheses, can be ignored). We note that a straightforward use of Eq. (6.3) runs through all the [model, basis, line] combinations and thus the computational cost can be immense. By exploiting the geometric hash table prepared in the pre-processing stage, for each invariant inv_{s_k} hashed into the hash space, we access only its nearby nodes. For each node accessed, we credit $\log p(inv_{s_k}|[M_i, B_{M_i}, l_j])$ to the node, whose initial score value is set to 0. In computing $\log p(inv_{s_k}|[M_i, B_{M_i}, l_j])$, the background distribution function, compared with the spike induced by $[M_i, B_{M_i}, l_j]$ is usually negligible. We will use logarithm of Eq. (5.7) to approximate $\log p(inv_{s_k}|[M_i, B_{M_i}, l_j])$. Typically this Gaussian $p.d.f.$ falls off rapidly. We thus approximately credit the same weighted score, $\log c$, to all those not in the neighborhood of the hash. Equivalently, we may instead credit $\log p(inv_{s_k}|[M_i, B_{M_i}, l_j]) - \log c$ to those nearby nodes accessed and keep intact those not in the neighborhood. After all the invariants are processed, the support for all the $[M_i, B_{M_i}]$ combinations are histogramed. The top few $[M_i, b_{M_i}]$'s with sufficiently high votes are verified by transforming and superimposing the model onto the scene. A quality measure, QM, defined as the ratio of visible boundary, is computed. If the QM is above a preset threshold (e.g. 0.5), the hypothesis passes the verification. (The verification can proceed in a hierarchical way by first verifying the existence of the basis and reject immediately if the basis fails verification.)

In the above process, each probe of a scene basis is hypothesized as an instance of a particular basis of a certain model. It is possible that the scene basis probed does not correspond to any basis of any model. In the following section, we give a probabilistic analysis of the number of probes needed to detect all the instances in the scene.

6.4 An Analysis of the Number of Probings Needed

Let there be n lines in the scene, $n = n_1 + n_2$, where n_2 lines are spurious. Let also there be M model instances in the scene. Assuming that the degree of occlusion of each instance is the same. Thus on average each model appearing in the scene has $\frac{n_1}{M}$ lines.

The probability of choosing a scene basis, consisting of k features, which happens to

correspond to a basis of a model is

$$p = \prod_{i=0}^{k-1} (\frac{\frac{n_1}{M} - i}{n - i}).$$

If we are to detect all M model instances at once, in the best case we have to probe only M scene bases from the scene. This happens when each time a basis is probed, it happens to correspond to a basis of a certain model and next time another basis is probed, it corresponds to a basis of another model. However, the probability for this to happen is p^M, which is obviously small. The probability of detecting exactly i different models after t trials (i.e., among the t trials, $t = i' + i'', i' \geq i$ probes cover bases of each of the i models and i'' probes correspond to no bases of any model) is

$$p_i = \left(\begin{array}{c} t \\ i \end{array} \right) i! p^i, \ i = 0, ..., M.$$

We may derive the lower bound of t by restricting $p_i \geq \epsilon$, $0 \leq \epsilon \leq 1$:

$$t(t-1) \cdots (t-i+1) \geq \epsilon / p^i.$$

Thus,

$$t^i > \epsilon / p^i \text{ or } t > \sqrt[i]{\epsilon}/p.$$

The probability that at least j different models are detected is then

$$q_j = 1 - \sum_{i=0}^{j-1} p_i.$$

In practice, we have to try more probes than theoretically predicted, because some probes, though corresponding to some model bases, are bad (e.g., too small, too big or too much perturbed) bases and do not result in a stable transformation that transforms model to properly fit its scene instance to pass verification.

Chapter 7

The Experiments

In this chapter, we test our approach using both synthesized and real images. We choose to implement the case of affine transformations. The group of affine transformations is a supergroup of both rigid and similarity transformations. In addition, affine transformations are often appropriate approximations to perspective transformations under suitable viewing situations (referring to p. 79 of [35]) and thus can be used in recognition algorithms as substitutes for more general perspective transformations. It should be noted that a similar approach can be applied to the cases of other transformations we discussed in chapter 4. Implementation details, with experimental results, are presented in the following.

7.1 Implementation of Affine Invariant Matching

To apply the procedure described in section 6.3 to the case of affine group, a triplet of lines is necessary to form a basis. Eq. (4.21) and Eq. (4.22) in section 4.3.3 are used to compute the invariant and the spread function of the invariant is given by Eq. (5.8) in section 5.3.3. Our formulae involve many trigonometric evaluations. To accelerate the process, a fine-grained look-up table of trigonometric functions is pre-computed.

Considering numerical stability, we should avoid prevent using small bases or large bases for invariant computing. In our implementation, we use 256×256 images. If any side of the basis is less than 20 pixels or all sides of the basis is greater than 500 pixels, we consider it infeasible.

The hash table is implemented as a 2-D array to represent the 2-D hash space. Since

during recognition, we have to consider a neighborhood of a hash entry when it is retrieved, the boundary of the hash table has to be treated specially: The θ-axis of the 2-D hash table should be viewed as *round-wrapped* with *flip*, since the invariant (θ, r) is mathematically equivalent to $(\theta - \pi, -r)$ (e.g., $(179°, 2)$ is equivalent to $(-1°, -2)$, thus neighboring, say,$(0°, -2)$).

During evidence synthesis, care should be taken to prevent multiple accumulation of weighted votes: For each probing of scene bases, each hash node, $[M_i, b_{M_i}, l_j]$, should receive a vote only once, since model feature l_j can not match more than two scene features simultaneously. We thus have to keep track of the score each hash node receives and take only the maximum weighted score the node has received.

We also have to reject *reflexion* cases. Reflexion transformations form a subgroup of affine transformations. However, a 2-D object, say a triangle with vertex-$1, 2, 3$ in clockwise sequence, preserves this sequence, even its shape being seriously skewed when viewed from a tilted camera. This problem is tackled by detecting the *orientation* (clockwise or counterclockwise) of a basis by its cross product. Thus no such false match between scene basis and model basis will be hypothesized.

Since we are dealing with highly occluded scenes, we have found that even if a hypothesis has passed verification (by verifying its boundary), it can still be a false alarm. By the viewpoint consistency principle [41], the locations of all object features in an image should be consistent with the projection from a single viewpoint. We may show that all 2-D objects lying on the same plane have an identical enlarging (or shrinking) ratio of area, if viewed from the same camera, assuming approximation of affine transformations to perspective transformations. The quantity of this ratio is $|\det(\mathbf{A})|$, where $\mathbf{A}$ is the skewing matrix of an affine transformation $\mathbf{T} = (\mathbf{A}, \mathbf{b})$. We call it the *skewing factor* and request that all the hypothesized instances have the same skewing factor. (Appendix-A provides the formula for computing $\mathbf{T}$ from a correspondence of a scene basis and a model basis.)

For each probe, after histograming of the weighted votes, the top few hypotheses are verified. If the verification is successful, the skewing factor of the alleged instance is computed and used to vote for a common skewing factor.

After sufficient number of bases are probed, which is determined by a statistical estimation procedure (see section 6.4), a global skewing factor, which is taken to be the

$|\det(\mathbf{A})|$ with highest number of votes, is obtained. Those hypothesized model instances voting for the global skewing factor are passed onto a disambiguation process which we go on to describe below.

It is possible that a scene line is shared by many model instances, some of which are spurious. We could refine the result by classifying the shared lines to belong to that model instance most strongly supported by the evidence. One possible technique for disambiguation is *relaxation* [54]. However, relaxation is a complicated process. Since each model instance is associated with its quality measure, QM, as a measure of the strength of evidence supporting its existence (this definition of QM favors long segments of a model instance and is similar to that used in [4] with the same justification). The following straightforward technique proves to work well.

We maintain two data structures, a QM-buffer and a frame-buffer, of the same size as the image. After initializing the QM-buffer and the frame-buffer, we process each candidate model instance in an arbitrary order by superimposing the candidate model instance onto the QM-buffer, then tracing the boundary of the model instance. For each position traced, if the QM of the current model instance is greater than that value stored in the QM-buffer, we write the *id* of this candidate model instance to the corresponding position of the frame-buffer and replace that value in the QM-buffer by the current QM.

After all candidate model instances have been processed, we recompute the QM of each candidate on the frame-buffer by considering only positions with the same *id* as this candidate. Those with the new QM's less than a preset threshold (e.g. 0.5) are removed from the list of candidate model instances.

7.2 Best Least-Squares Match

The transformation between a model and its scene instance can be recovered by the correspondence of the model basis and the scene basis alone. However, scene lines detected by the Hough transform are usually somewhat distorted due to noise. This results in distortion in computing the transformation. Usually this distorted transformation transforms a model to match its scene instance with basis lines matching each other perfectly while the other lines deviating from their correspondences more or less. Knowledge of additional line correspondences between a model and its scene instance can be used to improve the

accuracy of the computed transformation. In this following, we discuss several methods to minimize the errors.

Method 1

Treat each line as a point with coordinate (θ, r) in (θ, r)-space and minimize the squared distance between (θ, r) and its correspondence (θ', r').

Discussion

The problem of this method is that θ and r are of different metrics. To minimize the squared distance between a point (θ, r) and its correspondence (θ', r'), we implicitly assume equal weight on both θ and r.

Method 2

To circumvent the problem caused by Method 1, we note that a line (θ, r) can be uniquely represented by the point $(r\cos\theta, r\sin\theta)$, which is the projection of the origin onto the line. To match line (θ, r) to its correspondence (θ', r'), we try to minimize the squared distance between $(r\cos\theta, r\sin\theta)$ and $(r'\cos\theta', r'\sin\theta')$.

Discussion

The drawback of this method is its dependency on the origin. The nearer the line is to the origin, the more weight is on r (think of the special case when both lines pass through the origin in the image).

Method 3[1]

The models in a model base are usually finite in the sense that though they are modeled by lines, they in fact consist of segments. We can minimize the squared distance of the endpoints of the transformed model line segments to their corresponding scene lines in image space.

We derive in the following the closed-form formula for the case of affine transformations. A similar technique can be applied for the cases of other transformation groups.

[1] suggested by Professor Jaiwei Hong.

Specifically, assuming that we are looking for an affine match between n scene lines l_j and endpoints of n segments, $\mathbf{u}_{j1}$ and $\mathbf{u}_{j2}$, $j = 1, \ldots, n$, we would like to find the affine transformation $\mathbf{T} = (\mathbf{A}, \mathbf{b})$, such that the summations of the squared distances between the sequence $\mathbf{T}(\mathbf{u}_{j1})$ to l_j and $\mathbf{T}(\mathbf{u}_{j2})$ to l_j, $j = 1, \ldots, n$, is minimized:

$$E = \min_{\mathbf{T}} \sum_{j=1}^{n} (\text{distance of } \mathbf{T}(\mathbf{u}_{j1}) \text{ and } l_j)^2 + (\text{distance of } \mathbf{T}(\mathbf{u}_{j2}) \text{ and } l_j)^2.$$

Let line l_j be with parameter (θ_j, r_j) and endpoints $\mathbf{u}_{ji}$ be $(x_{ji}, y_{ji}), j = 1, \ldots, n$ and $i = 1, 2$. Also let $\mathbf{T} = (\mathbf{A}, \mathbf{b})$ such that

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \quad \text{and} \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}.$$

Then

$$\mathbf{T}(\mathbf{u}_{ji}) = (a_{11}x_{ji} + a_{12}y_{ji} + b_1, a_{21}x_{ji} + a_{22}y_{ji} + b_2)^t$$

and

$$E = \min_{\mathbf{T}} \sum_{j=1}^{n} ((\cos\theta_j, \sin\theta_j)\mathbf{T}(\mathbf{u}_{j1}) - r_j)^2 + ((\cos\theta_j, \sin\theta_j)\mathbf{T}(\mathbf{u}_{j2}) - r_j)^2.$$

To minimize E, we have to solve the following system of equations:

$$\frac{\partial E}{\partial a_{11}} = 0, \quad \frac{\partial E}{\partial a_{12}} = 0, \quad \frac{\partial E}{\partial a_{21}} = 0, \quad \frac{\partial E}{\partial a_{22}} = 0, \quad \frac{\partial E}{\partial b_1} = 0 \quad \text{and} \quad \frac{\partial E}{\partial b_2} = 0.$$

Since E is a quadratic function in each of its unknowns, the above is a system of linear equations with six unknowns. We can rewrite it in the matrix form as follows. (See Appendix-B for the expressions of m_{ij} and n_i, $i, j = 1, ..., 6$.)

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} & m_{15} & m_{16} \\ m_{21} & m_{22} & m_{23} & m_{24} & m_{25} & m_{26} \\ m_{31} & m_{32} & m_{33} & m_{34} & m_{35} & m_{36} \\ m_{41} & m_{42} & m_{43} & m_{44} & m_{45} & m_{46} \\ m_{51} & m_{52} & m_{53} & m_{54} & m_{55} & m_{56} \\ m_{61} & m_{62} & m_{63} & m_{64} & m_{65} & m_{66} \end{pmatrix} \begin{pmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \\ b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} n_1 \\ n_2 \\ n_3 \\ n_4 \\ n_5 \\ n_6 \end{pmatrix}$$

The six unknowns can be solved by *Cramer's rule* (see p.89 of [39]) as follows:

$$\begin{aligned} a_{11} &= \det(M_1)/\det(M), & a_{12} &= \det(M_2)/\det(M), \\ a_{21} &= \det(M_3)/\det(M), & a_{22} &= \det(M_4)/\det(M), \\ b_1 &= \det(M_5)/\det(M), & b_2 &= \det(M_6)/\det(M), \end{aligned}$$

where $M = [m_{ij}]_{i,j=1,\ldots,6}$ and $M_k = M$ with k-th column substituted by $[n_i]_{i=1,\ldots,6}^t$, for $k = 1, \ldots, 6$.

Discussion

We could to do a weighted sum of the squared distances. More specifically, the two squared distances provided by the endpoints of long segments get more weight than those of short segments.

7.3 Experimental Results and Observations

We have done a series of experiments using synthesized images containing polygonal objects, which are modeled by line features, from the model base consisting of twenty models shown in Figure 3.3. The way the synthesized images are generated has been explained in section 3.3.

Figure 7.1, Figure 7.2 and Figure 7.3 show three examples of the experiments on synthesized images. The perturbation of the lines detected by the Hough transform is obtained from statistics of extensive simulations on images of different levels of degradation. We also did experiments on real images containing objects from the same model base, with flakes scattered on the objects. Two examples (Figure 7.5 and Figure 7.6) are shown with best least-squares match using Method-3 discussed before.

Figure 7.1(a) shows a composite overlapping scene of model-0, 3 and 4 (note: model-0 appears twice), which are significantly skewed. Figure 7.1(b) is the result of Hough analysis applied to this scene. 50 lines were detected. The covariance matrix used for the deviation of the detected lines from true lines was $\Sigma = \begin{pmatrix} 0.003384 & -0.00624 \\ -0.00624 & 2.5198 \end{pmatrix}$. Figure 7.1(c) shows the model instances hypothesized in the recognition stage. 5000 probes were tried. Figure 7.1(d) shows the final result after disambiguation.

Figure 7.2(a) shows a composite overlapping scene of model-0, 1, 2, 4 and 9. Figure 7.2(b) is the result of Hough analysis applied to this scene. 60 lines were detected. The covariance matrix used for the deviation of the detected lines from true lines is the same as in Figure 7.1. Figure 7.2(c) shows the model instances hypothesized in the recognition stage. 5000 probes were tried. Figure 7.2(d) shows the final result after disambiguation.

Figure 7.3(a) shows a composite overlapping scene of model-1, 3, 4, 13 and 19. Figure 7.3(b) is the result of Hough analysis applied to this scene. 60 lines were detected. The covariance matrix used for the deviation of the detected lines from true lines is the same as in Figure 7.1. Figure 7.3(c) shows the model instances hypothesized in the recognition stage. 5000 probes were tried. Figure 7.3(d) shows the final result after disambiguation. (Note that model-19 was not detected, since half of its boundary is invisible.)

Figure 7.4(a) shows a composite overlapping scene of model-3, 12 and 15. Figure 7.4(b) is the result of Hough analysis applied to this scene. 60 lines were detected. The covariance matrix used for the deviation of the detected lines from true lines is the same as in Figure 7.1. Figure 7.4(c) shows the model instances hypothesized in the recognition stage. 5000 probes were tried. Figure 7.4(d) shows the final result after disambiguation.

Figure 7.5(a) shows a real image of model-0, 2, 3 and 4 with flakes scattered. Figure 7.5(b) is the result of edge detection using Boie-Cox edge detector [9]. Figure 7.5(c) shows the result of Hough analysis applied to this scene. 50 lines are detected. Figure 7.5(d) shows the recognition result before disambiguation. Figure 7.5(e) shows the recognition result after disambiguation. Figure 7.5(f) shows the result after best least-squares match.

Figure 7.6(a) shows a real image of model-1, 2 and 3 (note: model-3 appears twice) with flakes scattered. Figure 7.6(b) is the result of edge detection using also Boie-Cox edge detector. Figure 7.6(c) shows the result of Hough analysis applied to this scene. 50 lines are detected. Figure 7.6(d) shows the recognition result before disambiguation. Figure 7.6(e) shows the recognition result after disambiguation. Figure 7.6(f) shows the result after best least-squares match.

When doing the experiments, in our first attempt we let the candidate model instances vote for the global skewing factor without verifying them first. A wrong global skewing factor was usually obtained. The reason involves: (1) The image is so noisy that there are too many lines detected and irrelevant lines are often chosen as a basis; (2) An affine transformation is so "powerful" that it is not difficult to have an affine transformation that maps a quadruplet of lines to another quadruplet of lines if some error tolerance is allowed. Our algorithm was modified to let only those verified candidate model instances vote for the global skewing factor. The trade-off is that verification takes greater computational time.

We also found from the experiments that when doing probing, if the probed triplet

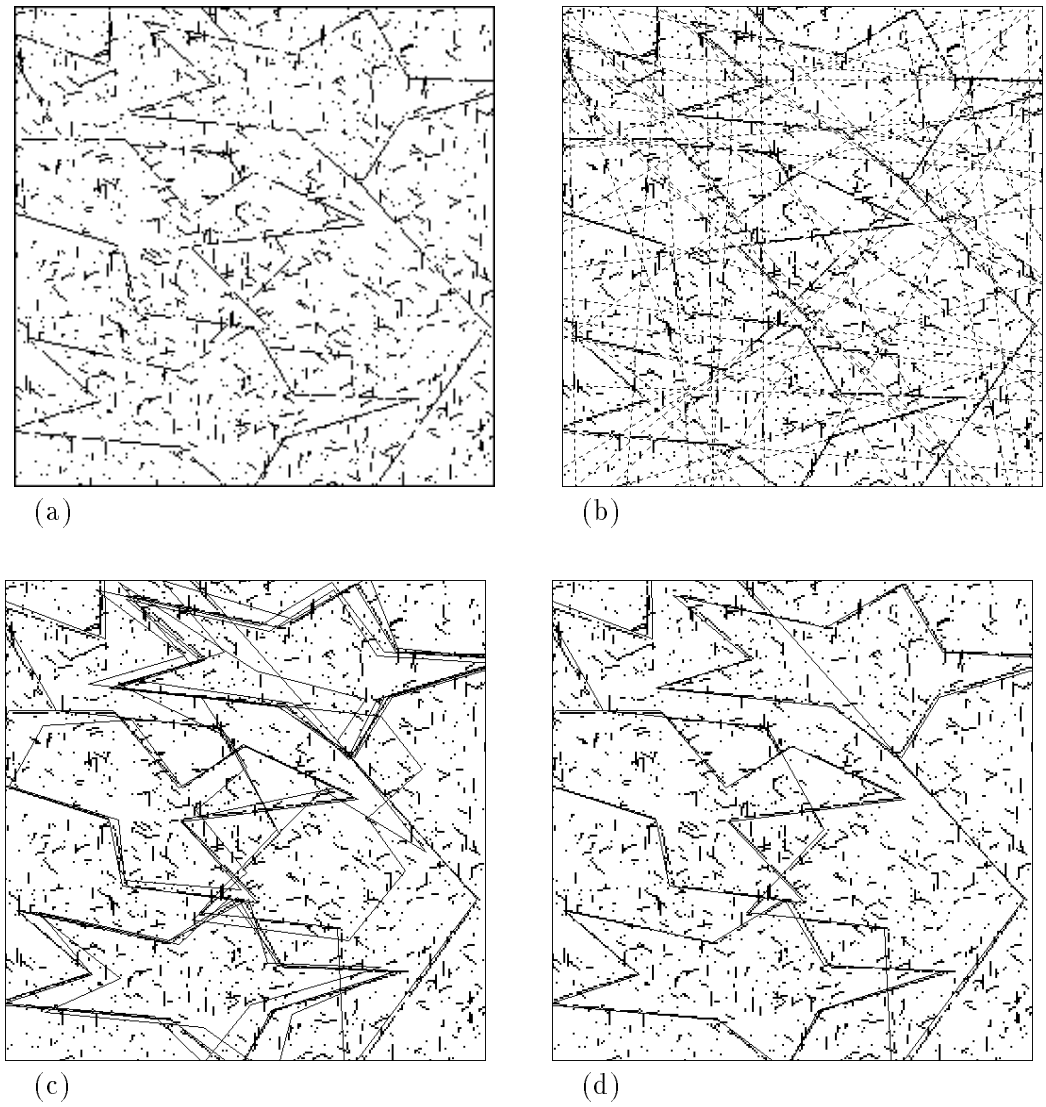


Figure 7.1: Experimental Example 1

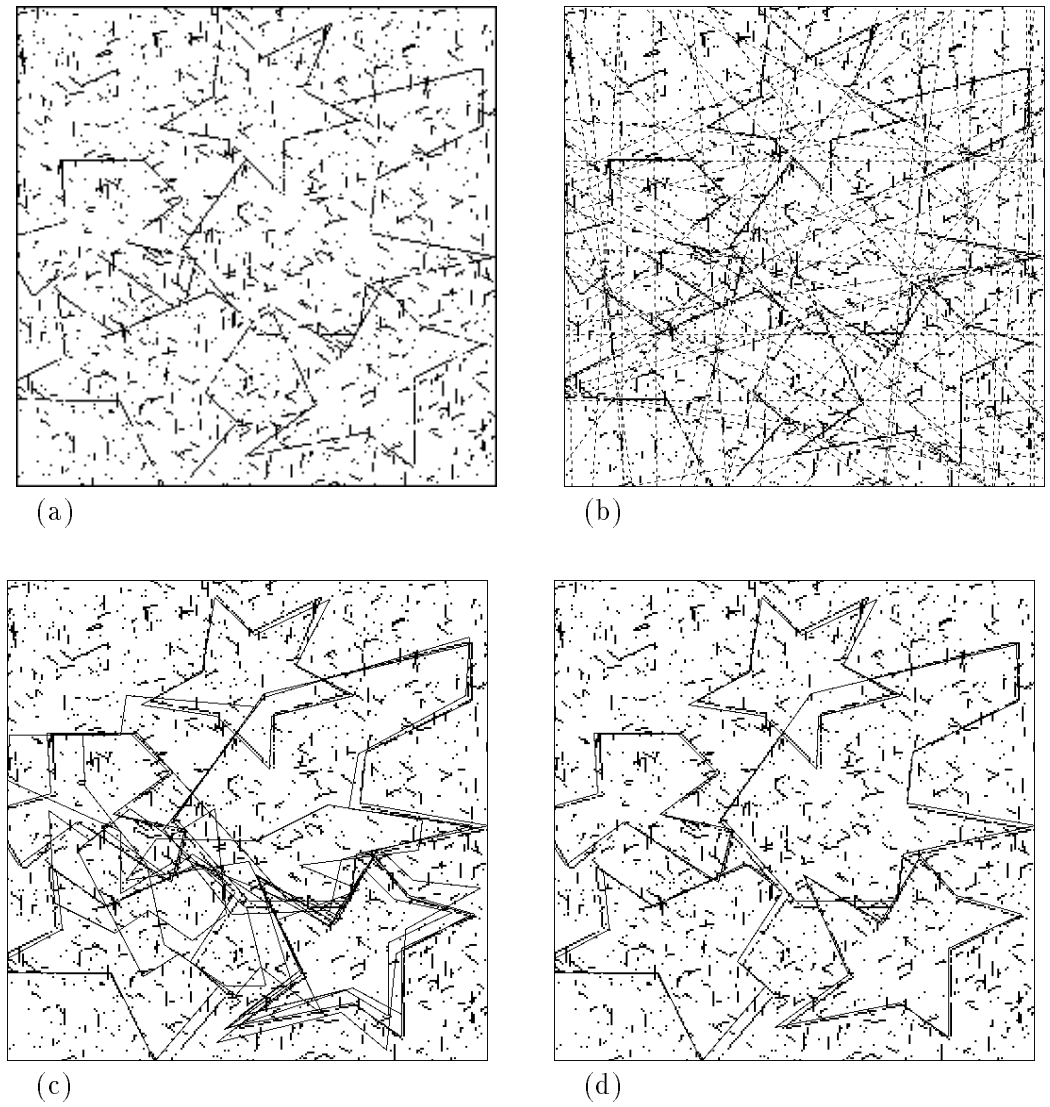


Figure 7.2: Experimental Example 2

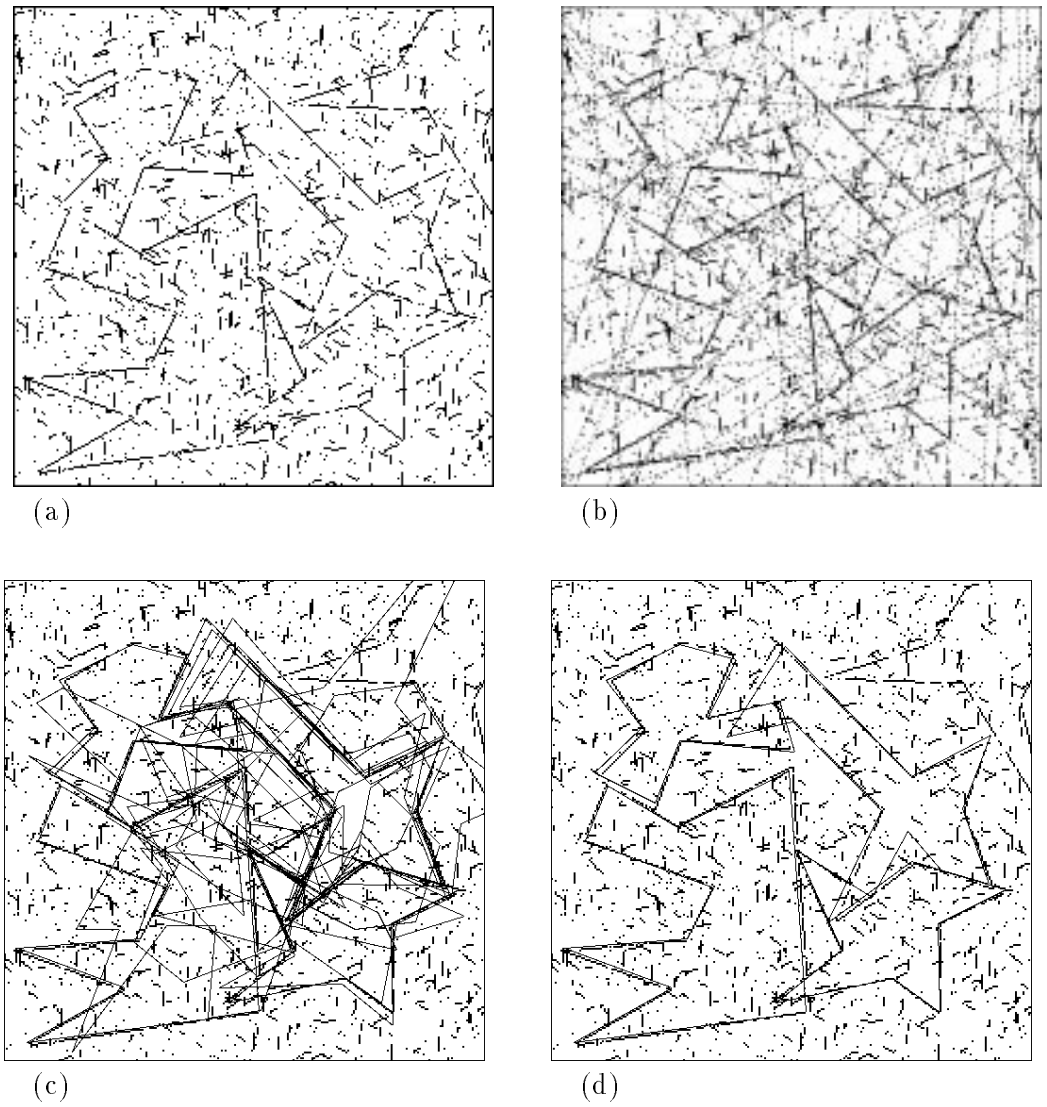


Figure 7.3: Experimental Example 3

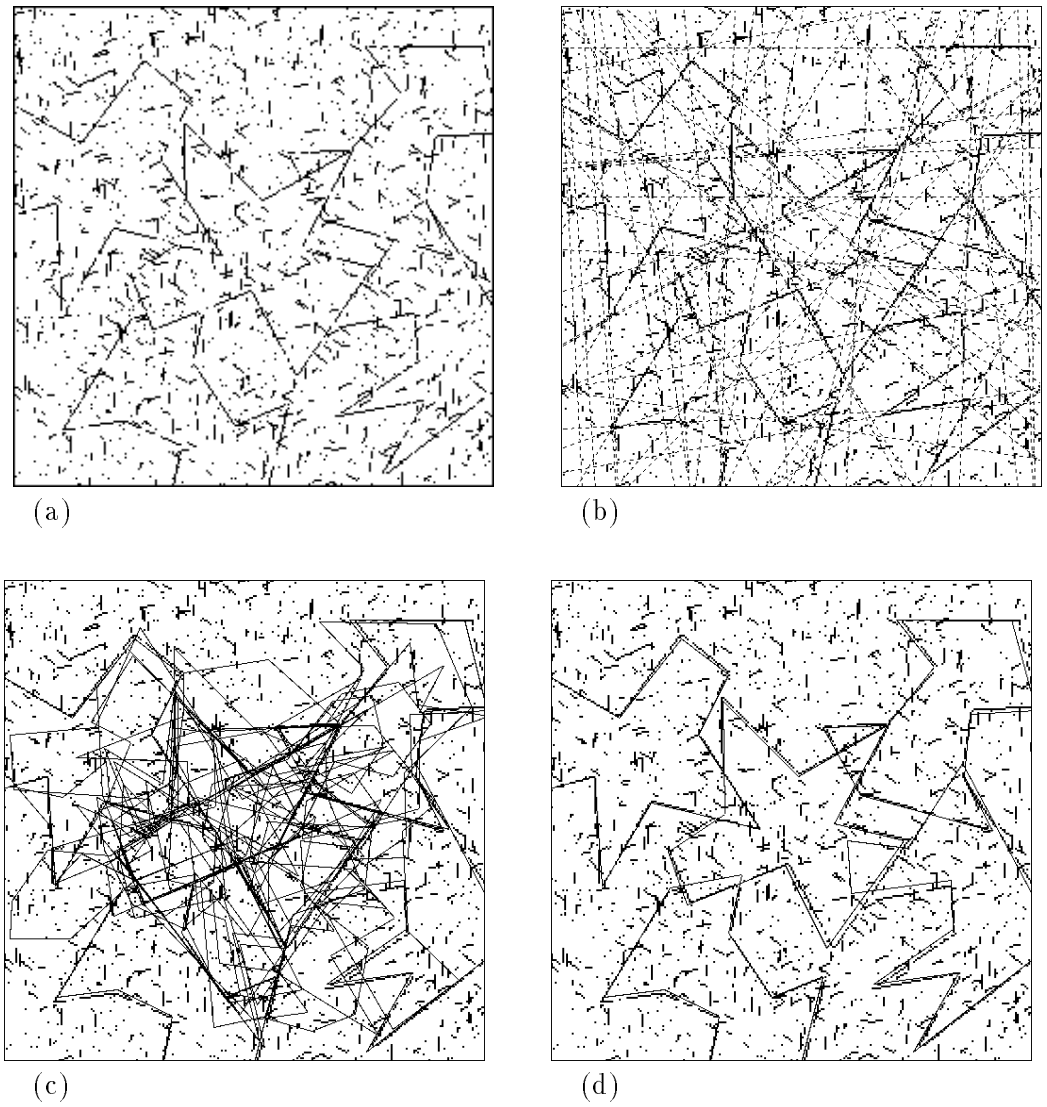
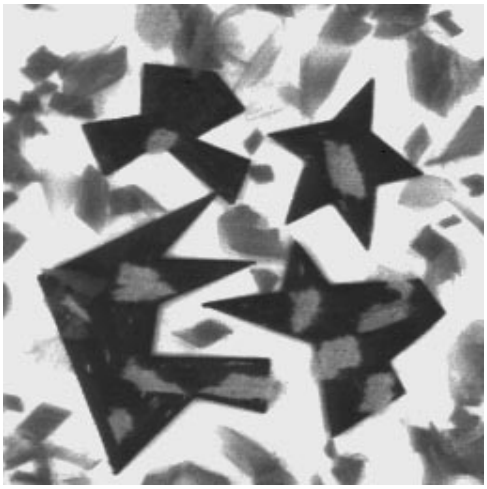
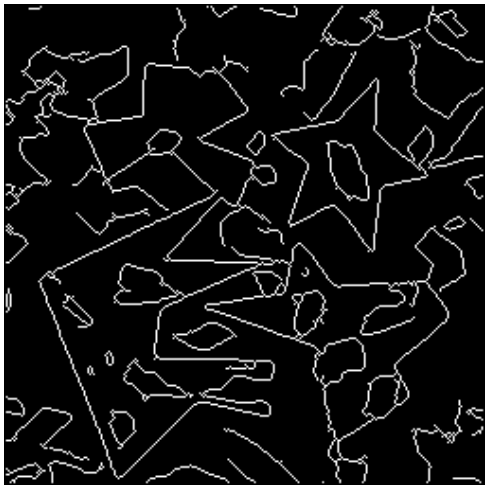


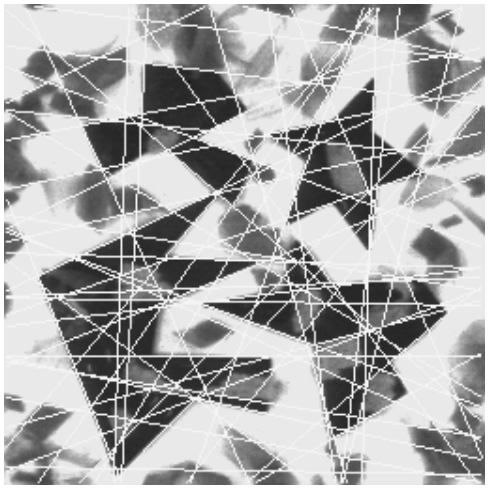
Figure 7.4: Experimental Example 4



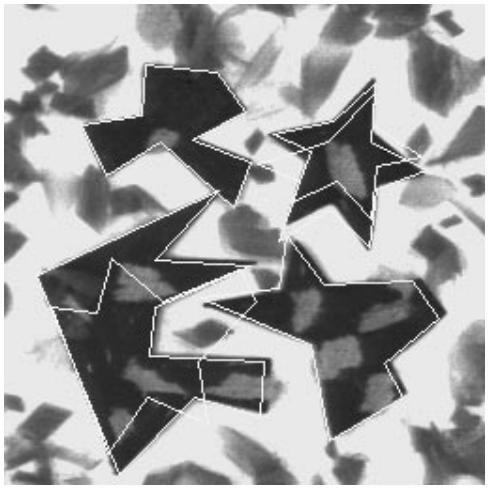
(a)



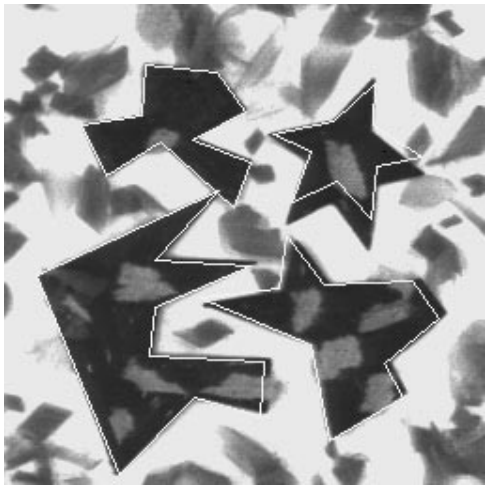
(b)



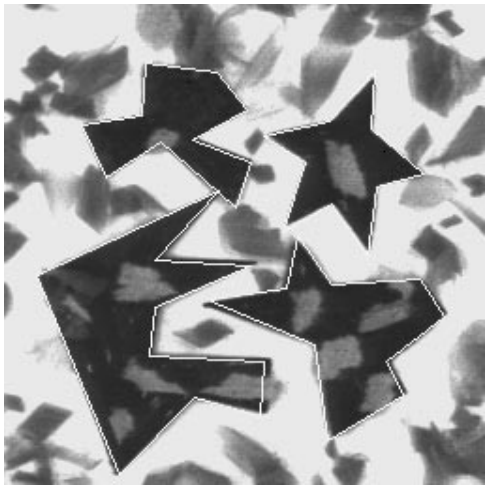
(c)



(d)



(e)

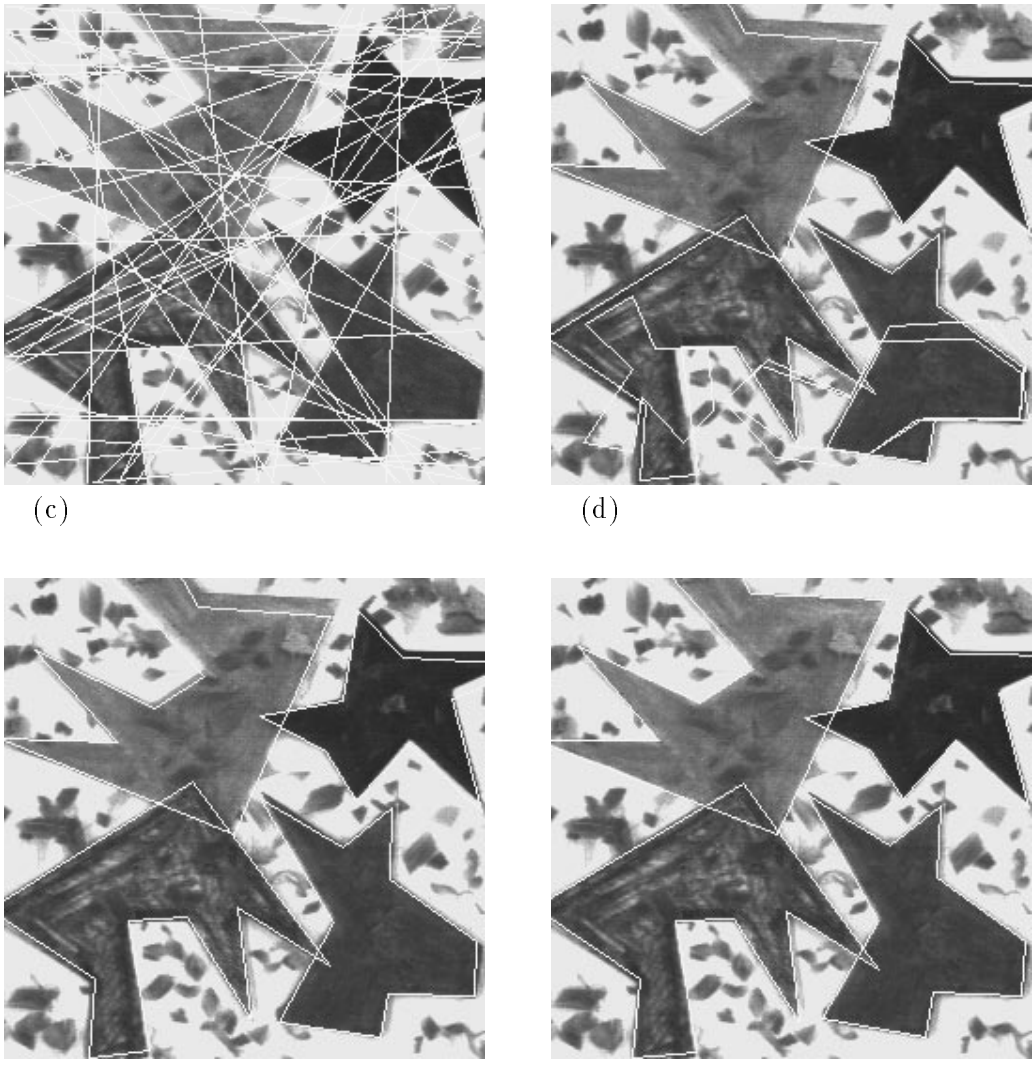


(f)

Figure 7.5: Experimental Example 5

(a)

(b)



(c) (d)

(e) (f)

Figure 7.6: Experimental Example 6

of lines happens to correspond to a basis of a certain model (i.e., a correct match), then this [model, basis] pair usually accumulates the highest weighted score among all others. If it does not rank the highest, the cause is usually due to insufficient lines detected for that particular model instance. However, its weighted vote is still among the highest few. The high-voted false alarms (i.e., matching this scene basis to a wrong [model, basis]) are usually rejected by verification. We conclude that this weighted voting scheme is quite effective.

The method described above does not assume any grouping of features, which is expected to greatly expedite the recognition stage (at least for scene basis selection). Lowe [40] first explicitly discussed the importance of grouping to recognition. However, if reliable segmentation is not available, intelligent grouping seems difficult. We note that in probing, for a given selected basis that does not correspond to any of model bases, it may happen that false alarms pass even the verification because of high noise in the scene. However, we point out here that statistically false alarms of such case disperse their votes for different skewing factors, while correct matches will accumulate their votes for the global skewing factor.

Chapter 8

Conclusions

8.1 Discussion and Summary

Geometric hashing is often compared to the generalized Hough transform. The main difference between them lies in the kind of "evidence" used for accumulation to generate hypotheses. While the generalized Hough transform accumulates "pose" evidence in a continuous infinite parameter space, geometric hashing accumulates "feature correspondence" evidence within discrete finite feature space of (model identifier, basis set)'s. However, the quantization problem exists not only in the generalized Hough transform technique. Geometric hashing implicitly transfers this problem to the preprocessing stage when making decisions about the quantization of hash space of invariants.

Grimson and Huttenlocher [21] analyze the performance of geometric hashing affine-invariant matching in the presence of noise on point features and give pessimistic predictions. They assert that it is impossible to construct a sparsely populated hash table due to the perturbation of the invariants computed from point features with uncertainty. Rigoutsos and Hummel [45] incorporate additive Gaussian noise into the model and analytically determine its effect on the computed invariants for the case where models are allowed to undergo similarity transformations. They have shown promising results.

However, there has not been heretofore an exploration of line features and their performance in seriously degraded intensity images. We have shown how the geometric hashing technique can be applied to line features by exploring line invariants under various geometric transformations, including rigid, similarity, affine and projective cases.

We have also done an analysis of noise sensitivity. Formulae that describe the spread of the computed invariants from the lines giving rise to these invariants are derived for the above geometric transformations. The basic underlying assumption here is that the perturbations of line features can be modeled by a Gaussian process. Since more image points are involved in line features, lines can be extracted by our improved Hough transform with good accuracy (small perturbation) even in seriously degraded images. This is the first of its kind for noise analysis of line features for geometric hashing.

The knowledge about noise sensitivity of the invariants allows us to give a weighted voting scheme for geometric hashing using line features. We have applied our technique for the case of affine transformations, which cover both rigid and similarity transformations and are often suitable substitutes for more general perspective transformations. It shows that the technique is noise resistant and suitable in an environment containing many occlusions.

8.2 Future Directions

We end this dissertation with a brief discussion of possible directions for future research.

It is obvious that the recognition stage can be easily parallelized. We may assign each basis to a processing element in a parallel computer, since each basis can be processed almost independently or assign each hash node a processing element. Rigoutsos [43] discuss a parallel implementation of geometric hashing with point features on the *Connection Machine* [26], which is equipped with $16K - 64K$ processors (for model CM-2). When the number of processors simultaneously needed exceeded the maximum number of physical processors in the machine, the machine can operate in a *virtual processor* mode. In this mode, a single processor emulates the work of several virtual processors, by serializing operations in time and partitioning the local memory associated with that processor.

With the advance of hardware techniques and the decrease of cost, parallel realization of image processing algorithms has become more and more popular.

Another topic to be explored is the intelligent fusion of multiple feature types. Since we assume no reliable segmentation is available, it seems that only line features can be extracted and used as out primitive feature. However, if the image can be well segmented and geometric features like points, segments, circles, ovals and so on can be reliably extracted, our technique can be helpful to design a suitable weighting function or to perform

hierarchical filtering by different feature types. This may require evaluation criteria for measuring the relative merit of different feature types and can be application-dependent.

Recognition of non-flat 3-D objects from 2-D images can also be a direction for extension.

A combined use of the transformation parameter space of the Hough transform and the invariant hash space of geometric hashing may possibly lead to a recognition system which can recognize parameterized objects.

Finally, the technique of "reasoning by parts" and the geometric hashing technique can benefit from each other. A complicated object usually can not be represented by only one primitive feature type. It may be more suitable to represent different parts of an object by different primitive feature types and recognize each part by the geometric hashing technique using different feature types.

Appendix A

Given a correspondence between pairs of triplets of lines in general position, the affine transformation $\mathbf{T} = (\mathbf{A},\mathbf{b})$, where $\mathbf{A}$ is a 2×2 non-singular skewing matrix and $\mathbf{b}$ is a 2×1 translation vector, can be uniquely determined such that $\mathbf{T}$ maps points of the first triplet of lines to their corresponding points of the second triplet of lines.

Let the first triplet of lines be $(\theta_i, r_i)^t_{i=1,2,3}$ and the second triplet of lines be $(\theta^{'}_i, r^{'}_i)^t_{i=1,2,3}$. We have the closed-form formula for $\mathbf{T} = (\mathbf{A},\mathbf{b})$ as follows:

$$\begin{aligned} \mathbf{A} &= \frac{1}{\det} \begin{pmatrix} a_{11} & a_{12} \\ -a_{21} & -a_{22} \end{pmatrix}, \\ \mathbf{b} &= \frac{1}{\det} \begin{pmatrix} -b_1 \\ b_2 \end{pmatrix}, \end{aligned}$$

where

$$\begin{aligned} a_{11} &= \cos\theta_3 \csc(\theta^{'}_1 - \theta^{'}_2) \sin(\theta_1 - \theta_2)(r^{'}_1 \sin\theta^{'}_2 - r^{'}_2 \sin\theta^{'}_1) + \\ &\quad \cos\theta_1 \csc(\theta^{'}_2 - \theta^{'}_3) \sin(\theta_2 - \theta_3)(r^{'}_2 \sin\theta^{'}_3 - r^{'}_3 \sin\theta^{'}_2) + \\ &\quad \cos\theta_2 \csc(\theta^{'}_3 - \theta^{'}_1) \sin(\theta_3 - \theta_1)(r^{'}_3 \sin\theta^{'}_1 - r^{'}_1 \sin\theta^{'}_3), \\ a_{21} &= \cos\theta_3 \csc(\theta^{'}_1 - \theta^{'}_2) \sin(\theta_1 - \theta_2)(r^{'}_1 \cos\theta^{'}_2 - r^{'}_2 \cos\theta^{'}_1) + \\ &\quad \cos\theta_1 \csc(\theta^{'}_2 - \theta^{'}_3) \sin(\theta_2 - \theta_3)(r^{'}_2 \cos\theta^{'}_3 - r^{'}_3 \cos\theta^{'}_2) + \\ &\quad \cos\theta_2 \csc(\theta^{'}_3 - \theta^{'}_1) \sin(\theta_3 - \theta_1)(r^{'}_3 \cos\theta^{'}_1 - r^{'}_1 \cos\theta^{'}_3), \\ a_{12} &= \sin\theta_3 \csc(\theta^{'}_1 - \theta^{'}_2) \sin(\theta_1 - \theta_2)(r^{'}_1 \sin\theta^{'}_2 - r^{'}_2 \sin\theta^{'}_1) + \\ &\quad \sin\theta_1 \csc(\theta^{'}_2 - \theta^{'}_3) \sin(\theta_2 - \theta_3)(r^{'}_2 \sin\theta^{'}_3 - r^{'}_3 \sin\theta^{'}_2) + \\ &\quad \sin\theta_2 \csc(\theta^{'}_3 - \theta^{'}_1) \sin(\theta_3 - \theta_1)(r^{'}_3 \sin\theta^{'}_1 - r^{'}_1 \sin\theta^{'}_3), \end{aligned}$$

$$
\begin{aligned}
a_{22} &= \sin\theta_3 \csc(\theta_1^{'} - \theta_2^{'}) \sin(\theta_1 - \theta_2)(r_1^{'} \cos\theta_2^{'} - r_2^{'} \cos\theta_1^{'}) + \\
&\quad \sin\theta_1 \csc(\theta_2^{'} - \theta_3^{'}) \sin(\theta_2 - \theta_3)(r_2^{'} \cos\theta_3^{'} - r_3^{'} \cos\theta_2^{'}) + \\
&\quad \sin\theta_2 \csc(\theta_3^{'} - \theta_1^{'}) \sin(\theta_3 - \theta_1)(r_3^{'} \cos\theta_1^{'} - r_1^{'} \cos\theta_3^{'}), \\
b_1 &= r_3 \csc(\theta_1^{'} - \theta_2^{'}) \sin(\theta_1 - \theta_2)(r_1^{'} \sin\theta_2^{'} - r_2^{'} \sin\theta_1^{'}) + \\
&\quad r_1 \csc(\theta_2^{'} - \theta_3^{'}) \sin(\theta_2 - \theta_3)(r_2^{'} \sin\theta_3^{'} - r_3^{'} \sin\theta_2^{'}) + \\
&\quad r_2 \csc(\theta_3^{'} - \theta_1^{'}) \sin(\theta_3 - \theta_1)(r_3^{'} \sin\theta_1^{'} - r_1^{'} \sin\theta_3^{'}), \\
b_2 &= r_3 \csc(\theta_1^{'} - \theta_2^{'}) \sin(\theta_1 - \theta_2)(r_1^{'} \cos\theta_2^{'} - r_2^{'} \cos\theta_1^{'}) + \\
&\quad r_1 \csc(\theta_2^{'} - \theta_3^{'}) \sin(\theta_2 - \theta_3)(r_2^{'} \cos\theta_3^{'} - r_3^{'} \cos\theta_2^{'}) + \\
&\quad r_2 \csc(\theta_3^{'} - \theta_1^{'}) \sin(\theta_3 - \theta_1)(r_3^{'} \cos\theta_1^{'} - r_1^{'} \cos\theta_3^{'}), \\
\det &= r_1 \sin(\theta_2 - \theta_3) + r_2 \sin(\theta_3 - \theta_1) + r_3 \sin(\theta_1 - \theta_2).
\end{aligned}
$$

Given the correspondence of the basis lines of a model object and its scene instance, we can use the above formula to transform the boundary of the model object onto the scene for verification.

Appendix B

The components of the matrix for the system of linear equations for best least-squares match are as follows:

$$m_{11} = \sum_{j=1}^{n} \cos^2\theta_j\,(x_{j1}^2 + x_{j2}^2), \qquad m_{31} = 2\sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1}^2 + x_{j2}^2),$$
$$m_{12} = \sum_{j=1}^{n} \cos^2\theta_j\,(x_{j1}y_{j1} + x_{j2}y_{j2}), \qquad m_{32} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1}y_{j1} + x_{j2}y_{j2}),$$
$$m_{13} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1}^2 + x_{j2}^2), \qquad m_{33} = \sum_{j=1}^{n} \sin^2\theta_j\,(x_{j1}^2 + x_{j2}^2),$$
$$m_{14} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1}y_{j1} + x_{j2}y_{j2}), \qquad m_{34} = \sum_{j=1}^{n} \sin^2\theta_j\,(x_{j1}y_{j1} + x_{j2}y_{j2}),$$
$$m_{15} = \sum_{j=1}^{n} \cos^2\theta_j\,(x_{j1} + x_{j2}), \qquad m_{35} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1} + x_{j2}),$$
$$m_{16} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1} + x_{j2}), \qquad m_{36} = \sum_{j=1}^{n} \sin^2\theta_j\,(x_{j1} + x_{j2}),$$

$$m_{21} = \sum_{j=1}^{n} \cos^2\theta_j\,(x_{j1}y_{j1} + x_{j2}y_{j2}), \qquad m_{41} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1}y_{j1} + x_{j2}y_{j2}),$$
$$m_{22} = \sum_{j=1}^{n} \cos^2\theta_j\,(y_{j1}^2 + y_{j2}^2), \qquad m_{42} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(y_{j1}^2 + y_{j2}^2),$$
$$m_{23} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1}y_{j1} + x_{j2}y_{j2}), \qquad m_{43} = \sum_{j=1}^{n} \sin^2\theta_j\,(x_{j1}y_{j1} + x_{j2}y_{j2}),$$
$$m_{24} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(y_{j1}^2 + y_{j2}^2), \qquad m_{44} = \sum_{j=1}^{n} \sin^2\theta_j\,(y_{j1}^2 + y_{j2}^2),$$
$$m_{25} = \sum_{j=1}^{n} \cos^2\theta_j\,(y_{j1} + y_{j2}), \qquad m_{45} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(y_{j1} + y_{j2}),$$
$$m_{26} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(y_{j1} + y_{j2}), \qquad m_{46} = \sum_{j=1}^{n} \sin^2\theta_j\,(y_{j1} + y_{j2}),$$

$$m_{51} = \sum_{j=1}^{n} \cos^2\theta_j\,(x_{j1} + x_{j2}), \qquad m_{61} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1} + x_{j2}),$$
$$m_{52} = \sum_{j=1}^{n} \cos^2\theta_j\,(y_{j1} + y_{j2}), \qquad m_{62} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(y_{j1} + y_{j2}),$$
$$m_{53} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(x_{j1} + x_{j2}), \qquad m_{63} = \sum_{j=1}^{n} \sin^2\theta_j\,(x_{j1} + x_{j2}),$$
$$m_{54} = \sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j\,(y_{j1} + y_{j2}), \qquad m_{64} = \sum_{j=1}^{n} \sin^2\theta_j\,(y_{j1} + y_{j2}),$$
$$m_{55} = 2\sum_{j=1}^{n} \cos^2\theta_j, \qquad m_{65} = 2\sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j,$$
$$m_{56} = 2\sum_{j=1}^{n} \cos\theta_j\,\sin\theta_j, \qquad m_{66} = 2\sum_{j=1}^{n} \sin^2\theta_j$$

$$n_1 = \sum_{j=1}^{n} \cos\theta_j \, r_j(x_{j1} + x_{j2}), \qquad n_3 = \sum_{j=1}^{n} \sin\theta_j \, r_j(x_{j1} + x_{j2}),$$
$$n_2 = \sum_{j=1}^{n} \cos\theta_j \, r_j(y_{j1} + y_{j2}), \qquad n_4 = \sum_{j=1}^{n} \sin\theta_j \, r_j(y_{j1} + y_{j2}),$$
$$n_5 = 2\sum_{j=1}^{n} \cos\theta_j \, r_j, \qquad n_6 = 2\sum_{j=1}^{n} \sin\theta_j \, r_j.$$

Bibliography

[1] A. Aho, J. Hopcroft, and J. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1984.

[2] K. Arbter, W. E. Snyder, H. Burkhardt, and Hirzinger. Application of Affine-Invariant Fourier Descriptors to Recognition of 3-D Objects. *IEEE Trans. on PAMI*, 12(7):640–647, 1990.

[3] L. P. Arkin, E. M .and Chew, D. P. Huttenlocher, K. Kedem, and J.S.B. Mitchell. An Efficiently Computable Metric for Comparing Polygonal Shapes. Technical Report TR 89-1007, Computer Science Dept., Cornell University, 1989.

[4] N. Ayache and O. D. Faugeras. HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects. *IEEE Trans. on PAMI*, 8(1):44–54, 1986.

[5] D. H. Ballard. Generalizing the Hough Transform to Detect Arbitrary Shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[6] H. G. Barrow and J. M. Tenenbaum. *Computational Vision*. Prentice-Hall, 1981.

[7] P. J. Besl and R. C. Jain. Three-Dimensional Object Recognition. *ACM Computing Surveys*, 17(1):75–154, 1985.

[8] T. O. Binfold. Survey of Model-Based Image Analysis Systems. *The Int. J. of Robotics Research*, 1(1):18–64, 1982.

[9] R. Boie and I. Cox. Two Dimensional Optimum Edge Recognition using Matched and Weiner Filter for Machine Vision. In *Proc. of the IEEE Int. Conf. on Computer Vision*, 1987.

[10] R. C. Bolles and R. A. Cain. Recognizing and Locating Partially Visible Objects: The Local-Feature-Focus Method. *The Int. J. of Robotics Research*, 1(3):57–82, 1982.

[11] R. A. Brooks. Model-Based Three-Dimensional Interpretations of Two-Dimensional Images. *IEEE Trans. on PAMI*, 5(2):140–149, 1983.

[12] E. Charniak and D. McDermott. *Introduction to Artificial Intelligence.* Addison-Wesley, 1985.

[13] R. T. Chin and C. R. Dyer. Model-Based Recognition in Robot Vision. *ACM Computing Surveys*, 18(1):67–108, 1986.

[14] P. R. Cohen and E. A. Feigenbaum (Eds.). *The Handbook of Artificial Intelligence, Vol. III.* William Kaufmann, 1982.

[15] R. O. Duda and P. E. Hart. Use of the Hough Transform to Detect Lines and Curves in Pictures. *Communication of ACM*, 15:11–15, 1972.

[16] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.

[17] P. J. Flynn and A. K. Jain. 3-D Object Recognition using Invariant Feature Indexing of Interpretation Table. *J. of Computer Vision, Graphics and Image Processing: Image Understanding*, 55(2):119–129, 1992.

[18] D. Gavrila and F. Groen. 3-D Object Recognition from 2-D Images Using Geometric Hashing. *Pattern Recognition Letters*, 13(4):263 – 278, 1992.

[19] P. G. Gottschalk, J. L. Turney, and T. N. Mudge. Efficient Recognition of Partially Visible Objects using a Logarithmic Complexity Matching Technique. *The Int. J. of Robotics Research*, 8(6):110–140, 1989.

[20] W. E. L. Grimson. On the Recognition of Parameterized 2-D Objects. *Int. J. of Computer Vision*, 2(4):353–372, 1989.

[21] W. E. L. Grimson and D. P. Huttenlocher. On the Sensitivity of Geometric Hashing. In *Proc. of the IEEE Int. Conf. on Computer Vision*, pages 334–338, 1990.

[22] W. E. L. Grimson and D. P. Huttenlocher. On the Sensitivity of the Hough Transform for Object Recognition. *IEEE Trans. on PAMI*, 12(3):255–274, 1990.

[23] W. E. L. Grimson and T. Lozano-Pérez. Localizing Overlapping Parts by Searching the Interpretation Tree. *IEEE Trans. on PAMI*, 9(4):469–482, 1987.

[24] A. Gueziec and N. Ayache. New Developments on Geometric Hashing for Curve Matching. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 703–704, 1993.

[25] Y. C. Hecker and R. M. Bolle. Invariant Feature Matching in Parameter Space with Application to Line Features. Technical Report RC 16447, IBM T.J.Watson Research Center, 1991.

[26] D. Hillis. *The Connection Machine*. MIT Press, 1985.

[27] J. Hong and X. Tan. Recognize the Similarity between Shapes under Affine Transformation. In *Proc. of the IEEE Int. Conf. on Computer Vision*, pages 489–493, 1988.

[28] R. Hummel and Rigoutsos I. Geometric Hashing as a Bayesian Maximum Likelihood Object Recognition Method. *Int. J. of Computer Vision*. Currently under review.

[29] R. Hummel and H. Wolfson. Affine Invariant Matching. In *Proc. of the DARPA IU Workshop*, pages 351–364, 1988.

[30] D. P. Huttenlocher and S. Ullman. Object Recognition using Alignment. In *Proc. of the IEEE Int. Conf. on Computer Vision*, pages 102–111, 1987.

[31] D. P. Huttenlocher and S. Ullman. Recognizing Solid Objects by Alignment with an Image. *Int. J. of Computer Vision*, 5(2):195–212, 1990.

[32] J. Illingworth and J. Kittler. A Survey of the Hough Transform. *J. of Computer Vision, Graphics and Image Processing*, 44:87–116, 1988.

[33] A. Kalvin, E. Schonberg, J. T. Schwartz, and M. Sharir. Two Dimensional Model Based Boundary Matching using Footprints. *The Int. J. of Robotics Research*, 5(4):38–55, 1986.

[34] E. Kishon. *Use of Three Dimensional Curves in Computer Vision*. PhD thesis, Computer Science Dept., Courant Institute of Mathematical Sciences, New York University, 1989.

[35] F. Klein. *Elementary Mathematics from an Advanced Standpoint ; Geometry.* Macmillan, 1925.

[36] Y. Lamdan. *Object Recognition by Geometric Hashing.* PhD thesis, Computer Science Dept., Courant Institute of Mathematical Sciences, New York University, 1989.

[37] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson. Object Recognition by Affine Invariant Matching. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 335–344, 1988.

[38] Y. Lamdan and H. J. Wolfson. On the Error Analysis of Geometric Hashing. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 22 – 27, 1991.

[39] S. Lang. *Linear Algebra.* Addison-Wesley, 1966.

[40] D. G. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer, 1985.

[41] D. G. Lowe. The Viewpoint Consistency Constraint. *Int. J. of Computer Vision*, 1(1):57–72, 1987.

[42] D. G. Lowe. Three-dimensional Object Recognition from Single Two-dimensional Images. *Artificial Intelligence*, 31:355–395, 1987.

[43] I. Rigoutsos. *Massively Parallel Bayesian Object Recognition.* PhD thesis, Computer Science Dept., Courant Institute of Mathematical Sciences, New York University, 1992.

[44] I. Rigoutsos and R. Hummel. A Bayesian Approach to Model Matching with Geometric Hashing. *J. of Computer Vision, Graphics and Image Processing: Image Understanding.* Currently under review.

[45] I. Rigoutsos and R. Hummel. Robust Similarity Invariant Matching in the Presence of Noise. In *Proc. of the 8th Israeli Conf. on Artificial Intelligence and Computer Vision*, 1991.

[46] I. Rigoutsos and R. Hummel. Massively Parallel Model Matching: Geometric Hashing on the Connection Machine. *IEEE Computer: Special Issue on Parallel Processing for Computer Vision and Image Understanding*, 1992.

[47] I. Rigoutsos and R. A. Hummel. Implementation of Geometric Hashing on the Connection Machine. In *IEEE Workshop on Directions in Aut. CAD-Based Vision*, 1991.

[48] T. Risse. The Hough Transform for Line Recognition: Complexity of Evidence Accumulation and Cluster Detection. *J. of Computer Vision, Graphics and Image Processing*, 46:327–345, 1989.

[49] J. T. Schwartz and M. Sharir. Identification of Partially Obscured Objects in Two Dimensions by Matching of Noisy 'Characteristic Curves'. *The Int. J. of Robotics Research*, 6(2):29–44, 1987.

[50] G. Stockman. Object Recognition and Localization via Pose Clustering. *J. of Computer Vision, Graphics and Image Processing*, 40:361–387, 1987.

[51] J. M. Tenenbaum and H. G. Barrow. A Paradigm for Integrating Image Segmentation and Interpretation. In *Proc. of the Int. Conf. on Pattern Recognition*, pages 504–513, 1976.

[52] D. W. Thompson and J. L. Mundy. Three-Dimensional Model Matching from an Unconstrained Viewpoint. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 208–220, 1987.

[53] L. W. Tucker, C. R. Feynman, and D. M. Fritzsche. Object Recognition using the Connection Machine. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 871–878, 1988.

[54] S. W. Zucker, R. Hummel, and A. Rosenfeld. An Application of Relaxation Labeling to Line and Curve Enhancement. *IEEE Trans. on Computers*, 26(4):394–403, 1977.