An Analysis of the Effect of Gaussian Error in Object Recognition

by

Karen Beth Sarachik

B.A., Barnard College (1983)
S.M., Massachusetts Institute of Technology (1989)

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

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 1994

An Analysis of the Effect of Gaussian Error in Object Recognition

by
Karen Beth Sarachik

Submitted to the Department of Electrical Engineering and Computer Science
on January 7, 1994, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

In model based recognition the problem is to locate an instance of one or several known objects in an image. The problem is compounded in real images by the presence of clutter (features not arising from the model), occlusion (absence in the image of features belonging to the model), and sensor error (displacement of features from their actual location). Since the locations of image features are used to hypothesize the object's pose in the image, these errors can lead to "false negatives", failures to recognize the presence of an object in the image, and "false positives", in which the algorithm incorrectly identifies an occurrence of the object when in fact there is none. This may happen if a set of features not arising from the object are located such that together they "look like" the object being sought. The probability of either of these events occurring is affected by parameters within the recognition algorithm, which are almost always chosen in an ad-hoc fashion. The implications of the parameter values for the algorithm's likelihood of producing false negatives and positives are usually not understood explicitly.

To address the problem, we have explicitly modelled the noise and clutter that occurs in the image. In a typical recognition algorithm, hypotheses about the position of the object are tested against the evidence in the image, and an overall score is assigned to each hypothesis. We use a statistical model to determine what score a correct or incorrect hypothesis is likely to have. We then use standard binary hypothesis testing techniques to decide the difference between correct and incorrect hypotheses. Using this approach we can compare algorithms and noise models, and automatically choose values for internal system thresholds to minimize the probability of making a mistake. Our analysis applies equally well to both the alignment method and geometric hashing.

Thesis Supervisor: W. Eric L. Grimson
Title: Associate Professor of Computer Science and Electrical Engineering

Acknowledgments

I have spent many years at the MIT AI Lab, and have many people to thank for my association with them during that time.

Eric Grimson has been a patient and supportive advisor, and helped me start out on the original problem. This thesis was improved tremendously by the meticulous reading and careful comments of Professor Jeff Shapiro.

I have benefited greatly from technical discussions with Sandy Wells, Randy Godwin, Marina Meila, David Clemens, David Jacobs, and Mike Bolotski, as well as from their friendship.

My friendship with many other lab members was a constant source of support, including David Siegel, Nancy Pollard, Ray Hirschfeld, Marty Hiller, Ian Horswill, Paul Viola, Tina Kapur, Patrick Sobolvarro, Lisa Dron, Ruth Schoenfeld, Anita Flynn, Sundar Narasimhan, Steve White, and in particular, Jonathan Amsterdam, Nomi Harris, and Barb Moore. I would also like to thank Mary Kocol, Michele Popper and Rooth Morgenstein for being extra-lab friends.

Lastly, I would like to thank my family, and Joel.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. This work has been supported in part by NSF contract IRI-8900267. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-91-J-4038.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In order to build machines capable of interacting intelligently in the real world, they must be capable of perceiving and interpreting their environs. To do this, they must be equipped with powerful sensing tools such as humans have, one of which is Vision — the ability to interpret light reflected from the scene to the eye. Computer Vision is the field which addresses the question of interpreting the light reflected from the scene as recorded by a camera. Humans are far more proficient at this visual interpretation task than any computer vision system yet built. Lest the reader think that this is because the human eye may somehow perceive more information from the scene than a camera can, we note that a human can also interpret camera images that a computer cannot — that is, a human outperforms the computer at visual interpretation tasks even when limited to the same visual input.

Model based recognition is a branch of computer vision whose goal is to detect the presence and position in the scene of one or more objects that the computer knows about beforehand. This capability is necessary for many tasks, though not all. For example, if the task is to navigate from one place to another, then the goal of the visual interpretation is to yield the positions of obstacles, regardless of their identity. If the task is to follow something, then the goal of the interpretation is to detect motion. However, if the task is to count trucks that pass through an intersection at a particular time of day, then the goal of the interpretation is to recognize trucks as opposed to any other vehicle.

Model based recognition is generally broken down into the following conceptual modules (Figure 1-1). There is a database of models, and each known model is represented in the database by a set of features (for example, straight edges, corners, etc.). In order to recognize any of the objects in a scene, an image of the scene is taken by a camera, some sort of feature extraction is done on the image, and then the features from the image are fed into a recognition algorithm along with model features retrieved from the model database. The task of the recognition algorithm is to de-

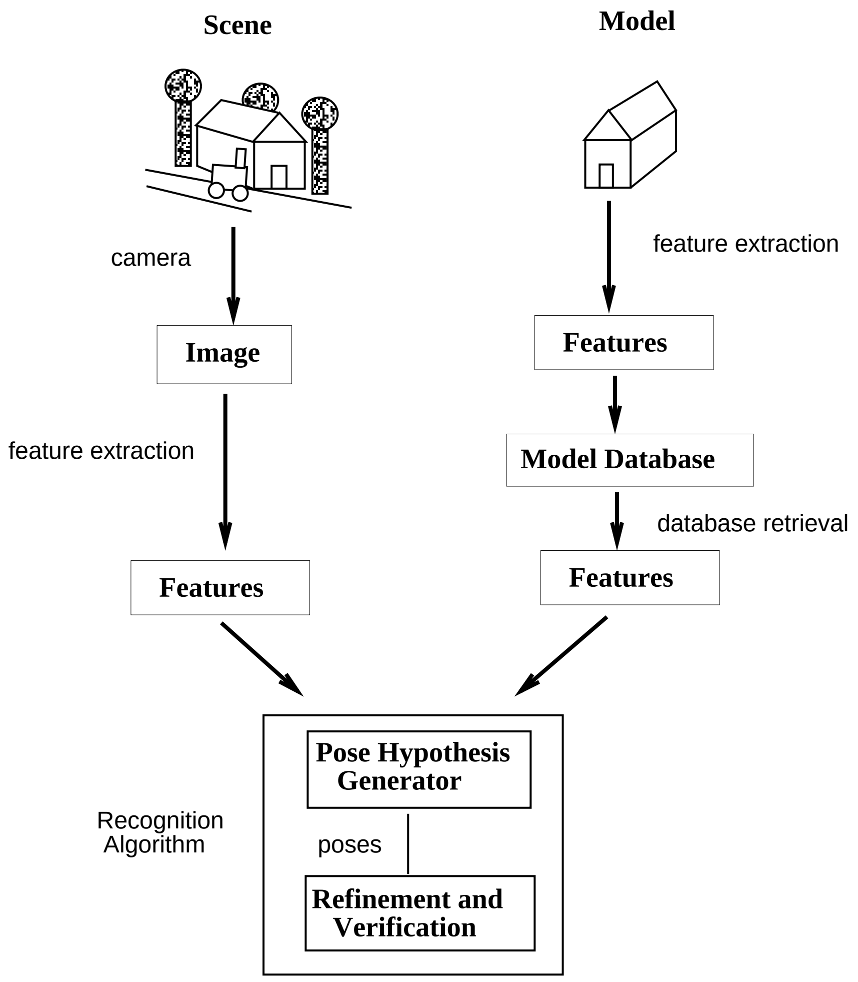


Figure 1-1: Stages in model based recognition.

termine the location of the object in the image, thereby solving for the object's pose (position of the object in the environment).

A typical recognition algorithm contains a stage which searches through pose hypotheses based on small sets of feature correspondences between the model and the image. For every one, the model is projected into the image under this pose assumption. Evidence from the image is collected in favor of this pose hypothesis, resulting in an overall goodness score. If the score passes some threshold θ, then it is accepted. In some algorithms this pose, which is based on a small initial correspondence, may be passed onto a refinement and verification stage. For the pose hypothesis generator, we use the term "correct hypothesis" to denote a pose based on a correct correspondence between model and image features.

If we knew in advance that testing a correct hypothesis for a particular model would result in an overall score of S, then recognizing the model in the image would be particularly simple — we would know that we had found a correct hypothesis when we found one that had a score of S. However, in real images there may be clutter, occlusion, and sensor noise, each of which will affect the scores of correct and incorrect hypotheses. Clutter is the term for features not arising from the model; such features may be aligned in such a way as to contribute to a score for an incorrect pose hypothesis. Occlusion is the absence in the image of features belonging to the model. This serves to possibly lower the score of a correct hypothesis. Lastly, sensor noise is the displacement of observed image features from their true location.

If we knew that part of the model was occluded in the scene and yet we keep the threshold for acceptance at S, we risk the possibility of the algorithm's not identifying a correct hypothesis, which may not score that high. Therefore, we may choose to lower the threshold for acceptance to something slightly less than S. However, the lower we set the threshold, the higher the possibility that an incorrect hypothesis will pass it. The goal is to use a threshold which maximizes the probability that the algorithm will identify a correct hypothesis (called a *true detection*) while minimizing the probability that it accepts an incorrect one (called a *false alarm*).

In this thesis, we determine the implications of using any particular threshold on the probability of true detection and false alarm for a particular recognition algorithm. The method applies to pose hypotheses based on minimal correspondences between model and image points (*i.e.*, size 3 correspondences). We explicitly model the kinds of noise that occurs in real images, and analytically derive probability density functions on the scores of correct and correct hypotheses. These distributions are then used to construct receiver operating characteristic curves (a standard tool borrowed from binary hypothesis testing theory) which indicate all possible triples of (*threshold, probability of false positive, probability of true positive*) pairs for an appropriately specified statistical ensemble. We have demonstrated that the method works well in the domain of both simulated and actual images.

1.2 Object Recognition as Information Recovery

To approach the problem in another way, we can think of the object recognition problem as a process of recovering a set of original parameters about a source. In this abstraction, there is some sort of information exchange between the source and the observer, the information might be corrupted in some fashion, the observer receives some subset of the information with added noise, and finally, processes the observed information in one or several stages to settle upon a hypothesis about the parameters of interest.

For example, in the case of message transmission, the parameters of interest are the message itself, the noise is introduced by the channel, and the observer tries to recover the original transmitted message. In sonar based distance measurement, the parameter of interest is the free distance along a particular direction from a source, the information is the reflected sonar beam, and the perceived information is the time delay between sending and receiving the beam. The observer then processes this

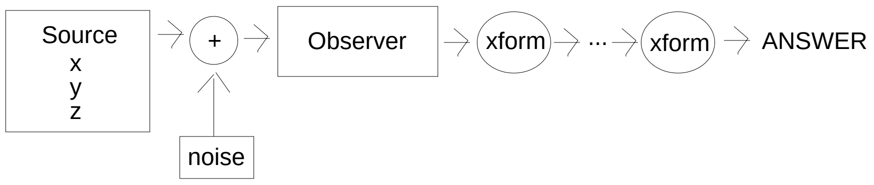


Figure 1-2: Recovering information from a source over a noisy channel.

information to derive a hypothesis about the free space along the particular direction.

In model based vision, the parameters of interest are the presence or absence of a model in a scene, and its pose in three dimensional space. The information is the light which is reflected from a source by the objects in the scene, and the perceived information is the light which enters the lens of a camera. The information goes through several processing stages to get transformed into a two dimensional array of brightness values representing an image, and then through several more steps to come to a hypothesis about the presence and pose of any particular model.

In this thesis we cast the problem as a binary hypothesis testing problem. Let the hypothesis H be "model is at pose P". We are trying to reliably distinguish between H and $\overline{H}$. It is generally not always possible to do this, especially as the noise goes up, but we can bound the probability of error as a function of the statistics of the problem, and can determine when the noise is too high to distinguish between the two hypotheses.

1.3 Overview of the Thesis

All of the definitions, terminology, conventions and formulas that we will use in the thesis are given in Appendix A.

Chapter 2 explains the model based recognition problem in more detail, and gives a very general overview of work relevant to this thesis. We will define the terms and concepts to which we will be referring in the rest of the work.

In Chapter 3 we present the detailed error analysis of the problem.

In Chapter 4 we present the ROC (receiver operating characteristic) curve, borrowed from hypothesis testing theory and recast in terms of the framework of model based recognition. The ROC curve compactly encompasses all the relevant information to predict *(threshold, probability of false positive, probability of false negative)* triples for an appropriately specified statistical ensemble. We also confirm the accuracy of the ROC curves performance predictions with actual experiments consisting of simulated images and models.

Chapter 5 explores the effect of varying some of the assumptions that were used in

Chapter 3. This chapter can be skipped without loss of continuity.

In the first part of Chapter 6 we measure the sensor noise associated with different feature types and imaging conditions. In the second part, we demonstrate the application of ROC curves to the problem of automatic threshold determination for real models and images.

In Chapter 7 we discuss implications of our work for geometric hashing, a recognition technique closely related to the one analysed in the thesis.

Finally, we conclude in Chapter 8 with potential applications and extensions.

Chapter 2

Problem Presentation and Background

We begin by setting the context for our problem. First we will define the terms to which we will be referring in the rest of the thesis. We will then talk about different techniques for solving the recognition problem, and finally we will discuss how these techniques are affected by incorporating an explicit error model.

2.1 Images, Models and Features

An image is simply a two dimensional array of brightness values, formed by light reflecting from objects in the scene and reaching the camera, whose position we assume is fixed (we will not talk about the details of the imaging process). An object in the scene has 6 degrees of freedom (3 translational and 3 rotational) with respect to the fixed camera position. This six dimensional space is commonly referred to as *transformation space*. The most brute force approach to finding an object in the scene would be to hypothesize the object at every point in the transformation space, project the object into the image plane, and perform pixel by pixel correlation between the image that would be formed by the hypothesis, and the actual image. This method is needlessly time consuming however, since the data provided by the image immediately eliminates much of the transformation space from consideration.

Using image features prunes down this vast search space to a more manageable size. What is meant by the term "image feature" is: something detectable in the image which could have been produced by a localizable physical aspect of the model (called *model feature*), regardless of the model's pose. For example, an image feature might be a brightness gradient, which might have been produced by any one of several model features — a sudden change in depth, indicating an edge or a boundary on the object, or a change in color or texture. An image feature can be simple, such as "something at pixel (x,y)", or arbitrarily complex, such as "a 45° straight edge starting at pixel (x,y) of length 5 separating blue pixels from orange pixels".

The utility of features lies in their ability to eliminate entire searches from consideration. For example, if the model description consisted of only corner features bordering a blue region, but there were no such corners detected in the image, this information would obviate the need to search the image for that object. Another way to use features is to form correspondences between image and model features. This constrains the possible poses of the object, since not all points in transformation space will result in the two features being aligned in the image. In fact, depending on the complexity of the feature, sometimes they cannot be aligned at all — for instance, there is no point in transformation space that will align a 45° corner with a curved edge. The more complex the feature, the fewer correspondences are required to constrain the pose completely. For instance, if the features consist of a $2D$ location plus orientation, only two correspondences are required to solve for the pose of the object. If the features are $2D$ points without orientation, then a correspondence between 3 image and model features (referred to as a *size 3 correspondence*) constrains the pose completely.

It would seem intuitively that the richer the feature, the more discriminative power it imparts, since one not need check correspondences that contain incompatible feature pairings. In fact, there is an entire body of work devoted to using feature saliency [SU88, Swa90] to efficiently perform object recognition. It is true that one can use more complex features to prune the search space more drastically, but the more complex the feature, the more likely there is to be an error in the feature pairing process due to error and noise in the imaging and feature extraction processes. For simplicity, in this work we consider only point features, meaning that image and model features are completely characterized by their $2D$ and $3D$ locations, respectively.

2.2 Categorizing Error

We use the term "error" to describe any effect which causes an image of a model in a known pose to deviate from what we expect. The kinds of errors which occur in the recognition process can be grouped into three categories:

- Occlusion — in real scenes, some of the features we expect to find may be blocked by other objects in the scene. There are several models for occlusion: the simplest is to model it as an independent process, *i.e.*, we can say that we expect some percentage of features to be blocked, and consider every feature to have the same probability of being occluded independent of any other feature. Or, we can use a view based method which takes into account which features are self-occluded due to pose. More recently, Breuel [Bre93] has presented a new model that uses locality of features to determine the likelihood of occlusion; that is, if one feature is occluded under a specific pose hypothesis, an adjacent feature is more likely to be occluded.

- Clutter or Noise — these are extraneous features present in the image not arising from the object of interest, or arising from unmodelled processes (for example, highlights). Generally these are modelled as points that are independently and uniformly distributed over the image. These will be referred to as clutter or sometimes, "random image points".

- Sensor Measurement Error — image features formed by objects in the scene may be displaced from their true locations by many causes, among them: lens distortion, illumination variation, quantization error, or algorithmic processing (for instance, a brightness gradient may be slightly moved due to the size of the smoothing mask used in edge detection, or the location of point feature may be shifted due to artifacts of the feature extraction process). This may be referred to as simply "error".

The interest in error models for vision is a fairly recent phenomenon which has been motivated by the fact that for any recognition algorithm, these errors almost always lead to finding an instance of the object where it doesn't appear (called a *false positive*), or missing an actual appearance of an object (a *false negative*).

We will present an overview of recognition algorithms, first assuming that none of these effects are present, and subsequently we will discuss the implications of incorporating explicit models for these processes.

2.3 Search Methods

Much of the work done in model based recognition uses pairings between model and image features, and can be loosely grouped into two categories: correspondence space search and transformation space search. I will treat another method, *indexing*, as a separate category, though it could be argued that it falls within the realm of transformation space search. The error analysis presented in this work applies to those approaches falling in a particular formulation of the transformation space search category. In this section we discuss the general methods, assuming no explicit error modeling.

2.3.1 Correspondence Space

In this approach, the problem is formulated as finding the largest mutually consistent subset of all possible pairings between model and image features, a set whose size is on the order of m^n (in which m is the number of model features, and n is the number of image features). Finding this subset has been formalized as a consistent graph labelling problem [Bha84], or, by connecting pairs of mutually consistent correspondences with edges, as a maximal clique problem [BC82], and as a tree search problem in [GLP84, GLP87]. The running time of all of these methods is at worst exponential, however, at least in the latter approach Grimson has shown that with pruning, fruitless branches of the tree can be abandoned early on, so that this particular method's expected running time is polynomial [Gri90].

2.3.2 Transformation Space

In the transformation space approach, all size G correspondences are tested, where G is the size of the smallest correspondence required to uniquely solve for the transformation needed to bring the G image and model features into correspondence. The transformation thus found is then used to project the rest of the model into the image to search for other corresponding features. The size of the search space is polynomial, $O(n^G m^G)$ to be precise. This overall method has come to be associated with Huttenlocher and Ullman ([HU87]), who dubbed it "alignment", though other previous work used transformation space search (for example, the Hough transform method [Bal81] as well as [Bai84, FB80, TM87], and others). One of the contributions of Huttenlocher's work was to show that a feature pairing of size 3 was necessary and sufficient to solve uniquely for the model pose, and how to do it. Another characteristic of the alignment method as presented in [Hut88] was to use a small number of simple features to form an initial rough pose hypothesis, and to iteratively add features to stabilize and refine the pose. Finally, for the pose to be accepted it must pass a final test in which more complex model and image features must correspond reasonably well, for example, some percentage of the model contour must line up with edges in the image under this pose hypothesis. This last stage is referred to as "verification". Since it is computationally more expensive than generating pose hypotheses, it is more efficient to only verify pose hypotheses that have a reasonable chance of success.

2.3.3 Indexing Methods

Lastly, we come to indexing methods. Here, instead of checking all poses implied by all size 3 pairings between model and image features, the search space is further reduced by using larger image feature groups than the minimum of 3 and to pair them only with groups in the model that could have formed them. This requires a way to access only such model groups without checking all of them. To do this, the recognition process is split into two stages, a model preprocessing stage in which for each group of size G, some distinguishing property of all possible images of that group is computed and used to store the group into a table, indexed by that property. This preprocessing stage takes time $O(m^G)$ (where m is the number of model points). At recognition time, each size G image group is used to index into the table to find the model groups that could have formed it, for a running time of $O(n^G)$ (where n is the number of image points).

At one extreme, we could use an index space of dimension $2G$ (assuming the features are two dimensional) and simply store the model at all positions $(x_1, y_1, ...x_G, y_G)$ for every pose of the model. However, this saves us nothing, since the space requirements for the lookup table would be enormous and the preprocessing stage at least as time consuming as a straight transformation space approach. The trick is to find the lowest dimensional space which will compactly represent all views of a model without sacrificing discriminating power.

Lamdan, Schwartz, Wolfson and Hummel [LSW87, HW88] demonstrate a method, called geometric hashing, to do this in the special case of planar models. Their algorithm takes advantage two things — first, for a group of 3 non-collinear points in the plane, the affine coordinates of any fourth point with respect to the first three as bases is invariant to an affine transformation of the entire model plane. That is, any fourth point can be written in terms of the first three:

$$\mathbf{m}_3 = \mathbf{m}_0 + \alpha(\mathbf{m}_1 - \mathbf{m}_0) + \beta(\mathbf{m}_2 - \mathbf{m}_0).$$

We can think of (α, β) as the affine coordinates of $\mathbf{m}_3$ in the coordinate system established by mapping $\mathbf{m}_0$, $\mathbf{m}_1$, $\mathbf{m}_2$ to $(0,0)$, $(1,0)$, $(0,1)$. These affine coordinates are invariant to a linear transformation T of the model plane.

Second, there is a one-to-one relationship between an image of a planar model in a $3D$ pose and an affine transformation of the model plane. We assume that the pose has 3 rotational and 2 translational degrees of freedom, and we use orthographic projection with scale as the imaging model. Then the $3D$ pose and subsequent projection collapses down to two dimensions:

$$s \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 0 \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ 0 \end{bmatrix} = \begin{bmatrix} sr_{1,1}x_i + sr_{1,2}y_i + t_x \\ sr_{2,1}x_i + sr_{2,2}y_i + t_y \\ 0 \end{bmatrix}$$

where s is the scale factor, and the matrices are the orthographic projection matrix and rotation matrix, respectively. Conversely, a three point correspondence between model and image features uniquely determines both an affine transformation of the model plane, and also a unique scale and pose for an object (up to a reflection over a plane parallel to the image plane; see [Hut88]).

Therefore, suppose we want to locate an ordered group of four model points in an image (where the model's $3D$ pose is unknown). The use of the affine coordinates of the fourth point with respect to the first three as basis to describe this model group is *pose invariant*, since no matter what pose the model has, if we come across the four image points formed by this model group, finding the coordinates of the fourth image point with respect to the first three yields the same affine coordinates.

Geometric hashing involves doing this for *all* model groups of size 4 at the same time. The algorithm requires the following preprocessing stage: for each model group of size 4, the affine coordinates of the fourth point are used as a pose invariant index into the table to store the first three points. This stage takes $O(m^4)$, where m is the number of model points. At recognition time, each size 3 image group is tested in the following way: for a fixed image basis B, (a) for every image point, the affine coordinates are found with respect to B, then (b) the affine coordinates are used to index into the hash table. All model bases stored at the location indexed by the affine coordinates are candidate matches for the image basis B. A score is incremented for each candidate model basis and the process is repeated for each image point. After all image points have been checked, the model basis that accumulated the highest score and passes some threshold is taken as a correct match for the image basis B.

In theory, the technique takes time $O(n^4 + m^4)$. More recently, Clemens and Jacobs formalized the indexing problem in [CJ91] and showed that 4 dimensions is the minimum required to represent $3D$ models in arbitrary poses. All views of a group of size 4 form a $2D$ manifold in this space, implying that unlike in the planar model domain, there exists no pose invariant property for $3D$ models in arbitrary poses. Other work involving geometric hashing can be found in [CHS90, RH91, Ols93, Tsa93].

2.4 The Effect of Error on Recognition Methods

All recognition algorithms test pose hypotheses by checking for a good match between the the image that would be formed by projecting the model using the tested pose hypothesis, and the actual image. We will discuss exactly what we mean by a "good match" shortly. The three kinds of errors cause qualitatively different problems for recognition algorithms. The effect of occlusion brings down the amount of evidence in favor of correct hypotheses, risking false negatives. The presence of clutter introduces the possibility that a clutter feature will arise randomly in a position such that it is counted as evidence in favor of an incorrect pose hypothesis, risking false positives. Sensor error has the effect of displacing points from their expected locations, such that a simple test of checking for a feature at a point location in the image turns into a search over a small disk, again risking the possibility of false positives.

It would appear that simply in terms of running time, the search techniques from (correspondence space search $\rightarrow$ transformation space search $\rightarrow$ indexing) go in order of worst to best. However, this ranking becomes less clear once the techniques are modified to take error into account. The differences between the approaches then become somewhat artificial in their implementations, since extra steps must often be added which blur their conceptual distinctions.

Correspondence space search is the most insensitive to error, since given the correct model-feature pairings, the globally best pose can be found by minimizing the sum of the model to image feature displacements.

For transformation space approaches, dealing with error turns the problem into a potentially exponential one. The reason is that the transformation space approach checks only those points in the space that are indicated by size 3 correspondences between model and image features. Though there are many correct image to feature correspondences, there is only one globally correct pose. The poses implied by all correct correspondences will be clustered near the globally correct pose in transformation space, but it is likely that none of them will actually land on it. Therefore, finding the globally best pose will require iteratively adding model-feature pairings to the initial correspondence and minimizing the total error. However, for each additional pairing, the model point in the pair can match to any image points which appears in a finite sized disk in the image. Assuming uniform clutter, some fraction

k of all the image points will appear in such a disk. If all of them have to be checked as candidate matches, this brings the search to size $O(m^{kn})$.

To conclude the discussion of the effect of noise on different techniques, we note that in general, the more efficient an algorithm, the more unstable it is in the presence of noise. This observation is not really surprising since the speed/reliability trade-off is as natural and ubiquitous in all computer science as the speed/space trade-off. In the remaining discusion and throughout the thesis, we will be dealing solely with transformation space search, and the analysis that we present is applicable equally well to both alignment and geometric hashing.

2.5 Error Models in Vision

The work incorporating explicit error models for vision has used either a uniform bounded error model, or a $2D$ Gaussian error model. A uniform bounded error model is one in which the difference between the sensed and actual location of a projected model point can be modeled as a vector drawn from a bounded disk with uniform, or flat, distribution. A Gaussian error model is one in which the sensed error vector is modeled with a two dimensional Gaussian distribution. Clutter and occlusion, when modeled, are done so as uniformly distributed and independent. Though there has not been a great deal of this type of work, there are some notable examples.

2.5.1 Uniform Bounded Error Models

Recently, Cass showed that finding the best pose in transformation space, assuming a uniform bounded error associated with each feature, can be reduced to the problem of finding the maximal intersection of spiral cylinders in transformation space. Stated this way, the optimal pose can be found in polynomial time ($O(m^6n^6)$) by sampling only the points at which pairs of these spiral cylinders intersect [Cas90]. Baird [Bai84] showed how to solve a similar problem for polygonal error bounds in polynomial time by formulating it in terms of finding the solution to a system of linear equations.

Grimson, Huttenlocher and Jacobs [GHJ91] did a detailed comparative error analysis of the both alignment and geometric hashing method of [LSW87, HW88]. They used a uniform bounded error model in the analysis and concentrated on determining the probability of false positives for each technique. Also, Jacobs demonstrates an indexing system for $3D$ models in [Jac92] which explicitly incorporates uniform bounded error.

2.5.2 Gaussian Error Models

The previous work all used a uniform bounded error model to analyze the effect of error on the recognition problem. This model is in some ways simpler to analyze, but in general it is too conservative a model in that it overestimates the effect of error. A Gaussian error model will often give analytically better results and so it is often assumed even when the actual distribution of error has not been extensively tested. It can be argued, however, that the underlying causes of error will contribute to a more Gaussian distribution of features, simply by citing the Central Limit Theorem. In [Wel92], Wells presented experimental evidence that indicates that for a TV sensor and a particular feature class, a Gaussian error model is in fact a more accurate noise model than the uniform. Even when the Gaussian model is assumed, there is often not a good idea of the standard deviation, and generally an arbitrary standard deviation is picked empirically.

Wells also solved the problem of finding the globally best pose and feature correspondence with Gaussian error by constructing an objective function over pose and correspondence space whose argmin was the best pose hypothesis in a Bayesian sense. To find this point in the space he used an expectation-maximization algorithm which converged quite quickly, in 10-40 iterations, though the technique was not guaranteed to converge to the likelihood maximum.

Rigoutsos and Hummel [RH91] and Costa, Haralick and Shapiro [CHS90] independently formulated a method to do geometric hashing with Gaussian error, and demonstrated results more encouraging that those predicted in Grimson, Huttenlocher and Jacobs' analysis of the uniform bounded model. Tsai also demonstrates an error analysis for geometric hashing using line invariants in [Tsa93].

Bolles, Quam, Fischler, and Wolf demonstrate an error analysis in the domain of recognizing terrain models from aerial photographs ([BQFW78]). In their work, a Gaussian error model was used to model the uncertainty in the camera parameters and camera to scene geometry, and it was shown that the under a particular hypothesis (which in this domain is the camera to scene geometry) the regions consistent with the projected model point locations (features in the terrain model) are ellipses in the image.

2.5.3 Bayesian Methods

The Gaussian error model work has used a Bayesian approach to pose estimation, *i.e.,*, it assumes a prior probability distribution on the poses and uses the rule

$$P(\text{pose} \mid \text{data}) = \frac{P(\text{pose})P(\text{data}|\text{pose})}{P(\text{data})}$$

to infer the most likely pose given the data. The noise model is used to determine the conditional probability of the data given the pose. In Bayesian techniques, the denominator in this expression is assumed to be uniform over all possible poses, and so can be disregarded ([Wel92, RH91, CHS90, Tsa93]). This assumes that one of the poses actually is correct, that is, that the object actually appears in the image. The pose which maximizes this expression is the globally optimal pose. However, if we do not know whether the model appears in the image at all, we cannot use the above criterion.

2.6 The Need for Decision Procedures

In general, it is possible to find the globally best pose with respect to some criterion, but if we have no information as to whether *any* of the possible poses are correct, that is, if we have no information as to the probability that the model appears in the image, then we must determine at what point even the most likely pose is compelling enough to accept it.

In this thesis we address this problem with respect to poses based on size 3 corresponces between image and model features. We will use the term "correct" hypotheses to denote correct size 3 correspondences. Such correct correspondences indicate points in transformation space that are close to the correct pose for the model in the image. Since transformation space search samples only those points in transformation space that are implied by size 3 correspondences, what we are doing is trying to determine when we have found a point in the space close enough to the correct pose to accept it or to pass it on to a more costly verification stage.

Suppose we were working with a model of size m in a domain with no occlusion, clutter, or error. In this case, a correct hypothesis would always have all corroborating evidence present. Therefore, to test if a hypothesis is correct or not, one would project the model into the image subject to the pose hypothesis implied by the correspondence, and test if there were m image points present where expected. We call this test a *decision procedure* and m the threshold. However, suppose we admit the possibility of occlusion and clutter, modeled as stated. Now it is not clear how many points we need to indicate a correct hypothesis, since the number of points in the image that will arise from the model is not constant. In particular, if there is the probability c for any given point to be occluded, then the number of points we will see for a correct hypothesis will be a random variable with binomial distribution. Deciding if a hypothesis is correct is a question of determining if the amount of evidence exceeds a reasonable threshold. So even without sensor error, we must have a decision procedure and with it, an associated probability of making a mistake.

When we also consider sensor error, the uncertainty in the sensed location of the 3 image points used in the correspondence to solve for the pose hypothesis magnifies the positional uncertainty of the remaining model points (Figure 2-1). Therefore since a model point could fall anywhere in this region, we have to count any feature which appears there as evidence in favor of the pose hypothesis. As the regions spread out spatially, there is a higher probability that a clutter feature will appear in such a region, even though it does not arise from the model. So now, instead of never finding any evidence corroborating an incorrect pose hypothesis (assuming only asymmetric models), the amount of evidence we find will also be a random variable with distribution dependent on the error model.

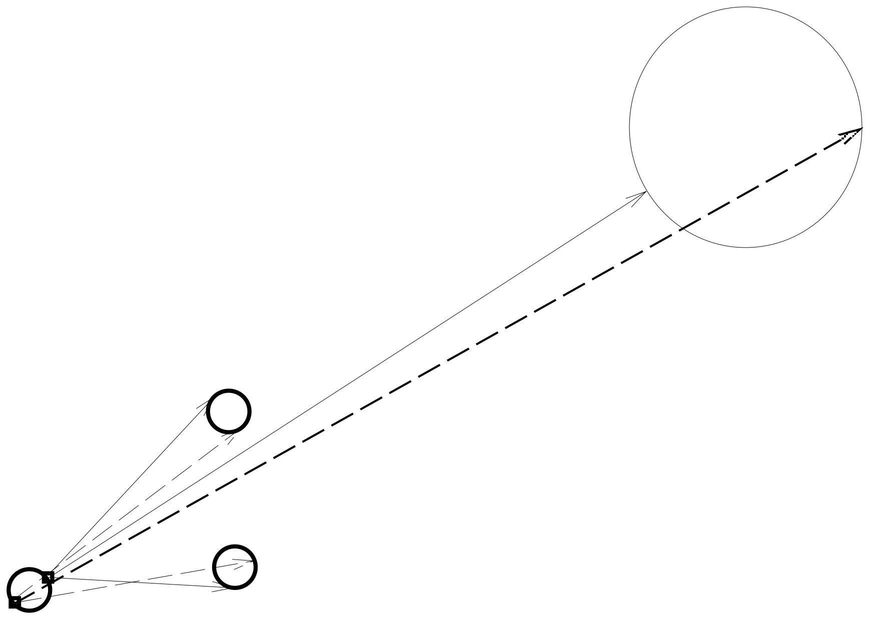


Figure 2-1: Possible positions of a model point due to positional uncertainty in the three points used in the correspondence to form the pose hypothesis.

It is important to understand the implications of using any particular threshold as a decision procedure, since when the distributions of the two random variables overlap, using a threshold will necessarily imply missing some good pose hypotheses and accepting some bad ones. Most working vision systems operate under conditions in which the random variables describing good and bad hypotheses are so widely separated that it is easy to tell the difference between them. Few try to determine how their system's performance degrades as the distributions approach each other until they are so close that it is not possible to distinguish between them.

It is this area that is addressed in this thesis. Our approach focuses not on the pose estimation problem, but rather on the decision problem, that is, given a particular pose hypothesis, what is the probability of making a mistake by either accepting or rejecting it? This question has seldom been dealt with, though one notable exception is the "Random Sample Consensus" (RANSAC) paradigm by Fischler and Bolles ([FB80]), in which measurement error, clutter and occlusion were modeled similarly as in our work, and the question of choosing thresholds in order to avoid false positives addressed as well. More recently, error analyses concentrating on the probability of false positives were presented in the domain of Hough transforms by [GH90], and in geometric hashing by [GHJ91], and much of the approach developed in this thesis owes a debt to that work.

Conclusion

We have structured this problem in a way which can be applied to those algorithms which sample transformation space at those points implied by correspondences between 3 model and image features. In the next few chapters we will present the method, and its predictive power for both simulated and real images.

Chapter 3

Presentation of the Method

In this chapter the problem we address is, given a model of an object and an image, how do we evaluate hypotheses about where the model appears in the image?

The basic recognition algorithm that we are assuming is a simple transformation space search equivalent to alignment, in which pose hypotheses are based on initial minimal correspondences between model and image points. The aim of the search is to identify correct correspondences between model and image points. We will refer to correct and incorrect correspondences as "correct hypotheses" and "incorrect hypotheses". Correct hypotheses specify points in transformation space that are close to the correct pose, and can be used as starting points for subsequent refinement and verification stages. The inner loop of the algorithm consists of testing the hypothesis for possible acceptance. The steps are:

(1) For a given 3 model points and 3 image points,

(2) Find the transformation for the model which aligns this triple of model points to the image points,

(3) Project the remaining model points into the image according to this transformation,

(4) Look for possible matching image points for each projected model point, and tally up a score depending on the amount of evidence found.

(5) If the score exceeds some threshold θ, then we say the hypothesis is correct.

Correspondences can be tested exhaustively, or the outer algorithm can use more global information (such as grouping) to guide the search towards correspondences which are more likely to be correct. The actual manner through which the correspondences are searched is not relevant to the functioning of the inner loop.

In the presentation of the algorithm, steps (4) and (5) are deliberately vague. In particular, how do we tally up the score, and how do we set the threshold? The answer to these two questions are linked to each other, and in order to answer them we need to select:

- A weighting scheme — that is, when we project the model back into the image, how we should weight any image points which fall near, but not exactly at, the expected location of the other model points. The weighting scheme should be determined by the model for sensor error.
- A method of accumulating evidence for a given hypothesis.
- A decision procedure — that is, how to set the threshold θ, which is the score needed to accept a hypotheses as being correct.

The first two choices determine the distributions of scores associated with correct and incorrect hypotheses. Different choices can make the analytic derivation of these distributions easier or harder; Chapter 5 will discuss some of these issues but for now we present a single scheme for which we can do the analysis.

After a brief presentation of the mechanics of the alignment algorithm, we will present the error assumptions we are using for occlusion, clutter, and sensor noise, and how these assumptions affect our scoring algorithm. For the remainder of the chapter we will present a particular scoring algorithm for hypotheses, and we will derive the score distributions associated with correct and incorrect hypotheses as a function of the scoring algorithm. Once we know these distributions, the question of determining the relationship between performance and the threshold used for acceptance will become straightforward.

In our analysis we limit ourselves to the domain of planar objects in $3D$ poses. We assume orthographic projection with scaling as our imaging model, and a Gaussian error model, that is, the appearance in the image of any point arising from the model is displaced by a vector drawn from a $2D$ circular Gaussian distribution. Because much of the error analysis work in this domain has assumed a bounded uniform model for sensor error, we will periodically refer to those results for the purpose of comparison.

3.1 Projection Model

In this problem, our input is an image of a planar object with arbitrary $3D$ pose. Under orthographic projection with scaling, we can represent the image location $[u_i, v_i]^T$ of each model point $[x_i, y_i]^T$ with a simple linear transformation:

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{3.1}$$

where the transformation matrix is a 2×2 non-singular matrix, and $[t_x, t_y]^T$ is the translation vector. To easily see why this is so, note that when the model is planar, the coordinate frame of the model can be chosen so that the third coordinate is always 0. In this case the $3D$ transformation collapses down to two dimensions:

$$s \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 0 \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ 0 \end{bmatrix} = \begin{bmatrix} sr_{1,1}x_i + sr_{1,2}y_i + t_x \\ sr_{2,1}x_i + sr_{2,2}y_i + t_y \\ 0 \end{bmatrix}$$

Here, s is the scale factor, and the matrices are the orthographic projection matrix and rotation matrix, respectively. Conversely, a three point correspondence between model and image features uniquely determines both an affine transformation of the model plane, and also a unique scale and pose for an object (up to a reflection over a plane parallel to the image plane; see [Hut88]).

3.2 Image, Model, and Affine Reference Frames

Conceptually, there are three different coordinate frames we utilize during the analysis. *Model space* is the global reference frame used for the model representation, and *image space* is the global reference frame of the image. The transformation from model space to image space is accomplished by the linear projection model discussed above.

A third coordinate frame, called *affine space*, is used for each correspondence tested. This coordinate frame is established by the three model points used in the initial correspondence (which must not be collinear, or they would not span a plane). The ordered triple of model and image points used in the correspondence is referred to as the *model basis* and *image basis*, respectively. Each model point can be uniquely expressed as a linear function of the model basis:

$$\mathbf{m}_i \quad = \quad \mathbf{m}_0 + \alpha_i(\mathbf{m}_1 - \mathbf{m}_0) + \beta_i(\mathbf{m}_2 - \mathbf{m}_0) \tag{3.2}$$

We can think of the vectors $(\mathbf{m}_1 - \mathbf{m}_0)$ and $(\mathbf{m}_2 - \mathbf{m}_0)$ as the unit basis vectors $(1, 0)$ and $(0, 1)$ establishing the affine coordinate frame, in which (α_i, β_i) are the affine coordinates of $\mathbf{m}_i$.

We convert from model space coordinates to affine space coordinates as follows: given model points $\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2$ (in model space coordinates), the coordinates of a fourth point $\mathbf{m}_i$ with respect to this basis are given by the expression

$$\begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} = \begin{bmatrix} \|\mathbf{m}'_i\| \frac{\sin(\phi - \psi)}{\|\mathbf{m}'_1\| \sin\phi} \\ \|\mathbf{m}'_i\| \frac{\sin\psi}{\|\mathbf{m}'_2\| \sin\phi} \end{bmatrix}$$

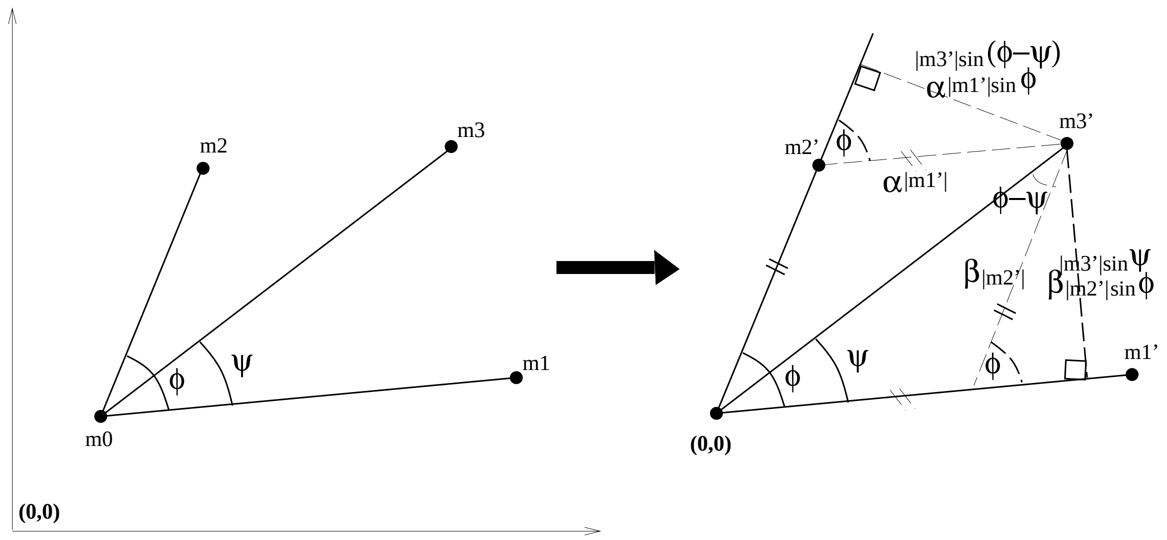


Figure 3-1: Calculating the affine coordinates of a fourth model point with respect to three model points as basis.

in which

$$\mathbf{m}'_1 = \mathbf{m}_1 - \mathbf{m}_0 \qquad \mathbf{m}'_2 = \mathbf{m}_2 - \mathbf{m}_0 \qquad \mathbf{m}'_i = \mathbf{m}_i - \mathbf{m}_0$$

$$\phi = \angle \mathbf{m}'_1\mathbf{m}'_2 \qquad \psi = \angle \mathbf{m}'_1\mathbf{m}'_i$$

and $\angle$ denotes the angle between the two vectors of the argument.

If we perform an affine transformation $\mathbf{T}$ plus a translation $\mathbf{t}$ such as in Equation 3.1 to both sides of Equation 3.2, we demonstrate that the affine space coordinates (α_i, β_i) of a model point $\mathbf{m}_i$ remain unchanged with respect to the transformed coordinates of the basis:

$$\begin{aligned}
\mathbf{T}[\mathbf{m}_i] + \mathbf{t} &= \mathbf{T}[\mathbf{m}_0 + \alpha_i(\mathbf{m}_1 - \mathbf{m}_0) + \beta_i(\mathbf{m}_2 - \mathbf{m}_0)] + \mathbf{t} \\
&= \mathbf{T}\mathbf{m}_0 + \mathbf{t} + \mathbf{T}[\alpha_i(\mathbf{m}_1 - \mathbf{m}_0)] + \mathbf{T}[\beta_i(\mathbf{m}_2 - \mathbf{m}_0)] \\
&= \mathbf{T}\mathbf{m}_0 + \mathbf{t} + \alpha_i(\mathbf{T}\mathbf{m}_1 - \mathbf{T}\mathbf{m}_0) + \beta_i(\mathbf{T}\mathbf{m}_2 - \mathbf{T}\mathbf{m}_0) \\
&= [\mathbf{T}\mathbf{m}_0 + \mathbf{t}] + \alpha_i([\mathbf{T}\mathbf{m}_1 + \mathbf{t}] - [\mathbf{T}\mathbf{m}_0 + \mathbf{t}]) + \beta_i([\mathbf{T}\mathbf{m}_2 + \mathbf{t}] - [\mathbf{T}\mathbf{m}_0 + \mathbf{t}])
\end{aligned}$$

This invariance of affine space coordinates under linear transformations (which we will call "affine invariance") gives us several advantages. First, we can find the projected image location of projected model points without having to solve directly for the transformation, since the image locations of all the model points can be expressed by such a linear operation. Therefore, the image location of a projected model point $\mathbf{m}_i$ with affine coordinates (α_i, β_i) with respect to a given basis, once a correspondence between model and image points has been established, is given by the expression

$$\mathbf{s}_i = \mathbf{s}_0 + \alpha_i(\mathbf{s}_1 - \mathbf{s}_0) + \beta_i(\mathbf{s}_2 - \mathbf{s}_0) \tag{3.3}$$

where $\mathbf{s}_i$ denotes the ith images point. Second, since there is a one-to-one correspondence between affine transformations and poses, the affine invariance of this representation implies that it is pose invariant as well. This is the key to a particular form of indexing, called geometric hashing. In particular, for the affine space established by every model basis, the affine coordinates of every other model point is used as pose invariant indices into a table into which the model basis is stored.

Because the affine representation is pose independent, it is the smallest model representation for indexing (see discussion of relative and absolute axes, [CJ91]); any other smallest representation must necessarily use coordinates which are functions of the coordinates on the axes formed by the model basis. Because of this, and the recent interest in affine coordinates in indexing and invariance, it is this representation that we discuss and analyze in the rest of the work. Using this representation, we will be able to apply the analysis to both alignment and geometric hashing.

3.3 Error Assumptions

We use the term "error" to describe any effect which causes an image of a model in a known pose to deviate from what we expect, using our projection model to form the image from the model. There are three kinds of error we will be assuming for the analysis — occlusion, clutter, and sensor error.

Occlusion occurs as a result of some part of the object in the scene being blocked, thereby preventing the model feature from appearing in the image. The way we model this process is to assume that all features on the model have the same probability c of being occluded, and that the occludedness of any particular feature does not affect any other. Though this independence assumption is probably not accurate, it is often assumed for the sake of simplicity.

Any image point which does not arise from the model is referred to as clutter. We assume that these points will be independently and uniformly distributed over the image.

Lastly, we refer to the difference between an image feature's observed to actual location as "sensor error". This displacement may arise due to artifacts of the imaging or feature extraction process. We assume the same standard deviation of the sensed error for all points from the same image, denoted by σ_0. The actual value of σ_0 will depend on things such as lighting conditions, camera, and feature type used, and may change from image to image. In the next section we will derive the effect this sensor error has on the possible projected locations of model points in the image.

3.4 Deriving the Projected Error Distribution

In this section we give an expression for the possible locations of the projected model points as a function of error in the observed image locations of the basis points. Any point which appears at one of these locations may have arisen from this hypothesis, and may be counted in favor of its being correct. We use both a uniform bounded error model and a Gaussian error model for the purpose of comparison.

3.4.1 Uniform Bounded Error

We are assuming that the sensed location of a point, $\mathbf{s}_i$, is displaced from its actual location by a vector drawn from a uniform bounded distribution. Let us use $\hat{\mathbf{s}}_i$ to denote the true location of the point, and $\mathbf{e}_i$ to denote the error vector. Therefore, for every image point,

$$\mathbf{s}_i = \hat{\mathbf{s}}_i + \mathbf{e}_i.$$

Let us assume that $\{\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2\} : \{\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2\}$ is a correct image to model correspondence, and let (α_i, β_i) be the affine space coordinates of a fourth model point $\mathbf{m}_i$. Then the true image location of $\mathbf{m}_i$ is a function of true image locations of $\{\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2\}$:

$$\hat{\mathbf{s}}_i = (1 - \alpha_i - \beta_i)\hat{\mathbf{s}}_0 + \alpha_i \hat{\mathbf{s}}_1 + \beta_i \hat{\mathbf{s}}_2$$

However, the computed location of $\mathbf{m}_i$ is a function of the locations of the image basis points. We will denote the computed location as $\tilde{\mathbf{s}}_i$, and

$$\begin{aligned} \tilde{\mathbf{s}}_i &= \mathbf{s}_0 + \alpha_i(\mathbf{s}_1 - \mathbf{s}_0) + \beta_i(\mathbf{s}_2 - \mathbf{s}_0) \\ &= (1 - \alpha_i - \beta_i)\mathbf{s}_0 + \alpha_i \mathbf{s}_1 + \beta_i \mathbf{s}_2 \end{aligned}$$

The expression for the displacement vector for the projected model point is given by the difference between its computed and true location:

$$\begin{aligned} \tilde{\mathbf{s}}_i - \hat{\mathbf{s}}_i &= (1 - \alpha_i - \beta_i)\mathbf{s}_0 + \alpha_i \mathbf{s}_1 + \beta_i \mathbf{s}_2 - (1 - \alpha_i - \beta_i)\hat{\mathbf{s}}_0 + \alpha_i \hat{\mathbf{s}}_1 + \beta_i \hat{\mathbf{s}}_2 \\ &= (1 - \alpha_i - \beta_i)[\hat{\mathbf{s}}_0 + \mathbf{e}_0] + \alpha_i[\hat{\mathbf{s}}_1 + \mathbf{e}_1] + \beta_i[\hat{\mathbf{s}}_2 + \mathbf{e}_2] \\ &\quad -(1 - \alpha_i - \beta_i)\hat{\mathbf{s}}_0 + \alpha_i \hat{\mathbf{s}}_1 + \beta_i \hat{\mathbf{s}}_2 \\ &= (1 - \alpha_i - \beta_i)\mathbf{e}_0 + \alpha_i \mathbf{e}_1 + \beta_i \mathbf{e}_2 \end{aligned}$$

When the error vectors $\mathbf{e}_i$ are drawn from a uniform circular distribution with radius ϵ_0, the vector given by this expression was shown in [GHJ91] to be distributed over a disk with radius

$$\epsilon_0(| 1 - \alpha_i - \beta_i | + | \alpha_i | + | \beta_i | + 1). \tag{3.4}$$

3.4.2 Gaussian Error

For the $2D$ Gaussian error model, we will use the terminology $X \sim N(m, \sigma^2)$ to denote that the random variable X is normally distributed with mean m and variance σ^2. Also, E$[X]$ denotes the expected value of the random variable X. We assume a fixed standard deviation σ_0 for the error distribution, and proceed as follows[1].

Let $\hat{\mathbf{s}}_i$ = true image location of model point $\mathbf{m}_i$:

$$\hat{\mathbf{s}}_i = (1-\alpha_i-\beta_i)\hat{\mathbf{s}}_0 + \alpha_i\hat{\mathbf{s}}_1 + \beta_i\hat{\mathbf{s}}_2.$$

Let $\mathbf{s}_i$ = observed image location of $\mathbf{m}_i$:

$$\mathbf{s}_i = \hat{\mathbf{s}}_i + \mathbf{e}_i$$

where $\mathbf{e}_i \sim N(0,\sigma_0^2)$. Then $\mathbf{s}_i$ is a random variable, $\mathbf{s}_i \sim N(\hat{\mathbf{s}}_i,\sigma_0^2)$.

Let $\tilde{\mathbf{s}}_i$ = computed image location of $\mathbf{m}_i$:

$$\tilde{\mathbf{s}}_i = (1-\alpha_i-\beta_i)\mathbf{s}_0 + \alpha_i\mathbf{s}_1 + \beta_i\mathbf{s}_2.$$

The projected error distribution in which we are interested is the difference between the computed and observed image location of $\mathbf{m}_i$:

$$\begin{aligned}
\Delta\hat{\mathbf{s}}_i &= \mathbf{s}_i - \tilde{\mathbf{s}}_i \\
\mathrm{E}[\Delta\hat{\mathbf{s}}_i] &= \mathrm{E}[\mathbf{s}_i] - \mathrm{E}[\tilde{\mathbf{s}}_i] \\
&= \mathrm{E}[\mathbf{s}_i] - ((1-\alpha_i-\beta_i)\mathrm{E}[\mathbf{s}_0] + \alpha_i\mathrm{E}[\mathbf{s}_1] + \beta_i\mathrm{E}[\mathbf{s}_2]) \\
&= \hat{\mathbf{s}}_i - ((1-\alpha_i-\beta_i)\hat{\mathbf{s}}_0 + \alpha_i\hat{\mathbf{s}}_1 + \beta_i\hat{\mathbf{s}}_2) \\
&= 0
\end{aligned}$$

For the covariance matrix, we want to find the relation between any two random variables $\Delta\hat{\mathbf{s}}_i$ and $\Delta\hat{\mathbf{s}}_j$:

$$\begin{aligned}
\mathrm{Cov}\,(\Delta\hat{\mathbf{s}}_i,\Delta\hat{\mathbf{s}}_j) &= \mathrm{E}\left[\Delta\hat{\mathbf{s}}_i\Delta\hat{\mathbf{s}}_j^T\right] - \mathrm{E}[\Delta\hat{\mathbf{s}}_i]\,\mathrm{E}\left[\Delta\hat{\mathbf{s}}_j^T\right] \\
&= \mathrm{E}\left[(\mathbf{s}_i-\tilde{\mathbf{s}}_i)(\mathbf{s}_j-\tilde{\mathbf{s}}_j)^T\right] - 0 \\
&= \mathrm{E}[(\hat{\mathbf{s}}_i+\mathbf{e}_i-(1-\alpha_i-\beta_i)[\hat{\mathbf{s}}_0+\mathbf{e}_0]+\alpha_i[\hat{\mathbf{s}}_1+\mathbf{e}_1]+\beta_i[\hat{\mathbf{s}}_2+\mathbf{e}_2])\,* \\
&\quad (\hat{\mathbf{s}}_j+\mathbf{e}_j-(1-\alpha_j-\beta_j)[\hat{\mathbf{s}}_0+\mathbf{e}_0]+\alpha_j[\hat{\mathbf{s}}_1+\mathbf{e}_1]+\beta_j[\hat{\mathbf{s}}_2+\mathbf{e}_2])^T] \\
&= \mathrm{E}[(\mathbf{e}_i-(1-\alpha_i-\beta_i)\mathbf{e}_0+\alpha_i\mathbf{e}_1+\beta_i\mathbf{e}_2)\,* \\
&\quad (\mathbf{e}_j-(1-\alpha_j-\beta_j)\mathbf{e}_0+\alpha_j\mathbf{e}_1+\beta_j\mathbf{e}_2)^T]
\end{aligned}$$

Since all the $\mathbf{e}_i$'s are independent, all terms $\mathbf{e}_i\mathbf{e}_j^T, i \neq j$ disappear when we multiply and average, leaving

$$\mathrm{E}\left[(1-\alpha_i-\beta_i)(1-\alpha_j-\beta_j)\mathbf{e}_0\mathbf{e}_0^T + \alpha_i\alpha_j\mathbf{e}_1\mathbf{e}_1^T + \beta_i\beta_j\mathbf{e}_2\mathbf{e}_2^T + \mathbf{e}_i\mathbf{e}_j^T\right] = \quad (3.5)$$

$$\begin{cases} [(1-\alpha_i-\beta_i)(1-\alpha_j-\beta_j)+\alpha_i\alpha_j+\beta_i\beta_j]\,\sigma_0^2 I & i \neq j \\ [(1-\alpha_i-\beta_i)^2+\alpha_i^2+\beta_i^2+1]\sigma_0^2 I & i = j \end{cases} \quad (3.6)$$

where I is the identity matrix. The difference between the terms for $i=j$ and $i \neq j$ comes from the fact that in the former case, $\mathrm{E}\left[\mathbf{e}_i\mathbf{e}_i^T\right] = \sigma_0^2 I$, while in the latter case

$\mathrm{E}\left[\mathbf{e}_i\mathbf{e}_j^T\right] = 0$. So, the distribution of the error vector for the ith point is a circular Gaussian with variance

$$\sigma_i^2 \equiv \sigma_0^2[(1-\alpha_i-\beta_i)^2 + \alpha_i^2 + \beta_i^2 + 1] \tag{3.7}$$

The fact that the distribution has non-zero covariance (Equation 3.6) indicates that the error vectors for the different projected points are not independent. Since this dependence is difficult to take into consideration, we will assume that they are independent and proceed with the analysis. This assumption will cause us to underestimate the true variance of the score distribution for correct hypotheses, as we will see in a later section.

3.5 Defining the Uniform and Gaussian Weight Disks

In our recognition algorithm, all size 3 correspondences are searched through in order to find a good pose. Each correspondence between model and image points is used to project the rest of the model points into the image, and for each projected model point location, if an image point appears at that location, this is counted towards a total score for this hypothesis. If after checking all the projected model point locations the total score exceeds some threshold, this hypothesis is accepted.

When a correct correspondence is tested in the absence of any error, there will always be an image point at the exact projected location of every model point. When we take sensor error into account, then any image point appearing within the range of the projected error distribution is a match candidate for the projected model point. It is clear that the larger the distribution, the more likely it is that a random image point will appear within its range.

In all previous work involving analyzing a bounded uniform sensor error model, the method of scoring a point which appears inside the range of the projected error distribution has been to accord it a full vote. Though the projected error distribution is in fact not uniform, these analyses have implicitly treated it as though it were, by according the same score to any point which appears inside it.

In order to differentiate the scoring method from the error model, we will define an entity called a *weight disk*, whose height at every point determines the score of an image point which appears at that location. For example, the weight disk implied by the scoring scheme just mentioned is a disk, centered at the projected model point location, with height 1 and radius given by Equation 3.4. This will be called the "uniform weight disk". Though an optimal weight disk for a given error model may exist, it may also be difficult to derive or use. In general, we will speak of comparing weight disks, rather than error models, unless we are comparing the optimal weight disks for the error models involved.

We now define the weight disk which we will use to assign scores to points appearing in locations consistent with the projected error distributions, assuming a Gaussian model for sensor noise. Because the projected Gaussian distribution is unbounded, it could give rise to a point appearing anywhere in the image with non-zero probability. In practice though, we will ignore all points appearing outside a disk of radius 2σ from the center. The reason for this is to reduce run time and will become clear when we discuss geometric hashing. Because of this limitation, we will assign a value of 0 to points from the part of the distribution extending from 2σ to ∞:

$$\begin{aligned}
\int_{2\sigma}^{\infty}\int_{0}^{2\pi}\frac{1}{2\pi\sigma^2}e^{-\frac{r^2}{2\sigma^2}}r\,dr\,d\theta &= \int_{2\sigma}^{\infty}\frac{r}{\sigma^2}e^{\frac{-r^2}{2\sigma^2}}dr \\
&= -e^{\frac{-r^2}{2\sigma^2}}\Big|_{2\sigma^2}^{\infty} \\
&= e^{-2}
\end{aligned}$$

That is, we will miss an image feature arising from a model point 13.5% of the time.

Next, for a point falling within the range of the truncated distribution, we will assign weights according to their proximity to the disk center. The weight is chosen to be:

$$v = \frac{1}{2\pi\sigma^2}e^{-\frac{d^2}{2\sigma^2}}$$

where d =distance from the point's location to the disk center. This is the actual height of the $2D$ Gaussian distribution at the location where the image point appears. Again, this weighting is not optimal for this error model, and we will discuss different weighting schemes and their implications in a Chapter 5. Therefore, the Gaussian weight disk is a Gaussian distribution, centered at the projected model location, and truncated at 2σ from the center.

Figure 3-2 illustrates the projected Gaussian and uniform weight disks. The figure shows that the Gaussian weight disks are smaller and more dense at the center than the uniform weight disks; this can also be seen by comparing the analytic expression for the radius of the uniform weight disk (Equation 3.4) against the radius of the Gaussian weight disk (where σ is given in Equation 3.7):

$$\epsilon_0(\mid 1-\alpha_i-\beta_i\mid + \mid \alpha_i \mid + \mid \beta_i \mid +1) \geq 2\sigma_0\sqrt{(1-\alpha_i-\beta_i)^2+\alpha_i^2+\beta_i^2+1}$$

This inequality holds because of the triangle inequality. For the comparison, $\epsilon_0 = 2\sigma_0$.

3.6 Scoring Algorithm with Gaussian Error

The exact method of determining a score for a correspondence is given by the following algorithm:

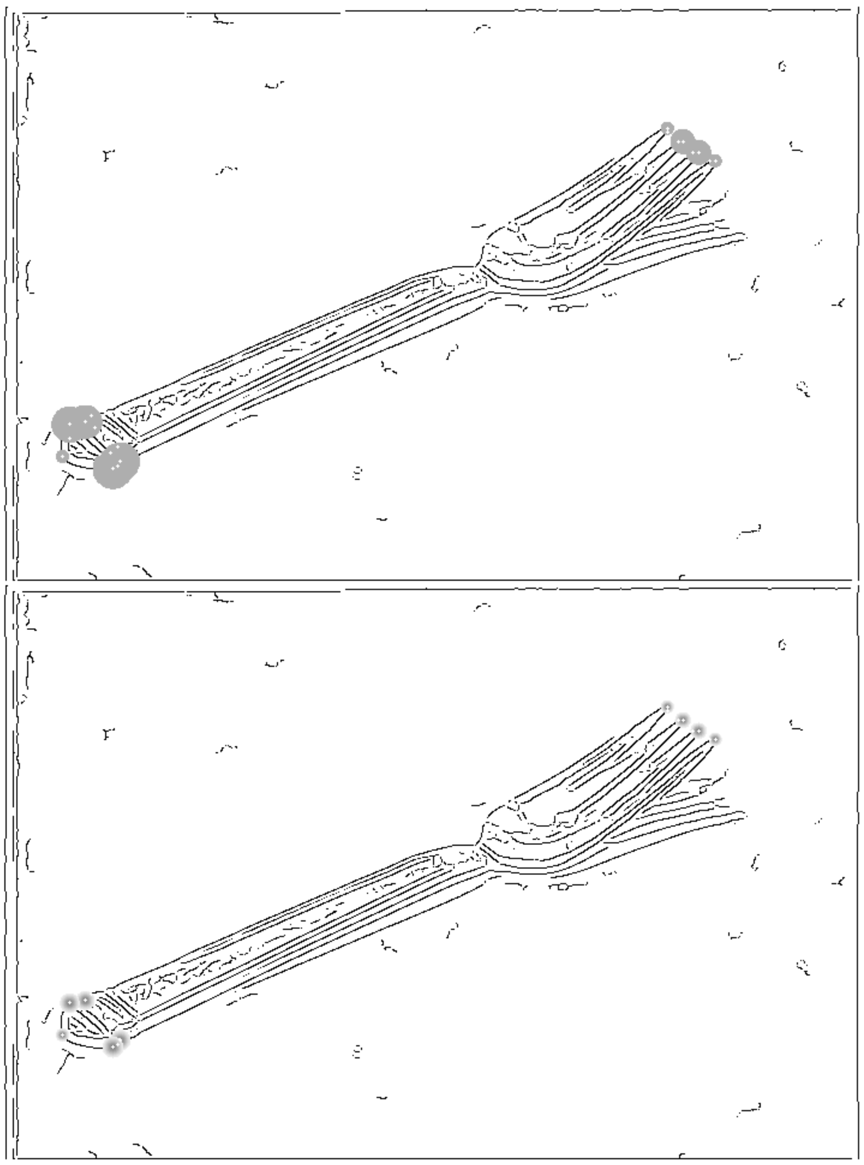


Figure 3-2: The top and bottom figures show the location and density of the projected uniform and Gaussian weight disks, respectively. The darkness of the disk indicates the weight accorded a point which falls at that location. The three points used for the matching are the bottom tip of the fork and the ends of the two outer prongs. The image points found within the weight disks are indicated as small white dots. Note that the uniform disks are bigger and more diffuse than the Gaussian disks.

(a) for a hypothesis $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2) : (\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2)$, find all Gaussian weight disk locations and sizes:

   (i) find affine coordinates $\mathbf{m}_j = (\alpha_j, \beta_j)$ with respect to basis $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2)$

   (ii) projected image location for $\mathbf{m}_j$ is $\mathbf{s}_0 + \alpha_j(\mathbf{s}_1 - \mathbf{s}_0) + \beta_j(\mathbf{s}_2 - \mathbf{s}_0)$

   (iii) projected Gaussian weight disk radius for $\mathbf{m}_j$ is $2\sigma = 2(\alpha_j^2 + \beta_j^2 + [1 - \alpha_j - \beta_j]^2)$

(b) for every image point $\mathbf{s}_j$, initially set $d = \infty$.

   (i) find the minimum distance d between $\mathbf{s}_j$ and $\mathbf{m}_j$ such that $d \leq 2\sigma$.

   (ii) add $v = \frac{1}{2\pi\sigma^2} e^{-\frac{d^2}{2\sigma^2}}$ (the height of the Gaussian weight disk at the image point location) to the sum w, which is the total score for this hypothesis. If this image point did not come within 2σ of any projected model point, then $v = 0$.

The collection method seems somewhat nonintuitive in that we accumulate evidence from every image point, instead of taking the contribution from at most one point per projected weight disk. The reason we chose to associate a random variable with each image point, rather than each weight disk, is that it is difficult to work with sums of random variables whose density function involves the max function. In Chapter 5 we will examine the implications of accumulating weight from at most a single image point per weight disk.

Now that we have selected a weighting scheme and a particular algorithm for accumulating scores for hypotheses, we can determine the score density associated with correct and incorrect hypotheses. As we can see from step (b:ii) in the algorithm, the score is a sum over the individual weights from the n image points (not including the three used for the basis correspondence). First we will define the random variables describing the score contributions from the individual image points.

Suppose we are testing a correct hypothesis. Then a particular model point $\mathbf{m}_i$ will give rise to an image point which falls within the projected Gaussian weight disk for model point $\mathbf{m}_i$ 86.5% of the time, since the weight disk only extends to a radius of $2\sigma_e$. The weight that this image point yields using our weighting scheme can be described by a random variable which we will call V_M. For convenience we will refer to such an image point as a "true" image point. To demonstrate what this means, in the simpler bounded uniform error case and with c denoting the probability of occlusion, the density of V_M is:

$$f_{V_M}(v) \quad = \quad c\delta(0) + (1 - c)\delta(v - 1)$$

where δ is the unit impulse function. This indicates the fact that when we are testing a true image point, it will always appear inside a projected weight disk and contribute a score of 1, unless it is occluded, in which case it contributes 0.

We also define the random variable $V_{\overline{M}}$ to describe the weight that a random point will yield for a tested hypothesis, where the term "random point" is taken to mean an image point which either does not arise from the model at all, or that does arise from the model, but the hypothesis being tested is incorrect. Note that we are assuming by this that a point which does arise from the model will contribute the same as a clutter point when the correspondence being tested is incorrect.

Next, we define the random variables W_H and $W_{\overline{H}}$ to describe the cumulative weight of correct and incorrect hypotheses. We let $n+3$ be the number of image points and $m+3$ be the number of model points. Then $W_{\overline{H}}$ is defined as

$$W_{\overline{H}} = \sum_{i=1}^{n} V_{\overline{M}} \tag{3.8}$$

where the n in the sum is due to the fact that 3 image points are used in the basis correspondence. Note that when we are testing an incorrect hypothesis we consider all the image points to be random, whether or not the model appears in the image.

The expression for W_H is slightly more complicated because of occlusion. If there were no occlusion, then W_H would receive contributions from m projected model points and $(n-m)$ clutter points, that is:

$$W_H = \sum_{i=1}^{m} V_M + \sum_{i=1}^{n-m} V_{\overline{M}} \tag{3.9}$$

$$\tag{3.10}$$

When $c \neq 0$, we observe n clutter points but we do not know how many of them arise from the model. The number of projected model points that we observe is actually a binomially distributed random variable M. Thus,

$$W_H = \sum_{i=1}^{M} V_M + \sum_{i=1}^{n-M} V_{\overline{M}} \tag{3.11}$$

$$\mathrm{P}\{M=k\} = \binom{m}{k}(1-c)^k c^{m-k} \tag{3.12}$$

To discriminate between correct and incorrect hypotheses, we must know the score that a correct hypothesis is likely to have versus an incorrect one. For this we need to first determine the probability densities of the variables V_M and $V_{\overline{M}}$ and subsequently the densities of W_H and $W_{\overline{H}}$. The derivations for the density of V_M and $V_{\overline{M}}$ given a particular value for the size of the weight disk is straightforward; however the size of the weight disk is itself dependent on the affine coordinates of the model points. We will define another random variable, σ_e, to describe the values of the standard deviation of the projected Gaussian error distribution, and we will discuss the how we estimate its density in the next section. Once we have this expression, we will find the densities of V_M and $V_{\overline{M}}$ by integrating the expressions:

$$\begin{aligned} f_{V_M}(v) &= \int_0^\infty f_{V_M|\sigma_e}(v \mid \sigma) f_{\sigma_e}(\sigma) d\sigma \\ f_{V_{\overline{M}}}(v) &= \int_0^\infty f_{V_{\overline{M}}|\sigma_e}(v \mid \sigma) f_{\sigma_e}(\sigma) d\sigma \end{aligned}$$

3.6.1 Determining the Density of σ_e

The motivation for treating the weight disk radius as a random variable is that we would like to remove the dependence of σ_e from the geometry of any particular model. Rather, we would like to find an expression for the probability density of the weight disk radius over all possible models. To do this, we estimate the density by generating thousands of models and keeping a histogram of the affine coordinates of all the model points in the affine space formed by randomly chosen model bases.

Specifically, the method is as follows: when a particular hypothesis is being evaluated, each model point is projected into the image with a weight disk whose radius is a function of the affine coordinates of the model point:

$$2\sigma = 2\sigma_0\sqrt{(1-\alpha_i-\beta_i)^2+\alpha_i^2+\beta_i^2+1}$$

in which σ_0, the standard deviation of the sensed Gaussian error, is a constant which must be determined empirically. We define a random variable σ_e which takes on the values of σ in the above expression, and in order to remove the dependence of σ_e on the constant σ_0, we define another random variable

$$\rho_e = \sqrt{(1-\alpha_i-\beta_i)^2+\alpha_i^2+\beta_i^2+1} \tag{3.13}$$

and we set

$$\sigma_e = \sigma_0\rho_e \tag{3.14}$$

In the analysis we use two different probability densities for ρ_e, one for correct basis matchings and one for incorrect basis matchings. Intuitively, this is due to the fact that when incorrect basis matchings are tested, more often than not the projected model points fall outside the image range and are thrown away, while when correct hypotheses are tested the remaining model points always project to within the image. In tests we have observed that over half of incorrect hypotheses tested are rejected for this reason, leading to an altered density for ρ_e.

Let us call the two densities $f_{\rho_e|H}$ and $f_{\rho_e|\overline{H}}$. We empirically estimate the former density by generating a random model of size 25, then for each ordered triple of model points as basis, we increment a histogram for the value of ρ_e as a function of α and β for all the other model points with respect to that basis. For the latter density, we generate a random model of size 4 and a random image, and histogram the values of ρ_e for only those cases in which the initial basis matching causes the remaining model point to fall within the image. The densities for ρ_e found in this manner have been observed to be surprisingly invariant over numbers of model points ranging from 1 to 30, over numbers of image points ranging from 1 to 1000, and even across ranges of σ_0 differing by as much as 10 to 1 (using a fixed image size).

The model is constrained such that the maximum distance between any two model points is not greater than 10 times the minimum distance, and in the basis selection, no basis is chosen such that the angle ψ between the two axes is $0 \leq \mid \psi \mid \leq \frac{\pi}{16}$ or $\frac{15}{16}\pi \leq \psi \leq \frac{17}{16}\pi$. This is done to avoid unstable bases, thereby bounding the size of the affine coordinates. For example, the coordinates of the point $P = (1, 1)$ with respect to the bases $(1, 0)$ and $(-1, 0)$ is (∞, ∞). A similar problem exists for the same point P with respect to the bases $(1, 0)$ and $(0, 0)$. The minimum value of ρ_e is found analytically by minimizing Equation 3.13 with respect to α and β, and occurs at $\alpha = \beta = \frac{1}{3}$, where $\rho_e = (\frac{4}{3})^{1/2}$. The maximum value of ρ_e occurs at the boundary conditions discussed above, and was determined empirically to be ≈ 40.

The results were almost identical in every test we ran; two typical normalized histograms are shown in Figure 3-3. The histograms very closely fit the curves

$$f_{\rho_e|H}(\rho) = a_1\rho^{-2} \qquad a_1 = 1.189$$

and

$$f_{\rho_e|\overline{H}}(\rho) = a_0\rho^{-4} \qquad a_0 = 4.624$$

between the ranges $r_1 = \sqrt{\frac{4}{3}}, r_2 = 40$. Note that the actual value for r_2 is not crucial, given the actual density functions — in fact, the difference in the analysis using $r_2 = 40$ or $r_2 = \infty$ turns out to be very small. Figure 3-3 shows the estimated density functions for ρ_e superimposed on the empirical density functions. The integral of the analytic expression thus defined = 1.018 and 1.052, respectively.

Using Equation 3.14, the density of ρ_e implies the density of σ_e:

$$f_{\sigma_e|H}(\sigma) = b_1\sigma^{-2} \qquad b_1 = a_1\sigma_0$$

and

$$f_{\sigma_e|\overline{H}}(\sigma) = b_0\sigma^{-4} \qquad b_0 = a_0\sigma_0^3$$

between the ranges $s_1 = \sigma_0 r_1$ and $s_2 = \sigma_0 r_2$. For the rest of the work we will work with the variable σ_e rather than ρ_e for convenience, keeping in mind that in the final analysis, the terms a_0, a_1, r_1 and r_2 are constants, and the terms b_0, b_1, s_1 and s_2, are variables dependent on them and on the value of σ_0.

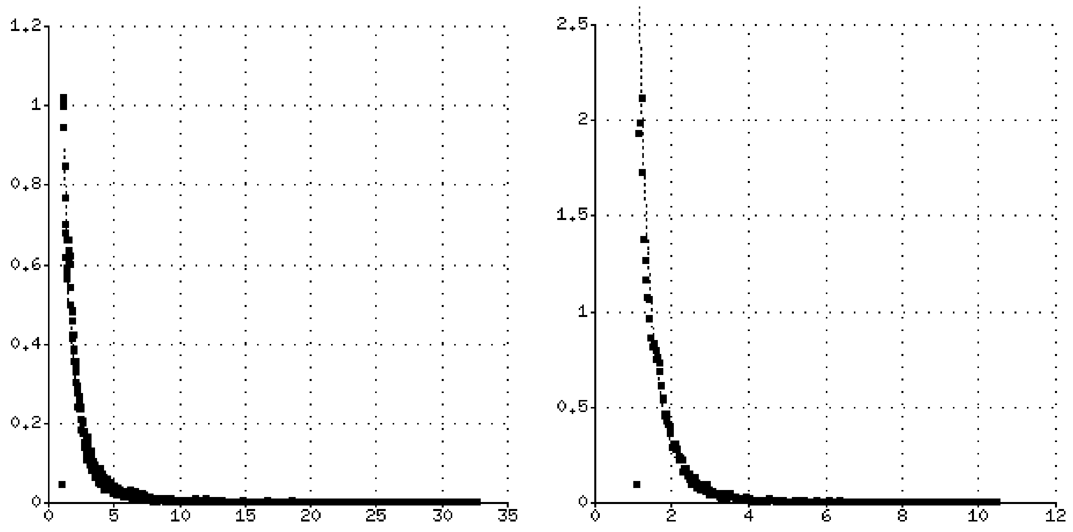


Figure 3-3: The density functions $f_{\rho_e|H}(\rho)$ and $f_{\rho_e|\overline{H}}(\rho)$, respectively.

3.6.2 Determining the Average Covariance of Two Projected Error Distributions

We have already derived the covariance between two projected error distributions, once the model to image correspondence has been fixed. This was shown to be

$$\mathrm{Cov}\,(\Delta\hat{\mathbf{s}}_i, \Delta\hat{\mathbf{s}}_j) \;=\; [(1-\alpha_i-\beta_i)(1-\alpha_j-\beta_j)+\alpha_i\alpha_j+\beta_i\beta_j]\,\sigma_0^2 I$$

The probability density of the expression $(1-\alpha_i-\beta_i)(1-\alpha_j-\beta_j)+\alpha_i\alpha_j+\beta_i\beta_j$ can be estimated in a similar manner as in the previous section to determine the average covariance between projected error distributions. The actual experiment performed was: for 1000 randomly generated models, subject to the same constraints as in the previous section, 25 random model bases were chosen (again, subject to the same constraints as in the previous section). For each basis, 25 random model pairs were tested for the value of the covariance, and the result histogrammed. The results indicated that the average covariance was always positive. The implication of the average covariance being positive is that our estimate of the variance of W_H will be too low, as we will observe.

3.7 Deriving the Single Point Densities

3.7.1 Finding $f_{V_M}(v)$

Given a correct hypothesis and no occlusion, the possible locations of a projected model point $\mathbf{m}_i$ can be modeled as a vector $\hat{\mathbf{s}}_i + \mathbf{e}_i$, where $\mathbf{e}_i = [R, \Theta]^T$. Let us treat the vector $\mathbf{e}_i$ as a pair of random variables R and Θ to avoid defining new notation. Then $\mathbf{e}_i$ given a fixed weight disk size is a displacement vector with Gaussian distribution (expressed in polar coordinates)

$$f_{R,\Theta|\sigma_e}(r, \theta \mid \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{r^2}{2\sigma^2}}$$

We now choose an evaluation function $g(r, \theta)$, which we use to weight a match that is offset by $\mathbf{e}_i$ from the predicted match location. We want to find its density, *i.e.*, we want $f_{g(R,\Theta)|\sigma_e}(v)$, where the joint density of R and Θ is as stated. As mentioned, we choose the evaluation function

$$g(r, \theta) = \frac{1}{2\pi\sigma^2} e^{-\frac{r^2}{2\sigma^2}}$$

We are assuming that the value of σ is fixed without actually denoting this in the function g. Since the evaluation function g is a really function of r alone, we need to know the density function of r. To find this, we integrate $f_{R,\Theta|\sigma_e}(r, \theta \mid \sigma)$ over θ:

$$f_{R|\sigma_e}(r \mid \sigma) = \int_0^{2\pi} f_{R,\Theta|\sigma_e}(r, \theta \mid \sigma) r d\theta = \frac{r}{\sigma^2} e^{\frac{-r^2}{2\sigma^2}}$$

Next, we want to find the density of the weight function $v = g(R)$. The change of variables formula for a monotonically decreasing function is given by Equation A.1, restated here:

$$f_{g(R)}(v) = \frac{-f_R(g^{-1}(v))}{g'(g^{-1}(v))}$$

Working through the steps, we find

$$\begin{aligned}
g(r) &= \frac{1}{2\pi\sigma^2} e^{-\frac{r^2}{2\sigma^2}} \\
g'(r) &= -\frac{r}{\sigma^2} g(r) \\
f_{R|\sigma_e}(r \mid \sigma) &= \frac{r}{\sigma^2} e^{\frac{-r^2}{2\sigma^2}} \\
&= 2\pi r g(r) \\
f_{R|\sigma_e}(g^{-1}(v) \mid \sigma) &= 2\pi g^{-1}(v) g(g^{-1}(v)) \\
&= 2\pi v g^{-1}(v) \\
g'(g^{-1}(v)) &= -\frac{g^{-1}(v)}{\sigma^2} g(g^{-1}(v))
\end{aligned}$$

$$\begin{aligned} &= -\frac{g^{-1}(v)}{\sigma^2}v \\ \Longrightarrow f_{g(R)|\sigma_e}(v \mid \sigma) &= \frac{-f_{R|\sigma_e}(g^{-1}(v))}{g'(g^{-1}(v))} \\ &= 2\pi v g^{-1}(v)\frac{\sigma^2}{vg^{-1}(v)} \\ &= 2\pi\sigma^2 \end{aligned}$$

It may seem counterintuitive that the resulting distribution is constant. However, this can be understood if one considers an example in which $f_{R,\Theta}(r,\theta)$ is uniformly distributed. Integrating over all angles yields a linearly increasing function in r. Assigning an evaluation function $g(r,\theta)$ which is inversely proportional to r yields a constant density function on $f_{R|\sigma_e}(v \mid \sigma)$. The same thing is happening here, only quadratically. Since we only score a point if it falls within a radius of 2σ from the center, we miss the entire part of the distribution from a radius of 2σ to ∞, which as we showed before is e^{-2}. So the probability density of V_M given a fixed sized weight disk is:

$$f_{V_M|\sigma_e}(v \mid \sigma) = \begin{cases} e^{-2}\delta(v) & v = 0 \\ 2\pi\sigma^2 & \frac{1}{2\pi\sigma^2 e^2} \le v \le \frac{1}{2\pi\sigma^2} \\ 0 & \text{otherwise} \end{cases}$$

This expression correctly integrates to 1.

We need to integrate this expression over all values of σ_e. Dealing first with the case first where $v \neq 0$, we get:

$$\begin{aligned} f_{V_M}(v) &= \int f_{V_M|\sigma_e}(v \mid \sigma) f_{\sigma_e|H}(\sigma) d\sigma \\ &= \int 2\pi\sigma^2 b_1 \sigma^{-2} d\sigma \\ &= 2\pi b_1 \int d\sigma \end{aligned}$$

There are two things to take into consideration when calculating the limits for the integration: first, the possible values of σ_e range from a lower limit s_1 to an upper limit s_2, due to limits on the values of the affine coordinates. Also, for a given $\sigma_e = \sigma$, it is clear that the maximum value we can achieve is when $r = 0 \Rightarrow v = \frac{1}{2\pi\sigma^2}$, and the minimum value we can achieve is at the cutoff point $r = 2\sigma \Rightarrow v = \frac{1}{2\pi\sigma^2}e^{-2}$. Setting v to each of these expressions and solving for σ leads to the conclusion that for a particular value v, the only values for σ_e such that $g(\mathbf{e}_i \mid \sigma_e)$ could equal v are in the range $(\frac{1}{\sqrt{2\pi v e}}, \frac{1}{\sqrt{2\pi v}})$. Therefore the lower bound on the integral is $\sigma = \max(s_1, \frac{1}{\sqrt{2\pi v e}})$, and the upper bound is $\sigma = \min(\frac{1}{\sqrt{2\pi v}}, s_2)$.

The bounds over which the integration is performed is illustrated in Figure 3-4. The third dimension of this graph (not shown) is the joint density function f_{V_M,σ_e}. Conceptually what we are doing is integrating over the σ axis. We split this integral into the three 3 regions defined by the integration bounds, and deal with the case $v = 0$

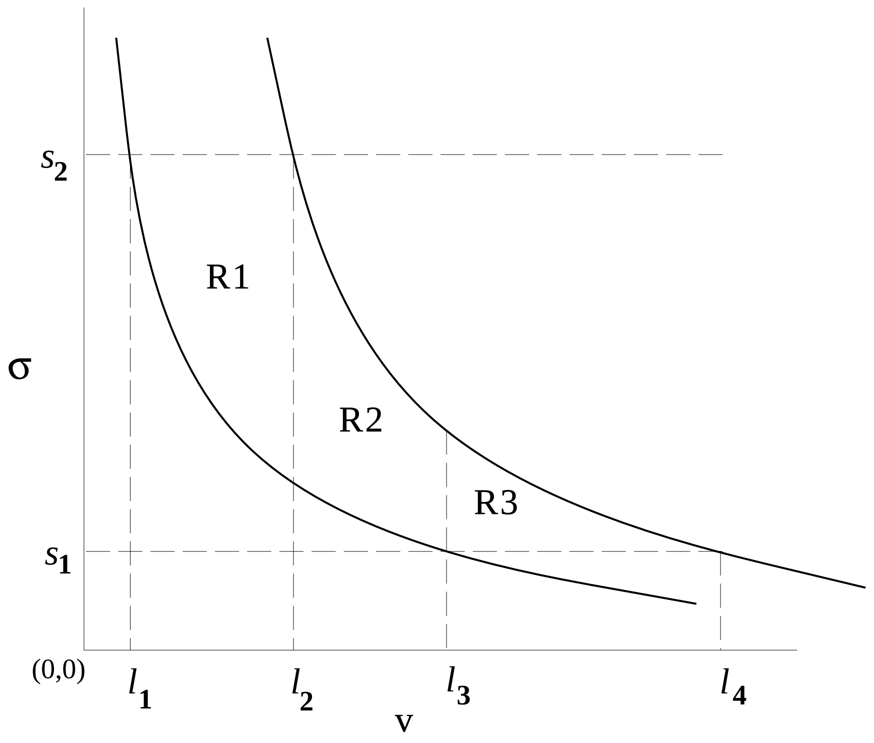


Figure 3-4: The figure shows the boundaries for the integration for both $f_{V_M}(v)$ and $f_{V_{\overline{M}}}(v \mid m = 1)$. The bottom curve is $\sigma = \frac{1}{\sqrt{2\pi v}}$, and the upper curve is $\sigma = \frac{1}{e\sqrt{2\pi v}}$ The third dimension of the graph (not illustrated) are the joint density functions $f_{V_M,\sigma_e}(v, \sigma)$ and $f_{V_{\overline{M}},\sigma_e}(v, \sigma \mid m = 1)$.

separately. Integrating, we get:

$$f_{V_M}(v) = \begin{cases} e^{-2}\delta(v) & \text{v=0} \\ 2\pi b_1(s_2 - \frac{1}{e\sqrt{2\pi v}}) & \ell_1 < v \leq \ell_2 \\ \pi b_1 \frac{e-1}{e\sqrt{2\pi v}} & \ell_2 < v \leq \ell_3 \\ 2\pi b_1(\frac{1}{\sqrt{2\pi v}} - s_1) & \ell_3 < v \leq \ell_4 \\ 0 & \text{otherwise} \end{cases}$$

where

$$\ell_1 = \frac{1}{2\pi {s_2}^2 e^2} \qquad \ell_2 = \frac{1}{2\pi {s_2}^2}$$

$$\ell_3 = \frac{1}{2\pi {s_1}^2 e^2} \qquad \ell_4 = \frac{1}{2\pi {s_1}^2}$$

and s_1, s_2 are the minimum and maximum allowable values for σ_e, respectively. This expression is graphed on the left in Figure 3-5 for $\sigma_0 = 2.5$. We break the equation into cases not out of necessity, but for legibility.

3.7.2 Finding $f_{V_{\overline{M}}}(v \mid m = 1)$

We do the same derivation for the distribution $f_{V_{\overline{M}}}(v \mid m = 1)$, that is, the value that a random point contributes to the basis of a model with 4 points. This is a prerequisite for finding the distribution for the case where the model has $m + 3$ points. Given a hypothesis and a random point, we calculate the distribution as follows: let event E = "point falls in a single hypothesized weight disk". Given $\sigma_e = \sigma$, the probability of event E equals the area of the weight disk divided by the area of the image A, *i.e.*,

$$\mathrm{P}\{E \mid \sigma_e = \sigma\} = \frac{4\pi\sigma^2}{A} \qquad \mathrm{P}\Big\{\overline{E} \mid \sigma_e = \sigma\Big\} = \frac{A - 4\pi\sigma^2}{A}$$

Now we calculate the probability that a point which is uniformly distributed inside a disk of radius 2σ contributes value v for an incorrect hypothesis, using the weighting function defined in the previous section. As before, we express a uniform distribution as a pair of random variables R, Θ, and then integrate over Θ to get the density of R alone, since the evaluation function g is a function of R:

$$\begin{aligned} f_{R,\Theta|\sigma_e}(r, \theta \mid \sigma) &= \frac{1}{\pi(2\sigma)^2} \\ f_{R|\sigma_e}(r \mid \sigma) &= \int_0^{2\pi} \frac{1}{\pi(2\sigma)^2} r\, d\theta \\ &= \frac{r}{2\sigma^2} \end{aligned}$$

As before, we calculate the density $f_{g(R)|\sigma_e}$ given that the clutter point falls in the weight disk with the new distribution for R and get:

$$\begin{aligned} f_{g(R)|\sigma_e}(v \mid \sigma, E) &= \frac{-f(g^{-1}(v))}{g'(g^{-1}(v))} \\ &= \frac{1}{2} v^{-1} \end{aligned}$$

Therefore, the density of $V_{\overline{M}}$ given a single weight disk with fixed σ_e is:

$$f_{V_{\overline{M}}|\sigma_e}(v \mid \sigma, m = 1) = \begin{cases} \mathrm{P}\Big\{\overline{E} \mid \sigma_e = \sigma\Big\}\,\delta(v) = [\frac{A - 4\pi\sigma_e^2}{A}]\delta(v) & v = 0 \\ f_{V_{\overline{M}}|\sigma_e}(v \mid \sigma, E)\mathrm{P}\{E \mid \sigma_e = \sigma\} = \frac{2\pi\sigma^2}{Av} & \frac{1}{2\pi\sigma^2 e^2} \le v \le \frac{1}{2\pi\sigma^2} \\ 0 & \text{otherwise} \end{cases}$$

Again, this expression correctly integrates to 1. As before, we need to integrate over σ_e:

$$\begin{aligned} f_{V_{\overline{M}}}(v \mid m = 1) &= \int f_{V_{\overline{M}}|\sigma_e}(v \mid \sigma, m = 1) f_{\sigma_e|\overline{H}}(\sigma)\, d\sigma \\ &= \int \left(\frac{2\pi\sigma^2}{Av}\right) b_0 \sigma^{-4}\, d\sigma \end{aligned}$$

$$= \frac{2\pi b_0}{Av} \int \sigma^{-2} d\sigma$$

Dealing with $v = 0$ as a separate case, and with the same bounds as before, integrating yields:

$$f_{V_{\overline{M}}}(v \mid m = 1) = \begin{cases} [1 - \frac{4\pi b_0}{A}(\frac{1}{s_1} - \frac{1}{s_2})]\delta(v) & \text{v=0} \\ \frac{2\pi b_0}{Av}(e\sqrt{2\pi v} - \frac{1}{s_2}) & \ell_1 < v \leq \ell_2 \\ \frac{2\pi b_0}{Av}(e-1)\sqrt{2\pi v} & \ell_2 < v \leq \ell_3 \\ \frac{2\pi b_0}{Av}(\frac{1}{s_1} - \sqrt{2\pi v}) & \ell_3 < v \leq \ell_4 \\ 0 & \text{otherwise} \end{cases}$$

where

$$\ell_1 = \frac{1}{2\pi {s_2}^2 e^2} \qquad \ell_2 = \frac{1}{2\pi {s_2}^2}$$

$$\ell_3 = \frac{1}{2\pi {s_1}^2 e^2} \qquad \ell_4 = \frac{1}{2\pi {s_1}^2}$$

This function is illustrated in on the right of Figure 3-5 for a value of $\sigma_0 = 2.5$.

We ran two simulations to verify the analysis of this section. In the first one, we tested the density function $f_{V_M}(v)$ as follows: we generated a random model of size 4, chose a random $3D$ pose and scale, projected the model into an image adding a Gaussianly distributed displacement vector to each point, chose a correct image to model correspondence, and histogrammed the value of the fourth point. This was repeated $15,000$ times. The second simulation differed only in that instead of projecting the model into the image, a random image was created and a random correspondence tested. The results of the simulations are also shown in Figure 3-5. Both graphs show a normalized histogram of the results of $15,000$ independent trials excluding the value at $v = 0$. The measured density of $f_{V_M}(v)$ does not fit the prediction at $v = 0$ because of binning problems at that value, but the rest of the first graph indicates the empirical results corroborating the predictions very closely. For the second graph, most of the density occurs at $v = 0$; for the remainder of the distribution a chi-squared test shows no significant difference between the empirical and analytic distributions (probability = .98 for $\chi^2 = 160$ with 199 degrees of freedom).

3.8 Deriving the Accumulated Densities

Having found the single point densities, we use them to find the density of the combined weight of points for correct and incorrect hypotheses.

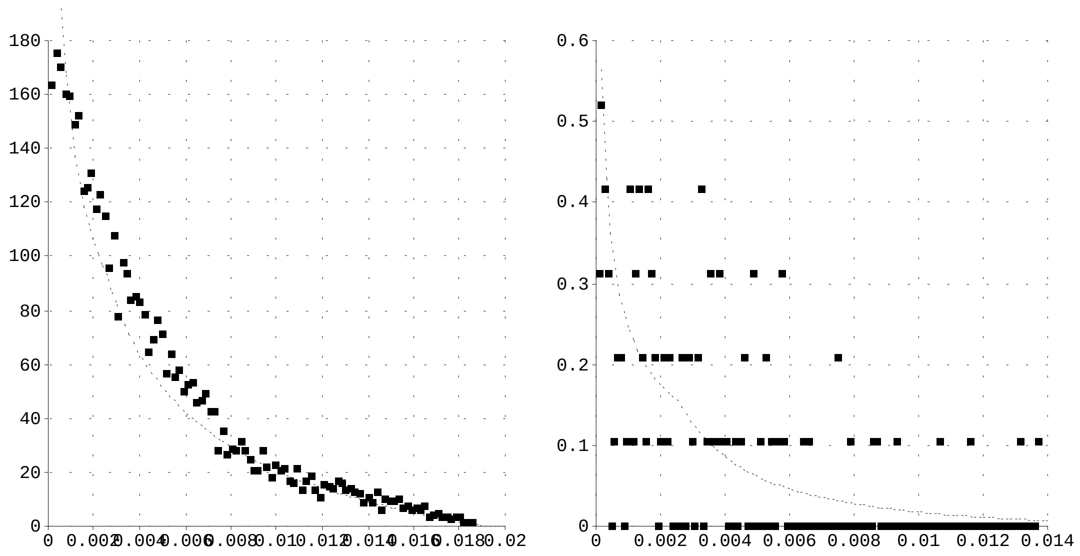


Figure 3-5: Distributions $f_{V_M}(v)$ and $f_{V_{\overline{M}}}(v \mid m = 1)$ for $v > 0$. For these distributions, $\sigma_0 = 2.5$. Note that the scale of the y axis in the first graph is approximately ten times greater than that of the second.

3.8.1 Finding $f_{W_H}(w)$

For a model of size $m + 3$ and an image of size $n + 3$, when a correct hypothesis is being tested, then there are M true points in the image, and $n - M$ random points, where M is binomially distributed and

$$\mathrm{P}\{M = k\} = \binom{m}{k}(1-c)^k c^{m-k}$$

The weight we collect for this hypothesis is a random variable with probability density

$$\begin{aligned} f_{W_H}(w) &= \overbrace{f_{V_M}(v) \otimes \cdots f_{V_M}(v)}^{M} \otimes \overbrace{f_{V_{\overline{M}}}(v) \otimes \cdots f_{V_{\overline{M}}}(v)}^{n-M} \\ &= \bigotimes_{i=1}^{M} f_{V_M}(v) \otimes \bigotimes_{i=1}^{n-M} f_{V_{\overline{M}}}(V_{\overline{M}}) \end{aligned}$$

where $\otimes$ denotes convolution. The above shortened notation will be used from now on for convenience. This formula assumes that each point contributes weight to its supporting basis independently of any other.

In order to avoid explicitly convolving the preceding distributions, we find the expected value and the standard deviation of V_M and $V_{\overline{M}}$, and invoke the central limit theorem to claim that the combined weight of a correct correspondence between a size $m + 3$ model in a size $n + 3$ image should roughly follow a normal distribution. The fact that M is binomially distributed when $c \neq 0$ means that this distribution will not be completely Gaussian, but we will assume that it is for simplicity. Again, $N(m, \sigma^2)$ denotes a normal distribution with mean m and variance σ^2:

$$f_{W_H} \sim N\left(\mathrm{E}\left[\sum_1^M V_M + \sum_1^{n-M} V_{\overline{M}}\right], \mathrm{Var}\left(\sum_1^M V_M + \sum_1^{n-M} V_{\overline{M}}\right)\right).$$

Lastly, we use the formulas for conditional mean and variance, given by Equations A.2 and A.3, restated here, to simplify the above expression:

$$\begin{aligned}
\mathrm{E}[X] &= \mathrm{E}[\mathrm{E}[X \mid Y]] \\
\mathrm{Var}\,(X) &= \mathrm{E}[\mathrm{Var}\,(X \mid Y)] + \mathrm{Var}\,(\mathrm{E}[X \mid Y])
\end{aligned}$$

Solving in stages, we find:

$$\begin{aligned}
\mathrm{E}\left[\sum_1^M V_M + \sum_1^{n-M} V_{\overline{M}} \mid M\right] &= M\mathrm{E}[V_M] + (n-M)\mathrm{E}[V_{\overline{M}}] \\
\mathrm{Var}\left(\sum_1^M V_M + \sum_1^{n-M} V_{\overline{M}} \mid M\right) &= M\mathrm{Var}\,(V_M) + (n-M)\mathrm{Var}\,(V_{\overline{M}}) \\
\mathrm{E}\left[\mathrm{E}\left[\sum_1^M V_M + \sum_1^{n-M} V_{\overline{M}} \mid M\right]\right] &= \mathrm{E}[M]\,\mathrm{E}[V_M] + \mathrm{E}[n-M]\,\mathrm{E}[V_{\overline{M}}] \\
\mathrm{E}\left[\mathrm{Var}\left(\sum_1^M V_M + \sum_1^{n-M} V_{\overline{M}} \mid M\right)\right] &= \mathrm{E}[M]\,\mathrm{Var}\,(V_M) + \mathrm{E}[n-M]\,\mathrm{Var}\,(V_{\overline{M}}) \\
\mathrm{Var}\left(\mathrm{E}\left[\sum_1^M V_M + \sum_1^{n-M} V_{\overline{M}} \mid M\right]\right) &= \mathrm{Var}\,(M)\,\mathrm{E}[V_M]^2 + \mathrm{Var}\,(n-M)\,\mathrm{E}[V_{\overline{M}}]^2
\end{aligned}$$

We use the values $\mathrm{E}[M] = (1-c)m$ and $\mathrm{Var}\,(M) = mc(1-c)$, and use the above formulas to find that:

$$f_{W_{\overline{H}}} \sim N(A, B)$$

in which

$$\begin{aligned}
A &= (1-c)m\mathrm{E}[V_M] + [n-(1-c)m]\mathrm{E}[V_{\overline{M}}] \\
B &= (1-c)m\mathrm{Var}\,(V_M) + mc(1-c)\mathrm{E}[V_M]^2 \\
&\quad +[n-(1-c)m]\mathrm{Var}\,(V_{\overline{M}}) + mc(1-c)\mathrm{E}[V_{\overline{M}}]^2
\end{aligned}$$

Solving for the remaining terms, we find

$$\mathrm{E}[V_M] = \int_0^{\ell_4} v f_{V_M}(v)dv$$

Integrating over the four regions of the distribution and using the equalities

$$\ell_1 = \frac{1}{2\pi {s_2}^2 e^2} \qquad \ell_2 = \frac{1}{2\pi {s_2}^2} \qquad \ell_3 = \frac{1}{2\pi {s_1}^2 e^2} \qquad \ell_4 = \frac{1}{2\pi {s_1}^2}$$

yields

$$\mathrm{E}[V_M] \quad = \quad b_1 \frac{e^4 - 1}{12\pi e^4} \left[\frac{1}{s_1^3} - \frac{1}{s_2^3} \right].$$

Further substitutions of the equalities

$$b_1 = a_1\sigma_0 \qquad s_1 = \sigma_0 r_1 \qquad s_2 = \sigma_0 r_2$$

from Section 3.6.1 yield

$$\mathrm{E}[V_M] \quad = \quad a_1 \frac{e^4 - 1}{12\pi\sigma_0^2 e^4} \left[\frac{1}{r_1^3} - \frac{1}{r_2^3} \right].$$

Finally, the substitutions $a_1 = 1.189, r_1 = \sqrt{\frac{4}{3}}, r_2 = 40$, also from Section 3.6.1, yield

$$\mathrm{E}[V_M] \quad = \quad 2.01 \times 10^{-2} \times \frac{1}{\sigma_0^2}$$

The remaining terms are found using the same steps:

$$\begin{aligned}
\mathrm{E}\left[V_M^2\right] &= \int_0^{\ell_4} v^2 f_{V_M}(v)dv \\
&= b_1 \frac{e^6 - 1}{60\pi^2 e^6} \left[\frac{1}{s_1^5} - \frac{1}{s_2^5} \right] \\
&= a_1 \frac{e^6 - 1}{60\pi^2 \sigma_0^4 e^6} \left[\frac{1}{r_1^5} - \frac{1}{r_2^5} \right] \\
&= 9.76 \times 10^{-4} \times \frac{1}{\sigma_0^4} \\
\mathrm{Var}\left(V_M\right) &= \mathrm{E}\left[V_M^2\right] - \mathrm{E}[V_M]^2
\end{aligned}$$

Note that the value of the limit r_2 was determined empirically and is a function of the constraints on the bases that were chosen. Without the basis constraints, r_2 tends to infinity, and in fact the values of these parameters for $r_2 = 40$ and $r_2 = \infty$ are not significantly different.

The values of $\mathrm{E}[V_{\overline{M}}]$ and $\mathrm{Var}\left(V_{\overline{M}}\right)$ are derived in the next section.

3.8.2 Finding $f_{W_{\overline{H}}}(w)$

For an incorrect hypothesis we look at the problem in two steps. First we derive, as above, the mean and standard deviation of $V_{\overline{M}} \mid m=1$, *i.e.*, the weight of a single random image point that drops into a single weight disk. From the distribution $f_{V_{\overline{M}}}(v \mid m=1)$, we find:

$$\begin{aligned}
\mathrm{E}[V_{\overline{M}} \mid m=1] &= \int_0^{\ell_4} v f_{V_{\overline{M}}}(v \mid m=1) dv \\
&= \frac{b_0(e^2-1)}{3e^2 A}\left[\frac{1}{s_1^3} - \frac{1}{s_2^3}\right]
\end{aligned}$$

Substituting $b_0 = a_0\sigma_0^3$ from Section 3.6.1, we get:

$$\mathrm{E}[V_{\overline{M}} \mid m=1] = \frac{a_0(e^2-1)}{3e^2 A}\left[\frac{1}{r_1^3} - \frac{1}{r_2^3}\right]$$

Lastly, we note from Section 3.6.1 that

$$a_0\left[\frac{1}{3r_1^3} - \frac{1}{3r_2^3}\right] = 1$$

since this is exactly the integral of the density $f_{\rho_e|\overline{H}}$. Therefore,

$$\begin{aligned}
\mathrm{E}[V_{\overline{M}} \mid m=1] &= \frac{e^2-1}{e^2 A} \\
&= \frac{.8656}{A}
\end{aligned}$$

We continue with $\mathrm{E}\left[V_{\overline{M}}^2 \mid m=1\right]$:

$$\begin{aligned}
\mathrm{E}\left[V_{\overline{M}}^2 \mid m=1\right] &= \int_0^{\ell_4} v^2 f_{V_{\overline{M}}}(V \mid m=1) dv \\
&= \frac{b_0(e^4-1)}{20e^4 A\pi}\left[\frac{1}{s_1^5} - \frac{1}{s_2^5}\right] \\
&= \frac{a_0(e^4-1)}{20e^4 A\pi\sigma_0^2}\left[\frac{1}{r_1^5} - \frac{1}{r_2^5}\right]
\end{aligned}$$

Substituting $a_0 = 4.624$, $r_1 = \sqrt{\frac{4}{3}}$, and $r_2 = 40$, from Section 3.6.1:

$$\begin{aligned}
\mathrm{E}\left[V_{\overline{M}}^2 \mid m=1\right] &= 3.52 \times 10^{-2} \times \frac{1}{\sigma_0^2 A} \\
\mathrm{Var}\left(V_{\overline{M}} \mid m=1\right) &= \mathrm{E}\left[V_{\overline{M}}^2 \mid m=1\right] - \mathrm{E}[V_{\overline{M}} \mid m=1]^2
\end{aligned}$$

Note that the mean $\mathrm{E}[V_{\overline{M}} \mid m=1]$ is not dependent on the value of σ_0.

Now, consider a single random image point (*i.e.*, $n = 4$; three for the hypothesis and one left over) dropped into an image where a model of size $m + 3 > 4$ is hypothesized to be. In this case the event that the random point will contribute weight v to this hypothesis is calculated as follows: Let event E_i = "point drops in the ith weight disk." Then,

$$f_{V_{\overline{M}}}(v \mid v \neq 0) \;=\; f_{V_{\overline{M}}}(v, E_1) + f_{V_{\overline{M}}}(v, E_2) + \ldots + f_{V_{\overline{M}}}(v, E_m)$$

where we are assuming the disks are disjoint, hence we are overestimating the probability of the point falling in any disk. The actual rate of detection will be lower than our assumption, especially as the m grows large.

$$f_{V_{\overline{M}}}(v) \;=\; \begin{cases} 1 - \frac{m4\pi b_0}{A}[\frac{1}{s_1} - \frac{1}{s_2}] & \text{v=0} \\ \frac{m2\pi b_0}{Av}[e\sqrt{2\pi v} - \frac{1}{s_2}] & \ell_1 < v \leq \ell_2 \\ \frac{m2\pi b_0}{Av}[(e-1)\sqrt{2\pi v}] & \ell_2 < v \leq \ell_3 \\ \frac{m2\pi b_0}{Av}[\frac{1}{s_1} - \sqrt{2\pi v}] & \ell_3 < v \leq \ell_4 \\ 0 & \text{otherwise} \end{cases}$$

As m grows large, $(1 - m\frac{4\pi b_0}{A}[\frac{1}{s_1} - \frac{1}{s_2}]) < 0$ so this expression is no longer a density function. This is the point at which the model covers so much of the image that a random point will *always* contribute to some incorrect hypothesis. Therefore, this analysis only applies to models for which

$$m < \frac{A}{\sigma_0^2} * \left(4\pi a_0 \left[\frac{1}{r_1} - \frac{1}{r_2}\right]\right)^{-1} = \frac{A}{\sigma_0^2}.02034$$

using the equalities $b_0 = a_0\sigma_0^3$, $s_1 = \sigma_0 r_1$ and $s_2 = \sigma_0 r_2$ from Section 3.6.1. For a $\sqrt{A} : \sigma$ ratio of $200 : 1$, $m < 800$, and for a ratio of $50 : 1$, $m < 50$.

The mean and standard deviation for the weight of one random point dropping into an image with m weight disks is:

$$\begin{aligned} \mathrm{E}[V_{\overline{M}}] &= \int_0^{\ell_4} v f_{V_{\overline{M}}}(v)dv \\ &= m\mathrm{E}[V_{\overline{M}} \mid m = 1] \\ \mathrm{E}\left[V_{\overline{M}}^2\right] &= \int_0^{\ell_4} v^2 f_{V_{\overline{M}}}(v)dv \\ &= m\mathrm{E}\left[V_{\overline{M}}^2 \mid m = 1\right] \\ \mathrm{Var}\,(V_{\overline{M}}) &= \mathrm{E}\left[V_{\overline{M}}^2\right] - \mathrm{E}[V_{\overline{M}}]^2 \\ &= m\mathrm{E}\left[V_{\overline{M}}^2 \mid m = 1\right] - m^2\mathrm{E}[V_{\overline{M}} \mid m = 1]^2 \end{aligned}$$

Dropping n points convolves this distribution with itself n times:

$$f_{W_{\overline{H}}}(w) = \bigotimes_{i=1}^{n} f_{V_{\overline{M}}}(v)$$

H		Mean			Variance		
m	n	Emp	Pred	Emp/Pred	Emp	Pred	Emp/Pred
1	1	3.695E-3	3.218E-3	1.15	1.519E-5	1.462E-5	1.04
1	100	3.838E-3	3.534E-3	1.09	1.735E-5	1.668E-5	1.04
1	500	4.803E-3	4.812E-3	.998	2.227E-5	2.498E-5	.891
5	5	1.966E-2	1.609E-2	1.22	1.493E-4	7.312E-5	2.04
10	10	4.199E-2	3.218E-2	1.30	5.413E-4	1.462E-4	3.70
10	100	4.451E-2	3.505E-2	1.27	5.340E-4	1.648E-4	3.24
10	500	5.548E-2	4.783E-2	1.16	5.748E-4	2.475E-4	2.32

$\overline{H}$		Mean			Variance		
m	n	Emp	Pred	Emp/Pred	Emp	Pred	Emp/Pred
1	1	3.241E-6	3.462E-6	.936	1.875E-8	2.251E-8	.833
1	100	3.068E-4	3.462E-4	.886	1.974E-6	2.251E-6	.877
1	500	1.634E-3	1.731E-3	.944	1.1163E-5	1.126E-5	.992
5	5	8.913E-5	8.656E-5	1.03	6.4808E-7	5.616E-7	1.15
10	10	3.495E-4	3.462E-4	1.01	2.400E-6	2.240E-6	1.07
10	100	3.508E-3	3.462E-3	1.01	2.328E-5	2.240E-5	1.04
10	500	1.629E-2	1.731E-2	.941	1.077E-4	1.120E-4	.961

Table 3.1: A table of predicted versus empirical means and variances of the distribution $f_{W_H}(w)$, in the top table, and $f_{W_{\overline{H}}}(w)$ in the bottom table, for different values of m and n.

and therefore the weight that an $n+3$-size random image contributes to an incorrectly hypothesized model of size $m+3$ follows the distribution:

$$N(n\mathrm{E}[V_{\overline{M}}]\,, n\mathrm{Var}\,(V_{\overline{M}}))$$

The means for both distributions were tested empirically from the same experiment as shown in Figures 3-5. That is, for W_H, we generated a random model of size $m+3$ and projected it into an image, adding a Gaussian displacement error to each point, and adding $n-m$ additional clutter points (distributed uniformly within the image). We only tested correct hypotheses, and kept track of the accumulated weight. We repeated this experiment for a given (m, n) pair until we had over a few thousand points. The same was done for $W_{\overline{H}}$ except that the image tested contained n random points (i.e., the model was not projected into the image) implying that only incorrect hypotheses were tested. A table of values for the means and variances of all the experiments is given in Table 3.1. For all the experiments, occlusion $= 0, \sigma_0 = 2.5$.

In all the experiments, the means are close those predicted for the experiments. For $W_{\overline{H}}$, the predicted variance is also quite accurate. The underestimate of the variance for W_H is due to the fact that our assumption that the true points contribute weight independently of any other true point is false, and in fact the average covariance between pairs of projected error distributions is positive. This can be seen also in

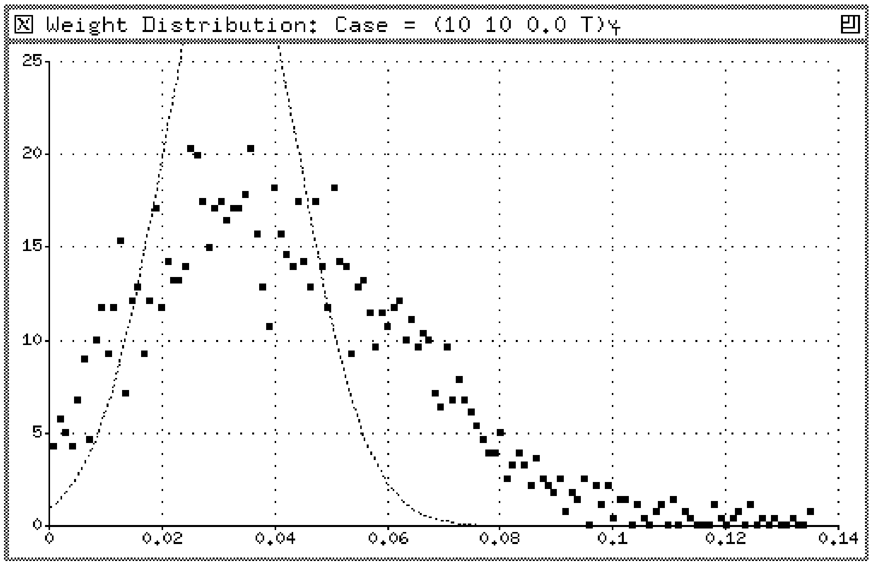


Figure 3-6: Comparison of predicted to empirical density of W_H, for m = 10, n = 10, c = 0, and σ_0 = 2.5. Note that the empirical density has much greater variance than predicted.

Figure 3-6, which shows the empirical versus analytical density of W_H for m, n, c, σ_0 = 10, 10, 0.0, 2.5.

Chapter Summary

In the analysis, we limited the model domain to planar objects. The reason for this was that the analytic expression for the projected error of the fourth model point, which for planar objects is given by Equation 3.4 and 3.7, is not known for either the uniform or Gaussian error model when the model is not planar. This is the only factor limiting the applicability of the analysis to $3D$ models; when such an expression becomes available, this method will easily be able to incorporate it.

In the beginning of the chapter we asked the questions, how do we accumulate evidence for a hypothesis, how do we decide if the hypothesis is correct, and how likely are we to have made a mistake in the decision? So far we have addressed the first question by selecting the recognition algorithm and noise model that we are using, and deriving the probability density functions associated with the scores that correct and incorrect hypotheses will accumulate using our algorithm. In the next chapter we will discuss how to decide whether a hypothesis is correct or not, given its score.

Chapter 4

Distinguishing Correct Hypotheses

In the last chapter we selected an algorithm, noise model and weighting scheme, and using these we derived expressions for f_{W_H} and $f_{W_{\overline{H}}}$, the weight densities of correct and incorrect hypotheses. What we need now is a way to decide, given a score for a hypothesis, if that score is high enough to warrant our deciding that the hypothesis is correct. In this chapter we will show how to use the probability densities derived in the previous chapter to do this. We briefly introduce the ROC (receiver operating characteristic) curve, a concept borrowed from standard hypothesis testing theory, and cast our problem in terms of this framework.

4.1 ROC: Introduction

Suppose we have the following problem: we are observing a world in which there are exactly two mutually exclusive and exhaustive events: H_0 and H_1. We are given the task of deciding which one of them is correct. The only hint we have is some quantity X that we can observe. We also know that if H_0 were true, then we would observe the value of X distributed in some known way, and similarly if H_1 were true, *i.e.,* we know $f_X(x \mid H_0)$ and $f_X(x \mid H_1)$. To relate this back to the object recognition problem, H_0 = "hypothesis being tested is incorrect" and H_1 = "hypothesis being tested is correct".

Let the space of all possible values of the random variable X be divided into two regions, Z_0 and Z_1, such that we decide H_0 if the value of X falls in Z_0, and H_1 if X falls in Z_1. Conversely, we can think of the decision procedure as defining the decision regions Z_0 and Z_1. Then we can define the quantities

$$\mathrm{P}\{\text{say } H_0 \mid H_0 \text{ is true}\} = \int_{Z_0} f_X(x \mid H_0)dx$$

$$\begin{aligned}
P_F &= \mathrm{P}\{\text{say } H_1 \mid H_0 \text{ is true}\} = \int_{Z_1} f_X(x \mid H_0)dx \\
P_M &= \mathrm{P}\{\text{say } H_0 \mid H_1 \text{ is true}\} = \int_{Z_0} f_X(x \mid H_1)dx \\
P_D &= \mathrm{P}\{\text{say } H_1 \mid H_1 \text{ is true}\} = \int_{Z_1} f_X(x \mid H_1)dx
\end{aligned}$$

These quantities are often referred to as P_M = "Probability of a miss", P_D = "Probability of detection", and P_F = "Probability of false alarm" for historical reasons.

In our problem we are assuming we have no prior knowledge of the probabilities of H_0 or H_1. In the absence of such information, a Neyman Pearson criterion, which maximizes P_D for a given P_F, is considered optimal [VT68]. This criterion uses a likelihood ratio test (LRT) to divide the observation space into decision regions, i.e.,

$$\frac{f_X(x \mid H_1)}{f_X(x \mid H_0)} \underset{\text{say } H_0}{\overset{\text{say } H_1}{\gtrless}} \eta$$

That is, we observe a particular value x, and compare the conditional probability density functions for that value of x. If the ratio of the conditional densities is greater than a fixed threshold η, choose H_1, otherwise choose H_0. Changing the value of the threshold η changes the decision regions and thus the values of P_F and P_D. The ROC (receiver-operating characteristic) curve is simply the graph of P_D versus P_F as a function of varying the threshold for the LRT. The optimal performance achievable is given by points on the curve.

If the prior probabilities of H_0 and H_1 are known, then the optimal Bayes decision rule is used. This test also involves a likelihood ratio test, in which the threshold η chosen to minimize the expected cost of the decision, and is a function of the costs and priors involved:

$$\eta = \frac{(C_{10} - C_{00})P_0}{(C_{01} - C_{11})P_1}$$

where C_{ij} is the cost associated with choosing hypothesis i given that hypothesis j is correct, P_i is the a-priori probability that hypothesis H_i is correct, and $C_{10} > C_{00}$ and $C_{01} > C_{11}$ have been assumed. This point necessarily lies on the ROC curve, thus the ROC curve encapsulates all information needed for either the Neyman Pearson or Bayes criterion.

For example, assume for our problem that $H_0 \sim N(m_0, \sigma_0^2)$ and $H_1 \sim N(m_1, \sigma_1^2)$, and assume that $m_1 > m_0$ and $\sigma_1 > \sigma_0$. The likelihood ratio test yields:

$$\frac{f_X(x \mid H_1)}{f_X(x \mid H_0)} \overset{\text{say } H_1}{\underset{\text{say } H_0}{\gtrless}} \eta$$

$$\frac{\frac{1}{\sqrt{2\pi}\sigma_1}\exp(-\frac{(x-m_1)^2}{2\sigma_1^2})}{\frac{1}{\sqrt{2\pi}\sigma_0}\exp(-\frac{(x-m_0)^2}{2\sigma_0^2})} \overset{\text{say } H_1}{\underset{\text{say } H_0}{\gtrless}} \eta$$

$$\exp\left(\frac{(x-m_0)^2}{2\sigma_0^2} - \frac{(x-m_1)^2}{2\sigma_1^2}\right) \overset{\text{say } H_1}{\underset{\text{say } H_0}{\gtrless}} \frac{\eta\sigma_1}{\sigma_0}$$

$$\left(\frac{x-m_0}{\sigma_0}\right)^2 - \left(\frac{x-m_1}{\sigma_1}\right)^2 \overset{\text{say } H_1}{\underset{\text{say } H_0}{\gtrless}} 2\ln\frac{\eta\sigma_1}{\sigma_0} = \gamma$$

The regions Z_0 and Z_1 are found by solving the above equation for equality,

$$x_1 = \frac{[(m_0\sigma_1^2 - m_1\sigma_0^2) - \sigma_0\sigma_1(\gamma[\sigma_1^2 - \sigma_0^2] + (m_0 - m_1)^2)^{1/2}]}{\sigma_1^2 - \sigma_0^2}$$
$$x_2 = \frac{[(m_0\sigma_1^2 - m_1\sigma_0^2) + \sigma_0\sigma_1(\gamma[\sigma_1^2 - \sigma_0^2] + (m_0 - m_1)^2)^{1/2}]}{\sigma_1^2 - \sigma_0^2}$$

The values of P_F and P_D are found by integrating the conditional probability densities $f_X(x \mid H_0)$ and $f_X(x \mid H_1)$ over these regions Z_0 and Z_1, where $Z_0 \equiv \{x : x_1 < x < x_2\}$ and $Z_1 \equiv \overline{Z_0}$.

$$P_F = \int_{Z_1} f_X(x \mid H_0)dx \;=\; 1 - \int_{x_1}^{x_2} \frac{1}{\sqrt{2\pi}\sigma_0} e^{-\frac{(x-m_0)^2}{2\sigma_0^2}}$$
$$P_D = \int_{Z_1} f_X(x \mid H_1)dx \;=\; 1 - \int_{x_1}^{x_2} \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x-m_1)^2}{2\sigma_1^2}}$$

In Figure 4-1 for example, we have plotted the ROC curve for the distributions $f_X(x \mid H_0)$ and $f_X(x \mid H_1)$ alongside. The axes are $x = P_F$, $y = P_D$. The line $x = y$ is a lower bound, since for a point on this line, any decision is as likely to be true as false, so the observed value of X gives us no information. Though an ROC curve is a 3D entity (i.e., a point in (P_F, P_D, η) space), we display its projection onto the

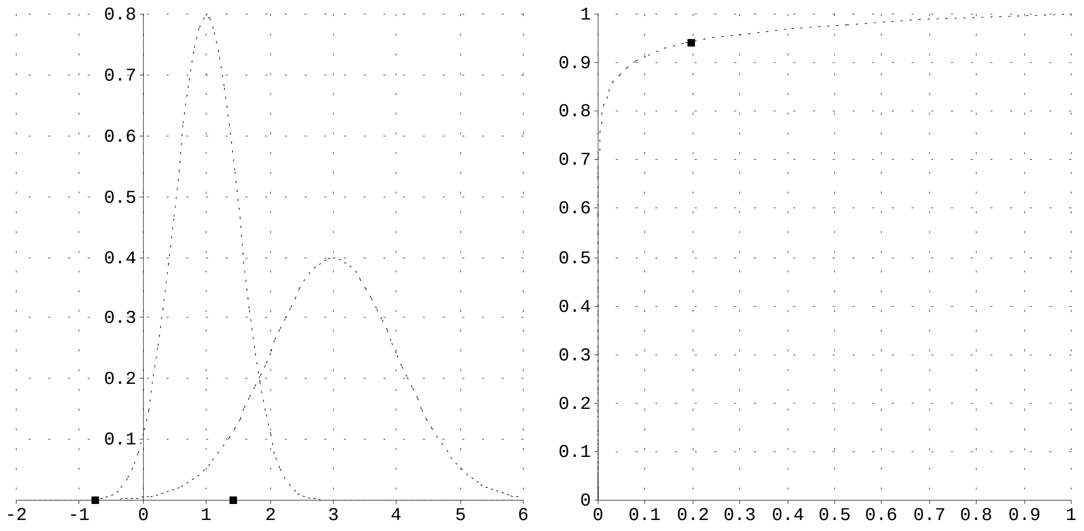


Figure 4-1: On the left is displayed the conditional probability density functions $f_X(x \mid H_0) \sim N(1, .25)$ and $f_X(x \mid H_1) \sim N(3, 1)$ of a random variable X. On the right is the associated ROC curve, where P_F and P_D correspond to the x and y axes, respectively. On the left graph, the boundaries $x_1 = -0.76$ and $x_2 = 1.42$ implied by the value $\gamma = -1.76$ are indicated by boxes. On the right, the ROC point for this γ value is shown. The P_F and P_D values are obtained by integrating the area under the curves $f_X(x \mid H_0)$ and $f_X(x \mid H_1)$ respectively, outside the boundaries. The integration yields the ROC point $(0.2, 0.94)$.

$\eta = 0$ plane and can easily find the associated η value for any (P_F, P_D) pair. When the threshold is infinite there is a 0 probability of false negative, but a 0 probability of correct identification as well. As the threshold goes down, the probabilities of both occurences go up until the threshold is 0, when both positive and false identification are certain. In our problem we assume that we do not have priors, so our goal is to pick a threshold such that we have a very high probability of identification and a low probability of false positives, *i.e.*, we are interested in picking a point as close to the upper left hand side as possible. Note that the larger the separation between the two hypothesis distributions, the more the curve is pushed towards that direction.

4.2 Applying the ROC to Object Recognition

In our problem formulation, H_0 = probability that the hypothesis is not correct, and H_1 = probability that it is. In our case, we have a different ROC curve associated with every fixed (m, n) pair, where $m + 3$ and $n + 3$ are the number of model and image features, respectively. We assume that H_0 and H_1 have Gaussian densities $f_{W_{\overline{H}}}$ and f_{W_H}, whose means and variances were derived in Chapter 3. Because in our formulation the variance of f_{W_H} is always greater than that of $f_{W_{\overline{H}}}$, the lower bound of the interval defining Z_0 is always negative. Since we can't in practice achieve any score lower than 0, we will treat the test as a threshold test, that is, we will accept a hypothesis as being correct if it falls above $\theta = x_2$.

Using this technique, we can predict thresholds for simulated experiments, as shown in the next section.

4.3 Experiment

The predictions of the previous section were tested in the following experiment: to test an ROC curve for model size $m+3$, image size $n+3$, occlusion c and sensor error σ_0, we run two sets of trials, one to test the probability of detection and one to test the probability of false alarm. In all our experiments we used a value $\sigma_0 = 2.5$. For P_D, a random model of size $m + 3$ consisting of point features was generated and projected into an image, with Gaussian noise added to the x and y positional components of each point feature, independently. Occlusion (c) is simulated by adding a c probability of not appearing in the resulting image for each projected model point. Only correct correspondences are tested, and the weight of each of these correct hypotheses is found using the algorithm, restated here:

(a) for a hypothesis $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2) : (\mathrm{s}_0, \mathrm{s}_1, \mathrm{s}_2)$, find all Gaussian weight disk locations and sizes:

   (i) find affine coordinates $\mathbf{m}_j = (\alpha_j, \beta_j)$ with respect to basis $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2)$

   (ii) projected image location for $\mathbf{m}_j$ is $\mathrm{s}_0 + \alpha_j(\mathrm{s}_1 - \mathrm{s}_0) + \beta_j(\mathrm{s}_2 - \mathrm{s}_0)$

(iii) projected Gaussian weight disk radius for $\mathbf{m}_j$ is $2\sigma = 2(\alpha_j^2 + \beta_j^2 + [1 - \alpha_j - \beta_j]^2)$

(b) for every image point $\mathbf{s}_j$, initially set $d = \infty$.

(i) find the minimum distance d between $\mathbf{s}_j$ and $\mathbf{m}_j$ such that $d \leq 2\sigma$.

(ii) add $v = \frac{1}{2\pi\sigma^2} e^{-\frac{d^2}{2\sigma^2}}$ (the height of the Gaussian weight disk at the image point location) to the sum w, which is the total score for this hypothesis. If this image point did not come within 2σ of any projected model point, then $v = 0$.

We performed this experiment, keeping a histogram of the weights, until there were 2500 sample points. To test the probability of false alarm, we run the same experiment using random images which do not contain the model we are looking for. The resultant histograms are normalized to yield the empirical density of W_H and $W_{\overline{H}}$ for the given values of m, n, c and σ_0. To construct the ROC curves we loop through 25 thresholds and tally the proportion of the empirical distributions of W_H and $W_{\overline{H}}$ that fall above the threshold, yielding a (P_F, P_D) pair for each one. The resulting P_D, P_F, and ROC curves as a function of threshold θ are shown in Figure 4-2 for $n = 10, 100, 500, 500$, occlusion $c = 0.0, 0.0, 0.0, 0.25$. The ROC curves for the same parameters are shown alongside. The axes for the graphs are $(x, y) = (\theta, P_F)$, $(x, y) = (\theta, P_D)$, and $(x, y) = (P_F, P_D)$.

The graphs of the P_F, P_D and ROC curves indicate that the predicted and actual curves match very well, with the best predictions when the number of clutter points is high. Turning to the P_D plots, we see that when the threshold is high we consistently underpredict the probability of detection. This error works in our favor, since it pushes the actual (P_F, P_D) points up above the predicted ones. This high threshold area corresponds to the region on the ROC curve along the P_D axis.

The discrepancies between the curves are due to assumptions we made in the analytic derivations, the most significant of which is the assumption that W_H and $W_{\overline{H}}$ are Gaussian. In fact, none of the displayed empirical curves are actually Gaussian, though when the clutter is high the distributions are more nearly so. In theory we could use Chernoff bounds to bound the expressions for P_D and P_F for a given threshold [VT68] but we will not explore this option. Instead, we will use the analytical curves as an approximation to the actual curves, and note that despite this modelling error, we still see a good fit between empirical and actual performance.

4.3.1 Using Model-Specific ROC Curves

The largest discrepancy between predicted and actual performance can be traced to P_D prediction, as seen in Figure 4-2. In our analysis, we assume two things that cause this mismatch. First, we assume that a correctly hypothesized model point accumulates weight in favor of a correct hypothesis independently of any other. Second, we

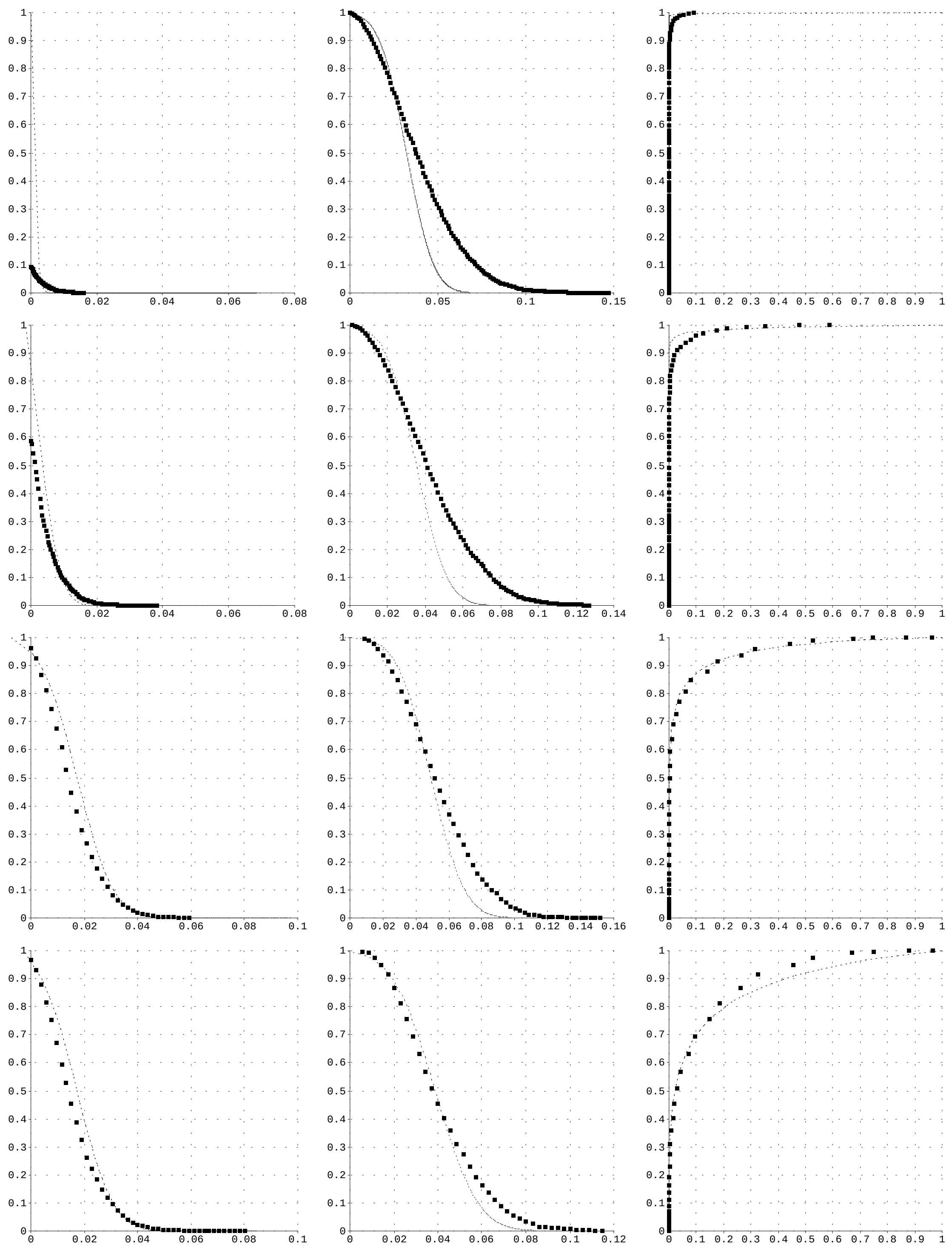


Figure 4-2: Comparison of predicted to empirical curves for probability of false alarm, probability of detection, ROC curves. The empirical curves are indicated by boxes. The axes for graphs, from left to right, are $(x, y) = (\theta, P_F)$, (θ, P_D), and (P_F, P_D). For all graphs, $m = 10$, $\sigma_0 = 2.5$. From top to bottom, $n = 10, 100, 500, 500$, occlusion $= 0, 0, 0, 0.25$.

assume that the number of model points is large enough so that we can approximate the probability density of W_H by a Gaussian. However, we have shown in the previous section how to empirically derive the actual W_H and P_D curves by simulation. We can use this technique to tailor the overall method to a particular model in order to improve the prediction for that model.

Specifically, the method would work as follows: the same simulation as was performed in the previous section is done, using the given model instead of a randomly generated one, with no occlusion or clutter. A simple function can be fit to the actual distribution for W_H, and this function will subsequently be used as the density of $W_{\overline{H}}$ (with not clutter or occlusion) for this model. The density of $W_{\overline{H}}$ for any other value of n and c can then easily be derived from this.

Summary

In this chapter we introduced the ROC curve, which enables us to encapsulate all the information needed to make a decision about choosing thresholds to determine performance. That is, for a particular image, model, and threshold for the weight that a hypothesized match must score in order to accept it, we can predict the probability that a correct or incorrect match will pass the threshold. Conversely, for a given model and image, we can predict the threshold required to achieve a given probability of true detection or false alarm. We applied this technique to simulated models and images and were able to successfully predict thresholds and performance for a wide range of model to image sizes.

The ROC curve also indicates the level of performance achievable for a particular model and image, so that we can determine when a desired level of performance (for instance, 0 probability of false alarm at the same time as a 1.0 probability of a true detection) is simply not possible for a given model and image. In effect, we are able to identify when an image is simply too noisy to be able to achieve any better performance than randomly guessing whether a given hypothesis is correct or not.

Chapter 5

Comparison of Weighting Schemes

In the previous chapters we talked about decision making for a particular weighting scheme; however, we can use the machinery we developed in the last chapter not only to evaluate hypotheses, but also to compare the relative merits of different possible weighting schemes. In this chapter we use the ROC to compare several such schemes. We have already discussed one weighting scheme which we will call Scheme 1. Scheme 2 will denote the weighting and decision scheme generally used with the uniform bounded error model, and Scheme 3 will denote the same weight disk as Scheme 1, but using a weight accumulation algorithm which collects evidence from at most one point per projected error disk. Ultimately we will decide to remain with the original scheme we developed in Chapter 3.

5.1 Uniform Weighting Scheme

The weight disk used with the uniform bounded error model assigns a full vote to any model point which falls inside it. For a model point with coordinates (α, β) in the frame established by the model basis used in the correspondence, the projected weight disk has radius

$$\epsilon_0(\mid 1 - \alpha - \beta \mid + \mid \alpha \mid + \mid \beta \mid + 1).$$

where ϵ_0 is the radius of the uniform error distribution for sensor noise. We will use the symbol ϵ_e to describe the values this expression takes on as the affine coordinates vary. The expected value of V_M under this scheme is $1 - c$, and for $V_{\overline{M}}$, $\mathrm{E}[V_{\overline{M}} \mid m = 1]$ is the probability that a random point will contribute a vote of 1 at a particular weight disk to an incorrect hypothesis. This is the expected size of the weight disk over the size of the image, which is $= \frac{\pi \mathrm{E}[\epsilon_e^2]}{A}$. This last expression was called the *redundancy factor* μ and was derived analytically in [GHJ91], but for our comparison we took the empirical value from simulations such as those described in Section 3.6.1. For an

$\epsilon : \sqrt{A}$ ratio of $1:100$, $\mu \approx 0.0034$. The expected value of $V_{\overline{M}}$ is the probability that a single random point dropping into an image with m circles of average area μ will fall into one or more of them. Using the notation

$$B(p;n,k) = \begin{cases} \binom{n}{k} p^k (1-p)^{n-k} & k \leq n \\ 0 & \text{otherwise} \end{cases}$$

the expression describing the probability that a clutter point will drop into one or more weight disks is given by the inclusion-exclusion principle, and is

$$\sum_{i=1}^{m} (-1)^{i-1} B(\mu; m, i)$$

If we upper bound this probability by assuming the weight disks are disjoint, this is simply $m\mu$. This approximation constrains the number of model points to be less than $m\mu^{-1} \approx 300$.

The random variable $W_{\overline{H}}$ is binomially distributed:

$$F_{W_{\overline{H}}}(k) = B(m\mu; n, k)$$

The distribution for W_H is a little more complicated; that is, in order to observe exactly k points, i points must have been observed from the model, and the remaining $k-i$ points were random, for all numbers from 0 to k:

$$F_{W_H}(k) = \sum_{i=0}^{k} B(1-c; m, i) B(m\mu; n, k-i)$$

This product of two binomial distributions is not itself binomial, and the optimal Neyman Pearson test to distinguish between them is complicated to derive. We will use a simple threshold test since it is widely used, though we have not proven that it is optimal with respect to the Neyman Pearson criterion. The probabilities of a true and false positive using a threshold test are, respectively

$$\begin{aligned} P_D &= 1 - \sum_{i=0}^{k} F_{W_H}(i) \\ P_F &= 1 - \sum_{i=0}^{k} F_{W_{\overline{H}}}(i) \end{aligned}$$

Figure 5-1 compares the ROC curves for Scheme 1 (Gaussian weight disk) and Scheme 2 (uniform weight disk) for $m = 10$,$n = 10, 50, 100, 500, 1000$, occlusion$= 0.0$ and 0.25. We can see that in the case of no occlusion and for small values of n, both techniques predict good P_F vs P_D curves, though the bounded uniform weight disk has better performance because there is no possibility of a false negative when occlusion$= 0$, while with the Gaussian weight disk there always is. However, as n increases, the performance of Scheme 2 breaks down more rapidly than Scheme 1 for both occlusion values. For occlusion$= 0.25$, both schemes perform about equally for small values of n (for example, at $n = 100$), but again as n increases, the performance of Scheme 2 degrades more dramatically than that of Scheme 1 ($n \geq 500$).

5.2 An Alternative Accumulation Procedure

In Section 3.8 we presented the basic recognition algorithm, and pointed out that in the accumulation procedure, we add the contributions from every *image point*, as opposed to at most one point per error disk. That is, if several image points fall within the same error disk, we add the contributions from all of them. Intuitively one would expect that this would not work as well as simply taking the value of the closest point to the center of each error disk.

In this section we investigate the what happens to the ROC curve if we modify the accumulation step to take only the "heaviest" point per error disk, *i.e.*, the one appearing closest to the disk center. This weight scheme will be called Scheme 3, and we will use the same variable names as we did for Scheme 1, but with a '*' in the name to differentiate the random variables and their distributions from those of Scheme 1. The derivations of the density functions for Scheme 3 are more difficult, and we will end up approximating the density function $f_{V^*_{\overline{M}}}$ such that $\mathrm{E}\left[V^*_{\overline{M}}\right]$ is underestimated. Surprisingly, we will see that even with this underestimate, the theoretical ROC curve for Scheme 3 is not as good as for Scheme 1.

We begin by defining two new random variables, V^*_M and $V^*_{\overline{M}}$. The difference between V_M and V^*_M is that the former variable described the weight that a true *image point* would yield — that is, a point which actually arises from the model when a correct correspondence between model and image points has been established, and the rest of the model points are projected into the image. V^*_M is the weight that a true *weight disk* will contribute to the accumulated sum — that is, a disk which is projected into the image when a correct hypothesis is being tested, when the image contains $n + 3$ points. The same distinction holds for the variables $V_{\overline{M}}$ and $V^*_{\overline{M}}$.

We begin with the random variable $V^*_{\overline{M}}$. Extending a derivation given in [BRB89] for the one dimensional case, we first define yet another random variable, $X_k =$ distance of the closest point to the center of a disk, when the disk contains k uniformly distributed points. We derive the probability density function as follows: Let a be the radius of the disk. We divide the disk into an inner disk and 2 rings:

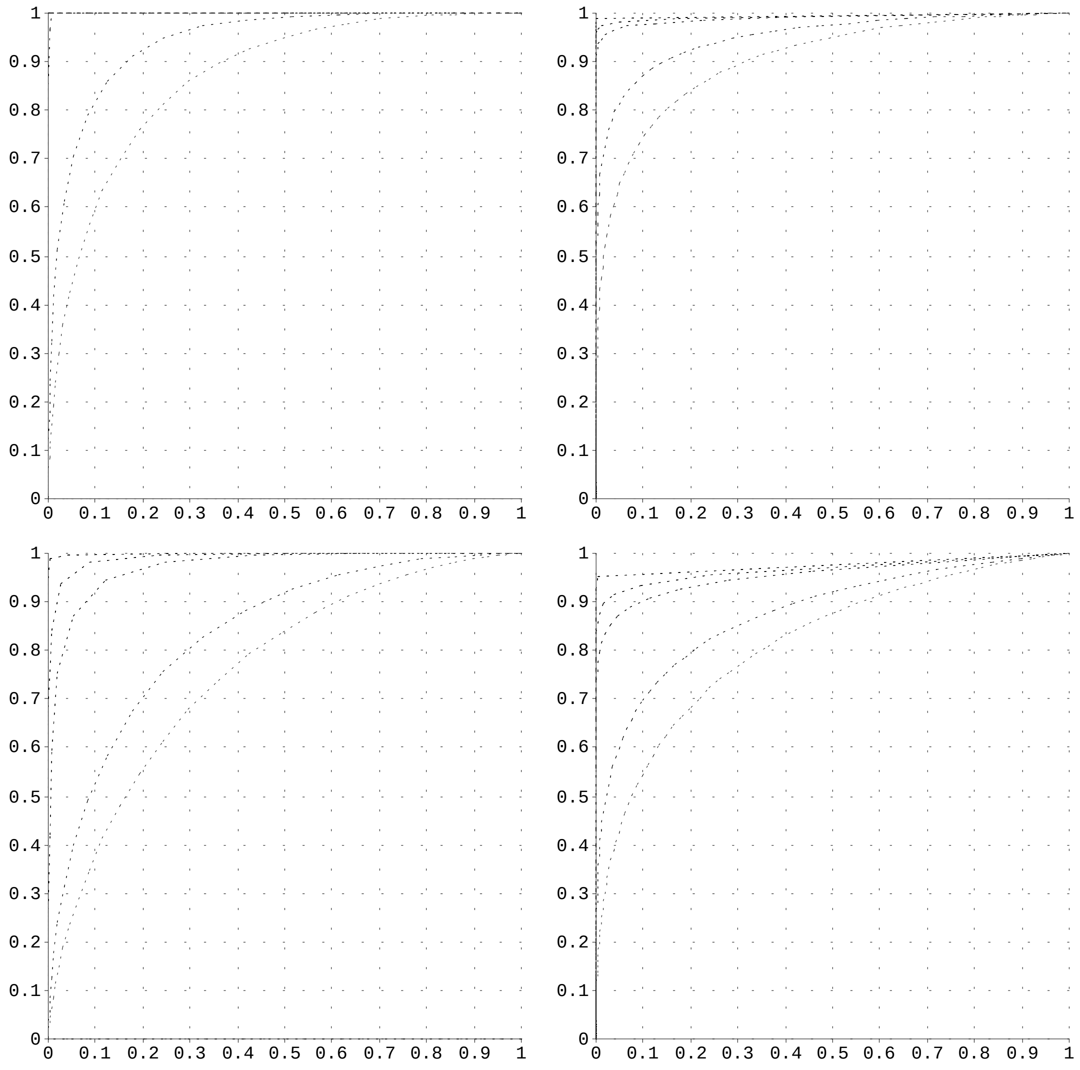
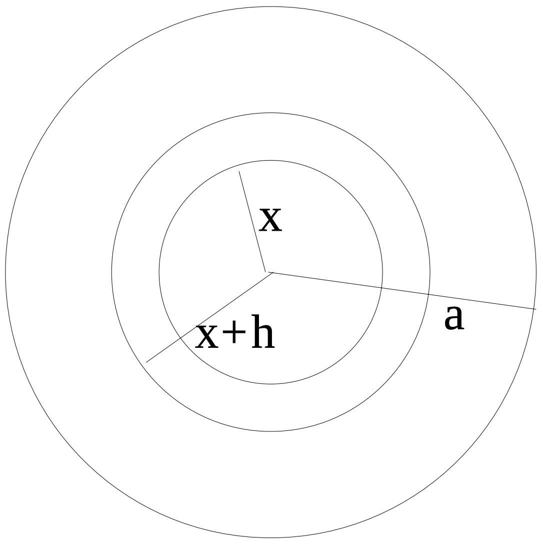


Figure 5-1: Comparisons of uniform and Gaussian weight disks for $m = 10$, $n = 10$, 50, 100, 500, 1000. Left: uniform weight disk, right: Gaussian weight disk. Top: Occlusion=0, bottom: occlusion=0.25. For all ROC curves, the x and y axes are P_F and P_D, respectively. A low threshold results in an ROC point on the upper right corner. As the threshold increases, the performance (P_F, P_D) moves along the curve toward the lower left corner.



To find the probability density of X_k, we first find the probability that of k points, none fall in the inner disk, one falls inside the h-ring and the remaining $k-1$ fall outside the ring:

$$\begin{aligned}
\mathrm{P}\{x \leq X_k \leq x+h\} &= \binom{k}{1} \left(\frac{(x+h)^2 - x^2}{a^2}\right)^1 \left(\frac{a^2 - (x+h)^2}{a^2}\right)^{k-1} \\
&= k\frac{(2hx+h^2)(a^2 - x^2 - 2ax - h^2)^{k-1}}{a^{2k}}
\end{aligned}$$

Now we take the above expression, divide by the width of the ring h, and take the limit as $h \rightarrow 0$:

$$\begin{aligned}
fX_k(x) &= \lim_{h\rightarrow 0} \frac{P(x \leq X_k \leq x+h)}{h} \\
&= \lim_{h\rightarrow 0} \frac{k}{a^{2k}}(2x+h)(a^2 - x^2 - 2hx - h^2)^{k-1} \\
&= \frac{k}{a^{2k}} 2x(a^2 - x^2)^{k-1}
\end{aligned}$$

Now, let Y_n = distance of closest point to the center of a given weight disk disk (*i.e.*, σ_e is fixed) in an image with $n+3$ points. So, the radius of the disk, a, is $2\sigma_e$. For legibility, let us set $\mu_g = \frac{4\pi\sigma_e^2}{A}$ to be the probability that a random image point falls in the error disk. Then the probability that the closest image point to the disk center is at a distance x equals the probability that exactly k points fall in the disk times the probability that the closest of the k points is x away from the center of the disk, for all k:

$$f_{Y_n|\sigma_e}(x \mid \sigma) = \sum_{k=1}^{n} \mathrm{P}\{k \text{ pts fall in disk}\}\, \mathrm{P}\{n-k \text{ pts fall outside}\}\, f_{X_k|\sigma_e}(x \mid \sigma)$$

$$= \sum_{k=1}^{n} \binom{n}{k} (\mu_g)^k (1-\mu_g)^{n-k} \left[\frac{k}{(4\sigma^2)^k} 2x(4\sigma^2 - x^2)^{k-1} \right]$$

To derive $f_{V^*_{\overline{M}}|\sigma_e}(v \mid \sigma)$, we have to determine the density of $g(Y_n)$ for a fixed σ_e, where $g(y) = \frac{1}{2\pi\sigma_e^2} e^{-\frac{y^2}{2\sigma_e^2}}$. This is extraordinarily complicated, and we will not even attempt it.

Instead, we do the following. First let us assume that $n < \frac{1}{\mu_g}$. We will justify this assumption shortly. This together with the fact that $(1-\mu_g) \approx 1$ means that not only is the binomial term decreasing after the first term, but that the second term is less than half the first. Therefore, we take the liberty of approximating the entire distribution by the first term. When we do this, the term $f_{X_k}(x)$ becomes much simpler, since we only have to worry about the case where a single random image point falls in the error disk, $k = 1$:

$$f_{X_1|\sigma_e}(x \mid \sigma) = \frac{x}{2\sigma^2}$$

Not surprisingly, this is the same expression as we derived back in Chapter 3 for the distance of a single point from the center of a disk, when the point is drawn from a uniform distribution. At that time we derived the weight that such a point will contribute when using our weighting scheme g:

$$f_{g(x)|\sigma_e}(v \mid \sigma) = \frac{1}{2} v^{-1}$$

So, combining this expression with the probability that a single point will fall in a given error disk, we get

$$\begin{aligned} f_{Y_n|\sigma_e}(v \mid \sigma) &= n\mu_g(1-\mu_g)^{n-1} f_{g(x)|\sigma_e}(v \mid \sigma) \\ &\approx n\frac{4\pi\sigma_e^2}{A}\left[\frac{1}{2}v^{-1}\right] \end{aligned}$$

which is exactly n times the distribution $f_{V_{\overline{M}}}(v \mid m = 1)$ that we derived back in Chapter 3. Without rehashing all the steps, we simply point out a few differences. In particular, in Equation 3.15, m was bounded above by $\frac{A}{\sigma_0^2} * (4\pi a_0[\frac{1}{r_1} - \frac{1}{r_2}])^{-1}$ in order for the distribution $f_{V_{\overline{M}}}(v)$ to be a density function. For the distribution $f_{V^*_{\overline{M}}}(v)$ the same bound must hold, but for n instead of m. Let us call N the maximum number of allowable image points. Now we can justify our first assumption that $n < \frac{1}{\mu_g}$: Let us use the expected area of the Gaussian error disk over all values of σ_e. This is given by the expression:

$$\begin{aligned} &\int_{s_1}^{s_2} \pi(2\sigma)^2 f_{\sigma_e|\overline{H}}(\sigma)d\sigma \\ = &\int_{s_1}^{s_2} \pi(2\sigma)^2 \frac{b_0}{\sigma^4} d\sigma \end{aligned}$$

$$\begin{aligned}
&= 4\pi b_0 \int_{s_1}^{s_2} \frac{1}{\sigma^2} d\sigma \\
&= 4\pi b_0 \left[\frac{1}{s_1} - \frac{1}{s_2}\right] \\
&= 4\pi a_0 \sigma_0^2 \left[\frac{1}{r_1} - \frac{1}{r_2}\right]
\end{aligned}$$

The maximum number of disks that can fit into the image (assuming the disks are disjoint) is the image area A divided by this expression, which is exactly the bound $\frac{1}{\mu_g}$ that we assumed above.

To sum up, we have derived the approximation

$$f_{V^*_{\overline{M}}}(v) \approx \min(n, N) f_{V_{\overline{M}}}(v \mid m = 1)$$

and therefore

$$\begin{aligned}
\mathrm{E}\left[V^*_{\overline{M}}\right] &\approx \min(n, N)\mathrm{E}[V_{\overline{M}} \mid m = 1] \\
\mathrm{E}\left[(V^*_{\overline{M}})^2\right] &\approx \min(n, N)\mathrm{E}\left[V^2_{\overline{M}} \mid m = 1\right] \\
\mathrm{Var}\left(V^*_{\overline{M}}\right) &\approx \min(n, N)\mathrm{E}\left[V^2_{\overline{M}} \mid m = 1\right] - \min(n, N)^2\mathrm{E}[V_{\overline{M}} \mid m = 1]
\end{aligned}$$

All the terms on the right hand side of these equations are known quantities that were derived back in Chapter 3. Note that because of our approximations, our prediction for $\mathrm{E}\left[V^*_{\overline{M}}\right]$ is an underestimate of the distribution's actual first moment.

Finally we derive $f_{V^*_M}(v)$. We first look at a single correctly hypothesized weight disk — that is, the weight that is scored by a disk which is projected into the image as a result of testing a correct hypothesis. The weight disk always contains the correctly projected model point, unless (a) the point is occluded, or (b) the point falls outside the $2\sigma_e$ Gaussian weight disk. If either of these two things happen, then all of the clutter points get a chance to score inside the weight disk. We will also assume for simplicity that if the true point appears inside the disk then we will take its contribution even if clutter points also appear inside the disk. Then the probability that we will see weight $v > 0$ is:

$$\begin{aligned}
&\mathrm{P}\{\text{disk gets weight } v > 0\} = \\
&\quad \mathrm{P}\{\text{disk gets weight } v > 0 \mid \text{true point seen}\} + \\
&\quad \mathrm{P}\{\text{true point not seen}\}\, \mathrm{P}\{\text{disk gets weight } v > 0 \mid \text{false point seen}\}
\end{aligned}$$

And for the case when $v = 0$:

$$\mathrm{P}\{\text{disk gets weight } v = 0\} = \mathrm{P}\{\text{true point not seen}\}\, \mathrm{P}\{\text{false point not seen}\}$$

For convenience let us call B the probability that no clutter point falls inside a true weight disk. In the case in which occlusion$=0$ the expression for the density $f_{V^*_M}$

would be correctly given by the expression

$$f_{V_M^*}(v) \quad = \quad \begin{cases} [c+(1-c)e^{-2}]B\delta(v) & v=0 \\ f_{V_M}(v)+[c+(1-c)e^{-2}]f_{V^*_{\overline{M}}}(v) & v\neq 0 \end{cases}$$

in which the c's would disappear. When occlusion $\neq 0$ we have the problem that we don't know how many of the observed image points are clutter points. Therefore, we must define a random variable M describing the number of model points that actually show up in the image. M is binomially distributed with mean $(1-c)m$ and variance $c(1-c)m$. Using this random variable instead of m in the expression for $f_{V^*_{\overline{M}}}$ in the above expression, the density $f_{V^*_M}$ becomes:

$$f_{V_M^*}(v) \quad = \quad \begin{cases} [c+(1-c)e^{-2}]B\delta(v) & v=0 \\ f_{V_M}(v)+[c+(1-c)e^{-2}]\min(n-M,N)f_{V_{\overline{M}}}(v \mid m=1) & v\neq 0 \end{cases}$$

Let us temporarily rename $p=c+(1-c)e^{-2}$, and assume that $\min(n-M,N)=n-M$ for ease of manipulation. Then

$$f_{V_M^*}(v) \quad = \quad \begin{cases} pe^{-2}B\delta(v) & v=0 \\ f_{V_M}(v)+p(n-M)f_{V_{\overline{M}}}(v \mid m=1) & v\neq 0 \end{cases}$$

We use Equations A.2 and A.3 to remove the random variable M, first for the mean:

$$\mathrm{E}[V_M^*] \quad = \quad \mathrm{E}[V_M]+p(n-(1-c)m)\mathrm{E}[V_{\overline{M}} \mid m=1] \tag{5.1}$$

and proceeding in stages for the variance:

$$\begin{aligned}
\mathrm{E}[V_M^* \mid M] &= \mathrm{E}[V_M]+p(n-M)\mathrm{E}[V_{\overline{M}} \mid m=1] \\
\mathrm{Var}\,(V_M^* \mid M) &= \mathrm{E}\Big[(V_M^*)^2 \mid M\Big]-\mathrm{E}[V_M^* \mid M]^2 \\
&= \Big[\mathrm{E}\Big[V_M^2\Big]+p(n-M)\mathrm{E}\Big[V_{\overline{M}}^2 \mid m=1\Big]\Big]- \\
&\quad [\mathrm{E}[V_M]+p(n-M)\mathrm{E}[V_{\overline{M}} \mid m=1]]^2 \\
\mathrm{Var}\,(\mathrm{E}[V_M^* \mid M]) &= p^2\mathrm{E}[V_{\overline{M}} \mid m=1]^2\,\mathrm{Var}\,(M) \\
\mathrm{E}[\mathrm{Var}\,(V_M^* \mid M)] &= \mathrm{E}\Big[V_M^2\Big]-\mathrm{E}[V_M]^2+p\mathrm{E}[(n-M)]\,\mathrm{E}\Big[V_{\overline{M}}^2 \mid m=1\Big] \\
&\quad -p^2\mathrm{E}\Big[(n-M)^2\Big]\,\mathrm{E}[V_{\overline{M}} \mid m=1]^2 \\
&\quad -2p\mathrm{E}[n-M]\,\mathrm{E}[V_M]\,\mathrm{E}[V_{\overline{M}} \mid m=1]
\end{aligned}$$

Next, substituting the expression $\mathrm{E}[M^2]=\mathrm{Var}\,(M)+\mathrm{E}[M]^2$ into the last equation and solving for the entire expression, we get:

$$\begin{aligned}
\mathrm{Var}\,(V_M^*) &= \mathrm{Var}\,(\mathrm{E}[V_M^* \mid M])+\mathrm{E}[\mathrm{Var}\,(V_M^* \mid M)] \\
&= p^2\mathrm{E}[V_{\overline{M}} \mid m=1]^2\,\mathrm{Var}\,(M)+\mathrm{E}\Big[V_M^2\Big]-\mathrm{E}[V_M]^2 \\
&\quad +p(n-\mathrm{E}[M])\mathrm{E}\Big[V_{\overline{M}}^2 \mid m=1\Big]
\end{aligned}$$

$$
\begin{aligned}
&-p^2[(n - \mathrm{E}[M])^2 + \mathrm{Var}\,(M)]\mathrm{E}[V_{\overline{M}} \mid m = 1]^2 \\
&-2p\mathrm{E}[n - M]\,\mathrm{E}[V_M]\,\mathrm{E}[V_{\overline{M}} \mid m = 1]
\end{aligned}
$$

Note that the first term in the sum cancels out the variance in the third line, leaving the expression:

$$
\begin{aligned}
\mathrm{Var}\,(V_M^*) \;=\;& \mathrm{E}\Big[V_M^2\Big] - \mathrm{E}[V_M]^2 \\
&+p(n - \mathrm{E}[M])\mathrm{E}\Big[V_{\overline{M}}^2 \mid m = 1\Big] - p^2(n - \mathrm{E}[M])^2\mathrm{E}[V_{\overline{M}} \mid m = 1]^2 \\
&-2p(n - \mathrm{E}[M])\mathrm{E}[V_M]\,\mathrm{E}[V_{\overline{M}} \mid m = 1]
\end{aligned}
$$

Putting back the min expression and substituting the value of E[M] we finally get

$$
\begin{aligned}
\mathrm{Var}\,(V_M^*) \;=\;& \mathrm{E}\Big[V_M^2\Big] - \mathrm{E}[V_M]^2 + p\min(n - (1 - c)m, N)\mathrm{E}\Big[V_{\overline{M}}^2 \mid m = 1\Big] \\
&-p^2\min(n - (1 - c)m, N)^2\mathrm{E}[V_{\overline{M}} \mid m = 1]^2 \\
&-2p\min(n - (1 - c)m, N)\mathrm{E}[V_M]\,\mathrm{E}[V_{\overline{M}} \mid m = 1]
\end{aligned}
$$

in which all the terms are known.

The accumulated densities $W_{\overline{H}}^*$ and W_H^* are simply collected over all m error disks independently, so that

$$
\begin{aligned}
W_H^* &\sim N(m\mathrm{E}[V_M^*]\,, m\mathrm{Var}\,(V_M^*)) \\
W_{\overline{H}}^* &\sim N(m\mathrm{E}\Big[V_{\overline{M}}^*\Big]\,, m\mathrm{Var}\,\Big(V_{\overline{M}}^*\Big))
\end{aligned}
$$

In Figure 5-2 we show an ROC comparison of Schemes 1 and 3. The new method, Scheme 3, performs very poorly in theory because as the number of clutter points go up, the chance of at least one point appearing in every disk is very high. When this happens it is no longer possible to distinguish between correct and incorrect hypotheses. For a $\sigma_0 : \sqrt{A}$ ratio of 1:200, the maximum number of image points allowable by the method is 818; at this point a random point will appear in every weight disk with probability 1 and the ROC curve becomes almost diagonal.

In Figure 5-3 we see the actual P_F, P_D and ROC curves using this weighting method. The predicted performance greatly underestimates the actual performance of the method. This discrepancy is due to a simplification which we glossed over in our analysis, which is the assumption that the weight disks are disjoint. When this is not the case, then this assumption overestimates the actual probability of a clutter point landing in a weight disk, thereby overestimating the mean of the random variable $V_{\overline{M}}^*$. The same effect occurs in Scheme 1 as well, though to a lesser extent.

Despite the fact that the actual performance of Scheme 3 is better than predicted, it is still the case that the actual ROC curves for Scheme 3 are not as good as those for Scheme 1, as can be seen in Figure 5-4.

It is clear that Scheme 1 is better than Scheme 3 most importantly because the latter performs better in actual simulations than Scheme 3. It also has the advantage that our predictions for the distributions of W_H and $W_{\overline{H}}$ are more accurate than those of W_H^* and $W_{\overline{H}}^*$. For both these reasons we will perform all experiments in the next chapter using Scheme 1.

Summary of Weighting Schemes

In this chapter we discussed two different possible weighting schemes and used the ROC curves to compare them to the original scheme we developed in Chapter 3. Ultimately we showed that our original scheme has better error performance than both alternative schemes. It is important to note that none of the schemes we have analyzed is optimal with respect to a maximum likelihood criterion, which would assign a better score to hypothesis H_A than to H_B if $\mathrm{P}\{image \mid H_A\} > \mathrm{P}\{image \mid H_B\}$. For the remainder of the thesis we will use the original scheme we developed in Chapter 3.

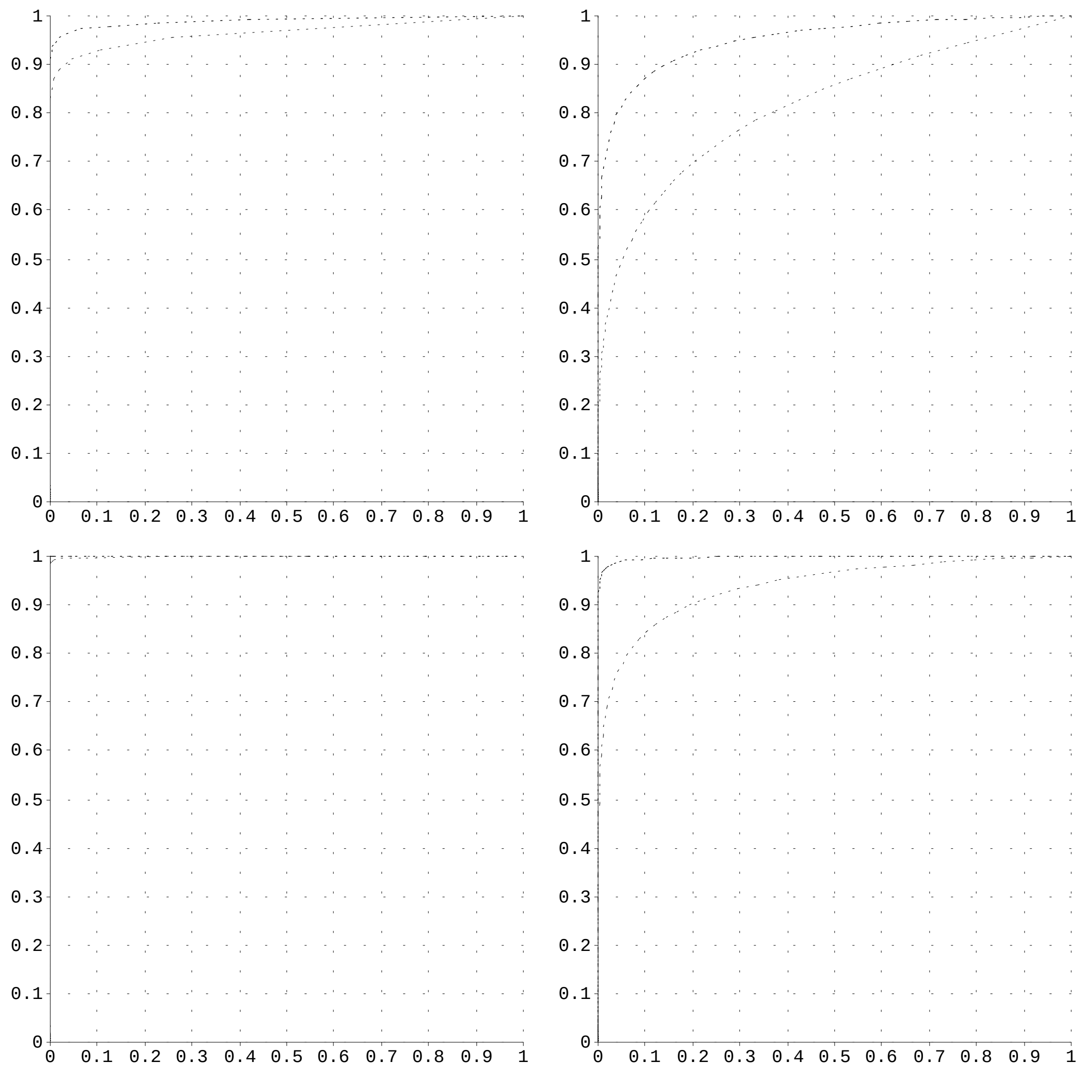


Figure 5-2: The graphs show the comparison of ROC curves for Scheme 1 (top curve) versus Scheme 3 (bottom curve). The x and y axes are P_F and P_D, respectively. Increasing (P_F, P_D) corresponds to a decreasing threshold for the direction of The (m,n) pairs are (10,100), (10,500), (30,100), and (30,500). For all graphs, occlusion = 0 and $\sigma_0 = 2.5$.

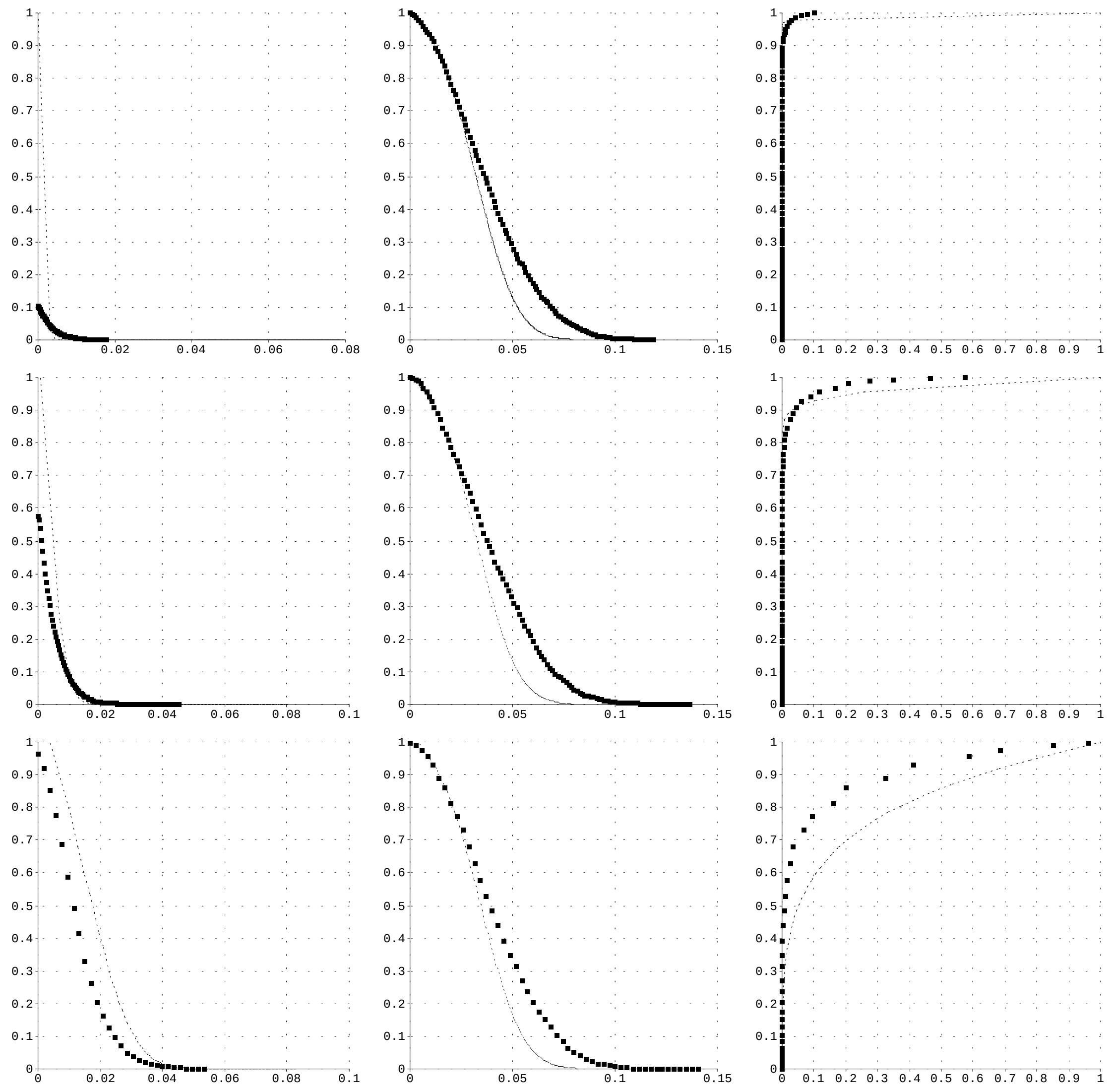


Figure 5-3: Comparison of predicted to empirical curves for probability of false alarm, probability of detection, and ROC curves for Scheme 3. The empirical points are indicated by boxes. The axes for graphs, from left to right, are $(x, y) = (\theta, P_F)$, (θ, P_D), and (P_F, P_D). For all graphs, $m = 10$, occlusion $= 0$, $\sigma_0 = 2.5$. From top to bottom, $n = 10, 100, 500$.

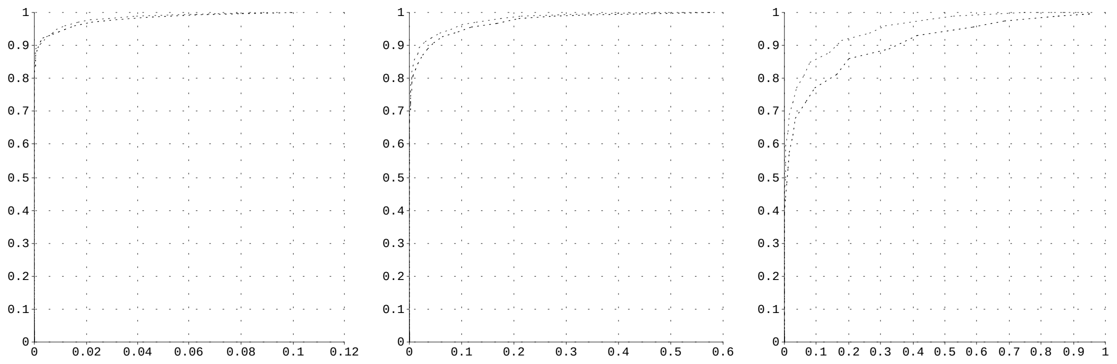


Figure 5-4: Comparison of empirical ROC curves for Schemes 1 and 3. In all graphs the ROC curve for Scheme 1 is above that of Scheme 3. The axes are $(x, y) = (P_F, P_D)$. For all graphs, $m = 10$, occlusion $= 0$, $\sigma_0 = 2.5$. From left to right, $n = 10, 100, 500$.

Chapter 6

A Feasibility Demonstration

In the preceding chapters we presented a theoretical approach to placing a bound on the probability of true versus false detection for the output of a recognition problem in a limited domain, and some simulations supporting the viability of the method. In this chapter we argue that the Gaussian error model is a reasonable approximation for point feature locations by measuring the noise associated with different point feature types. Finally, we demonstrate the process of applying the analysis to real images.

6.1 Measuring Noise

6.1.1 Feature Types

A point feature is a physical aspect of the model which can be detected at a $2D$ location in an image of the model, regardless of the model's pose. When considered in this light, we can see that there are two aspects of how powerful a feature type is as a representation — its ability to represent the model, and its ability to be reliably extracted from the image. Asada and Brady [AB86] discuss a model representation called a Curvature Primal Sketch, in which point features are defined as distinctive points in the curvature of the boundary of the object; *i.e.*, zero crossings, minima/maxima, and discontinuities, in the boundary curve's first derivative. In the domain of planar models, these model features are all invariant to affine transformations and by extension, pose (note that it is the location of the features, and not the magnitude of the boundary curve's first derivative at these points, that is affine invariant). However, reliably extracting these sorts of features from an image is difficult, since their location is extremely dependent on factors such as pixel resolution, image processing parameters, and even model pose, since at certain poses the magnitudes of the boundary curve's first derivative becomes so small that the features become undetectable.

Another possible representation is to limit the feature types to a single kind of curvature discontinuity, that is, intersections of straight line segments greater than some fixed length. Line crossings, junctions, and corners are all examples of this. This representation has its own problems since boundary curves on the model cannot be represented at all by line segments. From the image processing side, a curve appearing in an image gives rise to a indeterminate number of corner features, depending on the magnitude of the curvature and image processing parameters. However, the locations of intersections of long straight line segments might be more stably detected.

Wells [Wel92] uses as point features the center of mass of connected pixel strings of length k, broken randomly. This is similar, but not equivalent, to sampling the contour of the object at fixed intervals. One possible problem with this type of point feature is simply that there are so many — that is, if k is too small, we are not significantly pruning the search in transformation space by using these features to form hypotheses.

Few recognition systems in the literature use the simple point features described above; for example, SCERPO [Low86] and HYPER [AF86] both use entire line segments as features, the Local-Feature-Focus method of Bolles and Cain [BC82] uses oriented corners and holes, Huttenlocher's ORA system [Hut88] uses oriented points, and Ettinger's SAPPHIRE system ([Ett87]) uses the compound point features defined in the Curvature Primal Sketch. The advantage of using more information per feature is that the additional information will often eliminate hypotheses consisting of impossible image-feature pairings.

One disadvantage of using more complex features is that there is more likely to be errors in their extraction, and this error may prevent correct image-model feature pairings from being tested. Also, Jacobs has recently shown that some basic work using simple point features in the domains of linear combinations of models [UB89] and indexing of $3D$ models [CJ91] does not extend to oriented point features [Jac92].

We wish to sidestep the issue of which is the best feature representation by arguing that no matter what feature type is chosen, there will inevitably be some error in extracting the features from the image, no matter what dimensionality the feature type has, be it a simple $2D$ location, or a $2D$ location with orientation, magnitude, or any combination of other kinds of information. Our goal is to argue that, given a feature type, we can measure the noise associated with it and apply an error analysis to determine the probability of false versus positive identification. Because it was simpler to use simple point features, we have limited ourselves to using only these.

We have measured the noise associated with four feature types, under different conditions. They are

- intersections of straight line segments of a fixed minimum length,
- points of maximum curvature,
- inflection points,
- centers of mass of connected fixed length pixel strings.

The actual algorithm we used to extract these features is unimportant, since we are interested in measuring the variability of each type of feature, given a fixed feature finder. We have attempted to measure the noise per feature type as a function of

- different images of the same scene,
- different degrees of image smoothing,
- illumination.

Interestingly enough, there was some variation in feature locations even for the first image group, where one would expect the images to be identical. In fact, there are slight differences in pixel values between images, probably due to different amounts of light reaching the camera during the imaging stage (fluorescent lighting was used), or possibly due to quantization error. This introduces a level of uncertainty in all the subsequent image processing stages, from image smoothing to edge detection to boundary tracing to subsequent feature extraction.

6.1.2 Procedure for Measuring Noise

To measure the noise associated with each feature type under each kind of condition, three groups of images processed as follows:

- 5 images of a telephone, same illumination, at 5 second intervals. Each image was smoothed with a Gaussian mask with $\sigma = 2$ pixels and Canny edge detected with thresholds of 2 and 4.
- A single image of a fork, with 5 different sized Gaussian smoothing masks: $\sigma = 1, 1.5, 2, 2.5,$ and 3 pixels. Each image was Canny edge detected with thresholds of 2 and 4.
- 5 images of an army knife, varying illuminant position and strength. Each image was smoothed with a Gaussian mask with $\sigma = 2$ pixels and Canny edge detected with thresholds of 2 and 4.

The result of processing yielded 5 different edge maps for each group. The original images were all taken with a Panasonic TV camera with automatic gain control, using an 16mm lens and manually focused. The exact conditions were not measured precisely, since we are interested in them only insofar as they conform to "reasonable" operating conditions. All images consisted of a 720×484 pixel map. Only overhead lighting was used except for the last image group, for which a floodlight was used to change the direction of illumination.

For each edge map, all chains of connected pixels were computed and smoothed, and each feature type found:

- Intersection points — the chains were segmented into straight edge segments using a recursive line-splitting algorithm ([GLP87]), then all intersection points whose distance was < 10 pixels away from the ends of the segments which formed them were kept.

- Maximum curvature points — the derivative of the tangent along the curve was computed, then all minima and maxima exceeding a fixed threshold were kept as point features.

- Inflection points — the zero crossings of the tangent's derivative along the curve whose absolute slope exceeded a fixed threshold were kept as point features.

- Mass centers of chain fragments — the chains were broken into fragments of length 10, then the center of mass of each one taken as a point feature.

For each image group, the feature locations for each of the 5 images were indicated on a single bitmap, color coded by the index of the image from which it came. In Figures 6-1, 6-3 and 6-5, the features are shown only in white, due to reproduction limitations. The correspondences across images were manually indicated for the first three feature types by mousing on clusters which the user believed indicated a feature of a given type. For the fourth feature type, the correspondences were automatically formed by clustering together those features from every image who mutually agreed on their nearest corresponding feature. Figures 6-1, 6-3 and 6-5 show a representative image from each image group, and the point features from all the image groups with their correspondences indicated by circles. The feature types depicted are intersection points (phone), centers of mass (fork), and maximum curvature points (army knife). The inflection point feature was the most unstable, and is not illustrated.

For each cluster within a given feature type, the mean in both the x and y direction was calculated, then for each feature in the cluster, its distance from the mean of the cluster was histogrammed. This yielded, for each image, a histogram per feature type. This histogram is intended to be an accurate sample of the error distribution of the feature type. Some sample histograms are shown next to the pictures from which the features were clustered.

In addition, the error distribution of a third coordinate for each feature type was calculated. For the intersection points, the third dimension is the angle, for maximum curvature points, it is the magnitude of the curvature, for inflection points, the slope, and for centers of mass, the tangent of the curve at that point. The results indicate that the error distribution along this third dimension can also be modelled as Gaussian, and suggests that our method could be extended to incorporate this extra information, though we have not done so.

The calculated variances of each feature type per image group are:

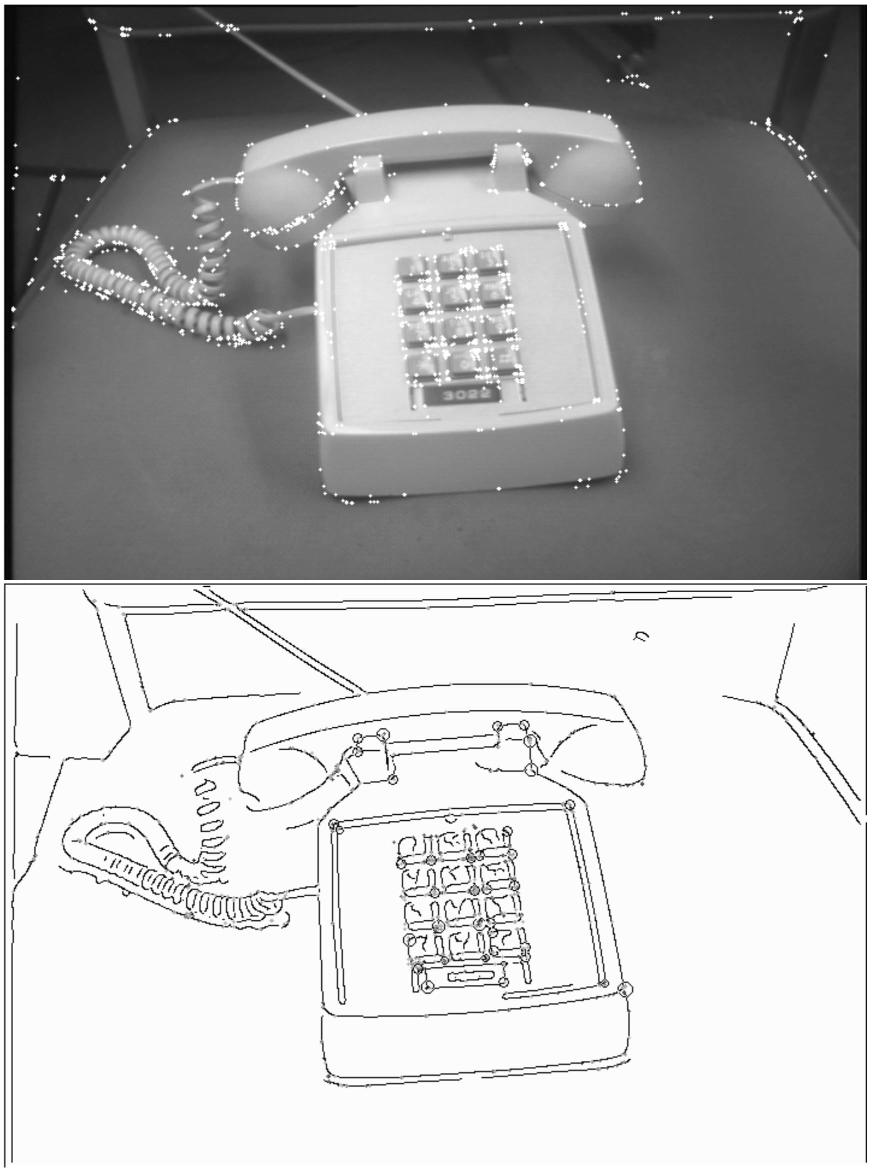


Figure 6-1: First image group. The group consists of 5 images of a telephone, same lighting conditions and smoothing mask. The top figure shows one of the images of the group, with the intersection of straight line segment features from all 5 images of the group superimposed in white. The bottom figure shows the result after Canny edge detection and chaining. The straight line segments from which the intersection points were taken are not illustrated. Superimposed on the bottom figure are the location of the clusters chosen for the noise measurements, indicated by circles. All intersection features located within the bounds of the circle were used as sample points for the noise measurement.

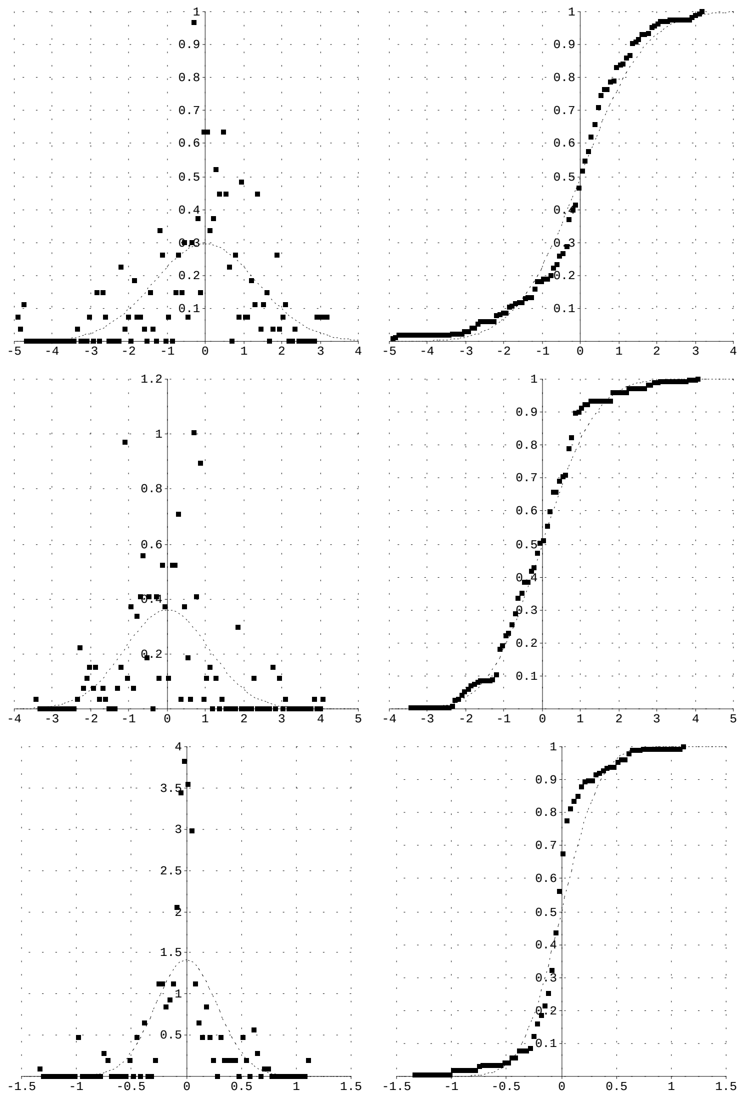


Figure 6-2: Left hand side, from top to bottom: the histograms of the x, y and z coordinates of the intersection features depicted in the previous figure. For intersection features, the z coordinate is the angle of intersection. The Gaussian distribution with mean and variance defined by the histogram is shown superimposed on the graph. On the right hand side is the cumulative histogram, again with the cumulative distribution superimposed.

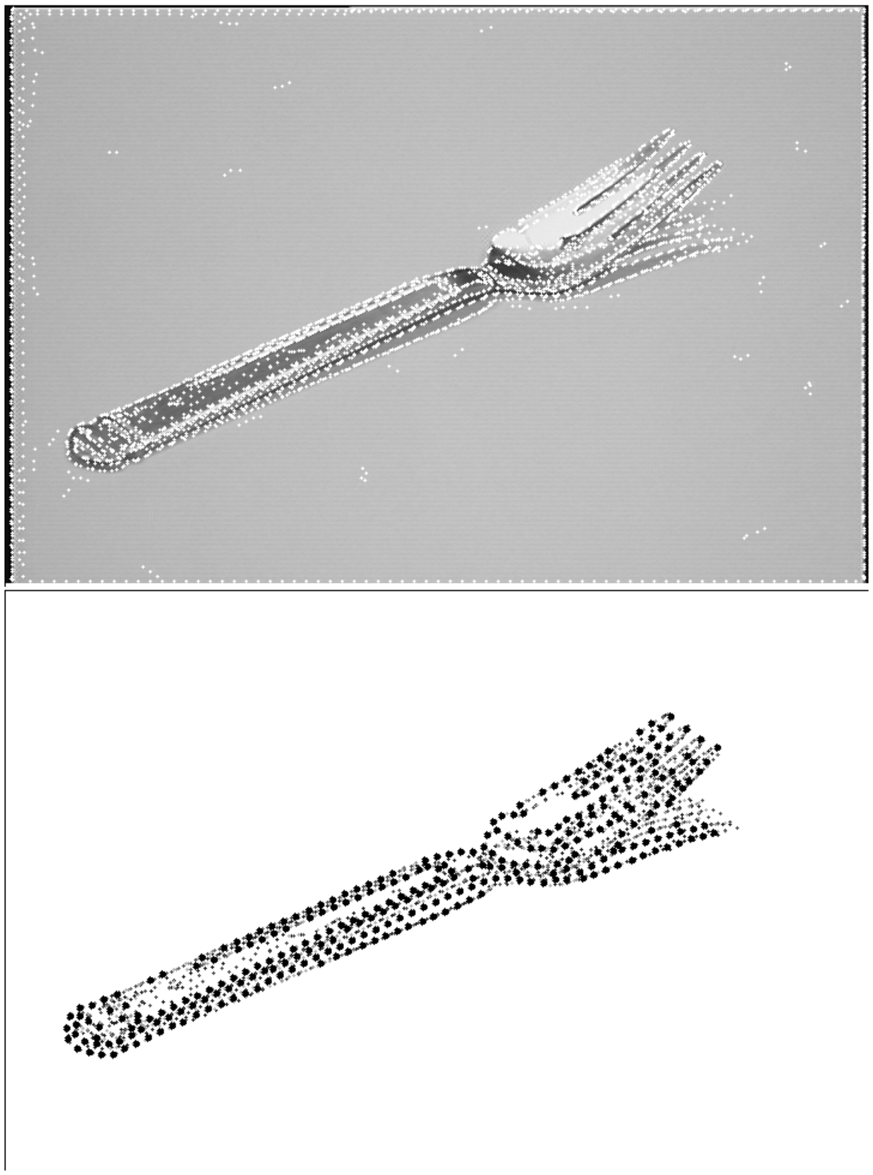


Figure 6-3: Second image group, consisting of the edges from 5 different smoothing masks of the same image. Again, the top figure shows one of the images of the group, but the features shown are the center of mass features from the boundary, randomly broken into segments of length 10. The bottom figure show the clusters, *i.e.*, groups of features which mutually agree upon their nearest neighbor features across all of the images.

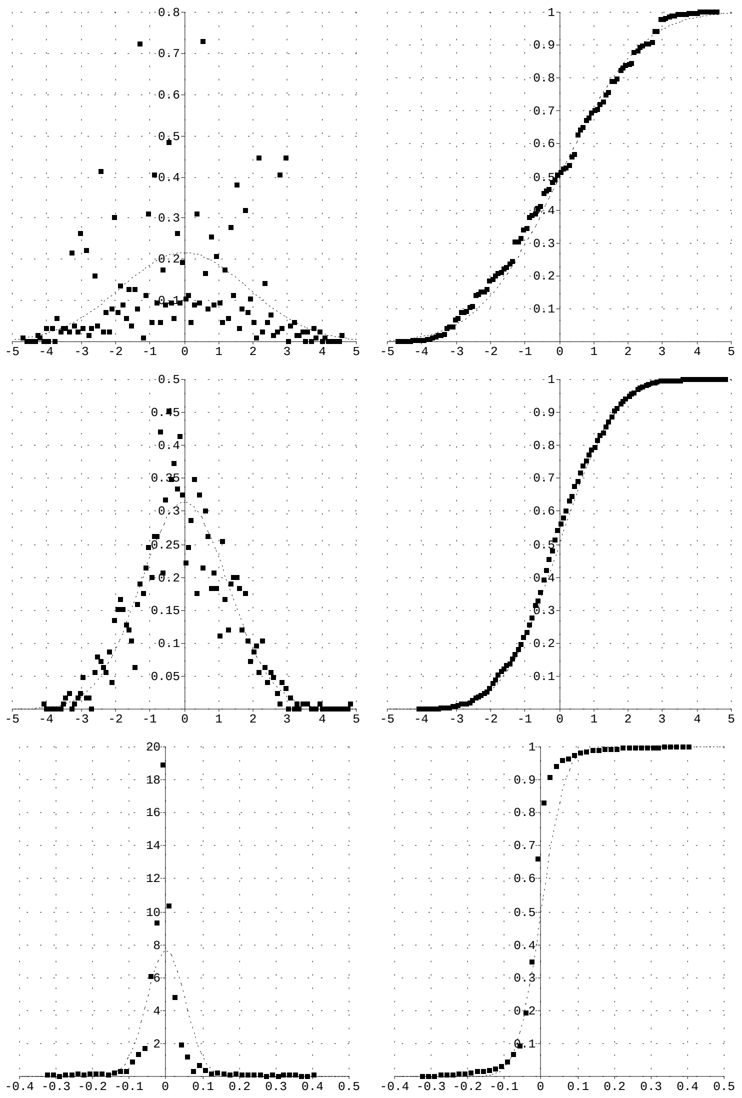


Figure 6-4: Histograms of the x, y and z coordinates of the center of mass features depicted in the previous figure. For this feature type, the z coordinate is the angle of the tangent to the curve at the center of mass.

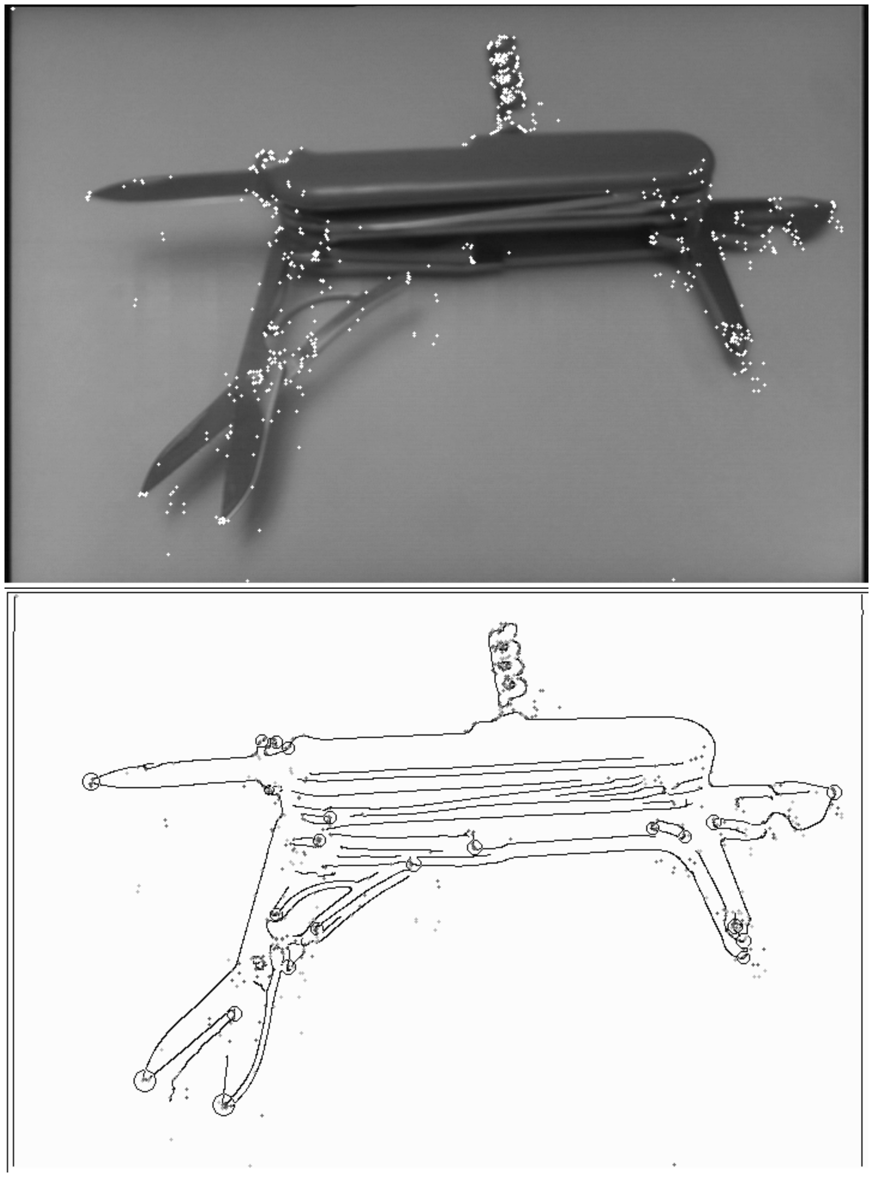


Figure 6-5: Third image group — an army knife under 5 different illuminations. The top figure shows one of the images of the group, with the maximum curvature features from all 5 images superimposed.

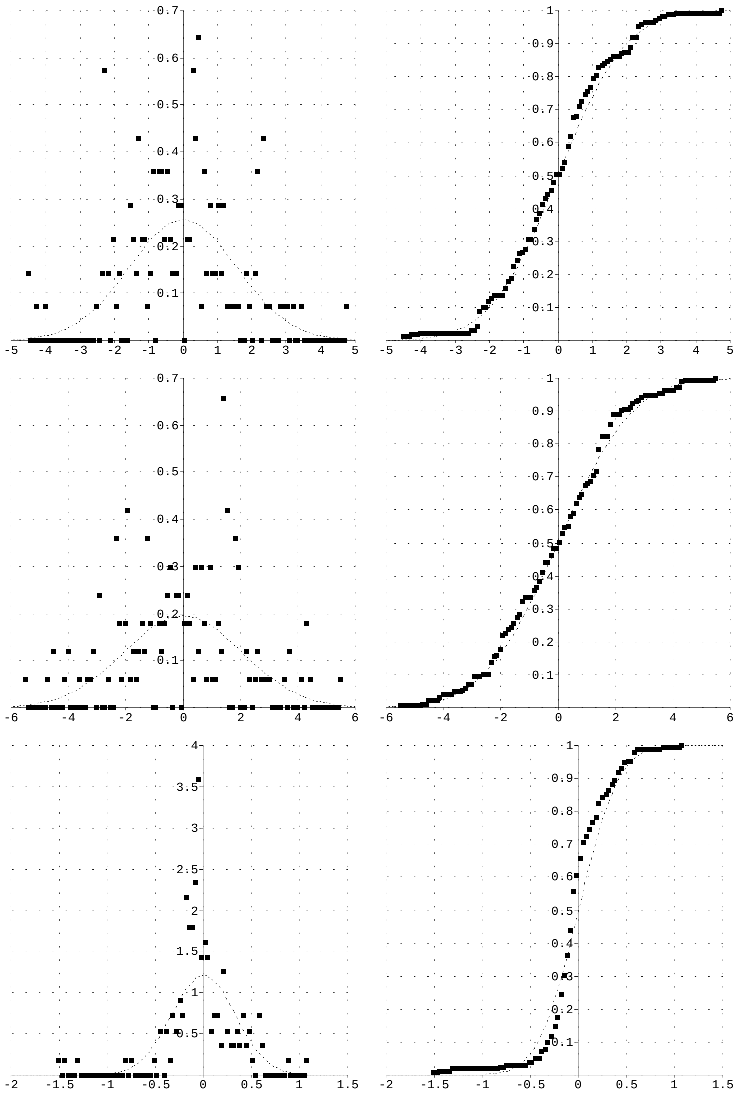


Figure 6-6: Histograms of the x, y and z coordinates of the maximum curvature features depicted in the previous figure. The z coordinate is the magnitude of the curvature.

	# clusters	# points	σ_x^2	σ_y^2	σ_z^2
INTERSECTIONS:					
Phone:	28	325	1.8	1.2	0.079
Fork:	7	231	8.3	2.9	0.52
Knife:	19	85	2.0	2.6	0.30
MAX CURVATURE:					
Phone:	38	221	0.54	0.27	0.017
Fork:	8	71	3.8	1.7	0.093
Knife:	23	169	2.4	4.2	0.11
CENTERS OF MASS:					
Phone:	523	2615	2.0	1.2	0.015
Fork:	305	1525	3.4	1.6	0.0027
Knife:	244	1220	2.7	2.1	0.027
INFLECTIONS:					
Phone:	10	48	0.68	0.96	4.3
Fork:	3	18	6.8	2.6	1.1
Knife:	5	26	5.7	3.3	3.5

Since there were so few reliable clusters of inflection points for each set of images, the results from the distribution can't be taken as representative of any sort of underlying distribution for this feature type. The intersection and maximum curvature points were fairly abundant and stable. For the centers of mass feature, note that the variances are about twice as large in the x direction as the y direction. This is because the contour of the objects were aligned more along the x direction, so the uncertainty would naturally be greater along the contour's tangent due to the manner in which these points were found. We expect that any directional bias of this sort will be symmetrised by the random orientation of the object in the image.

Because the results between image groups are so disparate, we chose to use the average variance for a particular image group as a guide for choosing σ_0 per experiment (again, assuming that the rotational component of the pose distribution allows us to do this). The calculation is simply

$$\sigma_0 = \sqrt{\frac{\sigma_x^2 + \sigma_y^2}{2}}$$

per feature type per experiment. In our subsequent work we will limit ourselves to using only maximum curvature feature types. For these features, the above calculation yields $\sigma_0 \approx .65, 1.7, 1.8$ for the phone, fork, and knife respectively. In the actual experiments it was found that better performance was achieved by using values that were slightly larger than these for the phone and the knife.

6.1.3 Discussion of the Method

There are several aspects of the noise calculation that an observer might take issue with. The first is simply the assumption that the *real* location of the sought after feature is the mean of every cluster; that is, there is no bias in the error distribution. This brings up the question, what is the *real* location of a feature? Suppose the particular feature finder we were using always displaced features to the left by 3 pixels, or, displaced features in an orientation-dependent fashion. Representing the distribution of vectors between where we believe the feature should be, and where the feature finder actually localizes them, can be a problem. However, the complexity disappears if we decide to let the feature finder be the judge of the "actual" location of the features. This will require the model representation to be determined by the feature finder (we will discuss exactly how in the next section). Also, any orientation-dependent directional bias of the feature finder should be randomized by the fact that the orientation of the model in the image is random as well.

Another objection might be that we are not measuring the correct thing: rather, what should be measured is the displacements of a single feature from, say, 100 different images. Instead, what we are really doing is sampling many different random variables, and as such, it is no wonder that we are ending up with a close to Gaussian error distribution, since by the Central Limit Theorem, the sum of many different random variables will be Gaussian, no matter what their individual distributions. The answer to this charge is that this sum of random variables is exactly the distribution that we are interested in measuring; far from invalidating the method, this objection reinforces it.

Another question might be about the manner in which the clusters were formed; that is, at least for the 3 out of 4 of the feature types, we manually clustered together those features that seemed to be close to a location at which it seemed reasonable that a feature should appear. There was no guarantee that there was exactly one feature from each image group in the cluster; some clusters probably were missing representative features from some images, some clusters probably contained several features from the same image. Also, we didn't take *all* possible clusters, only those which seemed subjectively appropriate. Despite these issues, we claim that since the model representation is chosen by the user (*i.e.*, which model features comprise the representation), it is not unreasonable for the user to determine the range of locations at which a projected model feature may appear. As to the question of variable number of features per image included in a single distribution, we note that if the image feature extraction process drops a feature for one of the images in the distribution, there is nothing we can do. If one image contains several features close to the desired feature location, then including all of them in the distribution implies that any of them is a feasible match for a model feature which projects to near that location.

6.1.4 Using Different Feature Types

Now that we have several feature types, each with their associated σ_0, we look at the problem of combining information for a hypothesis consisting of pairings of different feature types: i.e., suppose we have a hypothesis consisting of a size 3 pairing of feature types 1, 2 and 3 with associated error standard deviations σ_1, σ_2 and σ_3 respectively. Then the possible locations of a fourth point (α, β) of feature type 4 with error standard deviations σ_4 remains centered at the expected location, but with variance

$$\sigma_1^2(1 - \alpha - \beta)^2 + \sigma_2^2\alpha^2 + \sigma_3^2\beta^2 + \sigma_4^2$$

We can still weight the occurences of a corroborating point as before. However, the calculation of the densities for V_M and $V_{\overline{M}}$ become more involved, since we can no longer use the same approximation for $f_{\sigma_e|H}(\sigma)$ and $f_{\sigma_e|\overline{H}}(\sigma)$. If the σ's are not all equal, these density functions are dependent on the four new random variables $\sigma_i, i = 1 \ldots 4$, whose distributions are different for each model. Though it is still possible to do the calculation, it is much more complex than before.

6.2 Building the Planar Model

Building models for $2D$ planar objects is particularly easy, since a single image contains sufficient information to do it. In order to be able to use the error model which we have analysed and measured, we build our model as follows: a single image of the model at in an arbitrary pose is run through the feature detector. The user then clicks on clusters of points appearing near the location of a desired feature; the mean of this cluster is then incorporated into the model representation. This method of building the model is compatible with the way in which we measure and represent error in our analysis.

6.3 Applying the Error Analysis to Automatic Threshold Determination

Here's an example of an application of our error analysis to a typical problem in object recognition — automatic threshold determination for a system which uses our recognition algorithm. Optimally, we would like to build a demonstration system in which, given a model and some image, the user specifies a certainly level up front, *i.e.*, "I don't want the system to tell me about anything unless it is 90% certain that it is an instance of the model". In order to achieve this, it would have to be the case that

$$\frac{N_D P_D}{N_D P_D + N_F P_F} \geq 0.9$$

in which N_D and N_D are the total number of true and false hypotheses, respectively. However, in our problem these numbers are unknown; the only control we have is over the values of P_F and P_D and this does not bound the certainty of the result.

Thus our demo will be as follows: the user is able to specify a desired value for either P_F or P_D. The image is processed to find the number of model and image features, and then the system constructs the associated ROC curve and finds the implied P_D (if a P_F was specified, otherwise the implied P_D for the specified P_F), and what threshold will give that performance. It then notifies the user of the implications of the choice.

As it turns out, the simplifying assumption that the clutter is randomly distributed in real images is not only incorrect, but the deviation from the modeled error also very strongly affects the way the system works. The reasons for this are illustrated by an extreme example: imagine an image in which n feature points appear in the left side of the image while the right side has none. When we project an error disk into the image, if it appears in the left side it is twice as likely to encompass an image point at random than our prediction. In addition, the denser region will more likely be sampled (in this example, will definitely be sampled) for the 3 random image points chosen at random to form the pose hypothesis, making it much more likely that the remaining error disks also project to the left side of the image. These two effects result in a much higher effective density than indicated by the mere number of points appearing in the image.

We can attempt to fix to the problem in two ways: we can either estimate an effective image density as a function of density variability across the image and use a single ROC curve and threshold per image, as we have been doing up until now. Or, we can calculate the effective density per hypothesis (that is, density of the region in which the projected model falls), and use a different ROC and threshold per hypothesis. We chose the latter approach, that is, we chose to calculate an ROC curve not for an entire picture (since a single value for n does not suffice to describe all hypotheses when the density is so variable across the image) but rather on a per-hypothesis basis. So, instead of being able to predict a single threshold for all hypotheses emanating from an image, we calculate the threshold every time we test a different hypothesis. Since we are changing the ROC curve per hypothesis, we can choose the threshold to constrain either the false alarm rate or the true detection rate, but not both.

6.3.1 The Problem with the Uniform Clutter Assumption

In this section we illustrate in more detail the problem with the uniform clutter assumption. First we show the original demonstration in which we found a discrepancy between our predicted and actual behavior of the system, and subsequently, a sequence of experiments to isolate its cause.

Initially we built a demo which runs in one of two modes, *random* and *exhaustive*. For both modes, the system takes as its input

- The model, consisting of a list of $2D$ feature locations,
- The image, also consisting of a list of $2D$ feature locations,
- Estimated occlusion level, which is a number between 0 and 1, inclusive,
- The value of σ_0 for this feature type.

In addition there are two optional arguments SUB-MODEL and SUB-IMAGE, which are subsets of the model and image, respectively. If these optional arguments are non empty, the demo runs in *exhaustive* mode, otherwise it runs in *random* mode. The motivation for these optional arguments is to limit the number of hypotheses tested to a reasonable size, and to be able to include some correct hypotheses among those tested. For instance, in the telephone test with 33 model features and 250 image features, the number of hypotheses, though polynomial, is still $\approx 4 \times 10^{11}$. Even if we could check one hypothesis per second, this would still take 13 thousand years, risking a very dull demo for the user. However, when a sub-model and sub-image group of size 4 that correctly correspond to each other is specified, then the demo exhaustively tests 96 hypotheses of which 4 are correct.

When the demo runs in exhaustive mode, the user is asked to specify a desired P_F. The number of model and image features implies a single ROC curve, and the user specified P_F implies a particular P_D and threshold. The system reports to the user the implied P_D and proceeds to cycle through all size 3 hypotheses formed by correspondences between the model and image subsets, showing the user all hypotheses which score above the threshold. The user answers each query with "correct" or "incorrect", and the number of times the system makes a mistake is tallied.

When the demo runs in random mode, the user is asked to specify only a P_F, after which 1000 randomly chosen hypotheses are tested, the assumption being that the probability of randomly choosing a correct one is infinitesimal.

The output of the demo is a histogram of the weights of all the hypotheses tested. If the demo was in random mode, the normalized histogram should have the same distribution as $W_{\overline{H}}$. If the demo was in exhaustive mode, then the predicted to empirical (P_F, P_D) point is illustrated on a graph.

We ran the demo in exhaustive mode on the telephone image shown in Figure 6-7 with a user specified certainty level of .99. The first part of the figure shows the grey-scale image of the telephone with the points chosen to comprise the model indicated in white. The bottom figures show a correct and incorrect hypothesis that the system came up with that exceeded the threshold.

In terms of performance, the demo failed quite dramatically, showing far more incorrect hypotheses exceeding the predicted threshold than should have been the case. Upon inspection the cause for this breakdown is easily identified; running the demo in random mode with the same model and image produced the histogram shown in Figure 6-8. The figure shows the predicted density of $W_{\overline{H}}$ for $m = 30, n = 248$ superimposed on the normalized histogram of the actual density function. Both the mean and variance are much larger than they should be. The question is, what is causing such a large discrepancy?

We pinpointed the problem by performing the following sequence of experiments.

(A) First we eliminated the possibility that the sheer number of model and image points was the culprit by performing the same simulation as was done to derive the ROC curves in Chapter 4 for the same value for m and n as in the telephone model and image. The results of the simulation follow the predictions of the error analysis very satisfactorily and normalized histogram of the weights, approximating the density of $W_{\overline{H}}$, is shown in Figure 6-9. This indicates that the problem lies elsewhere.

(B) To eliminate the possibility that something about the model itself was causing the behavior (for instance, the model symmetry), we created an image in which the model was present, but all the remaining image points were redistributed uniformly over the image, maintaining the same values of m and n. We then ran the demo in random mode, and found that the resulting histogram of weights also conformed to the predictions of the error analysis (Figure 6-10). This also pinpoints the problem, since the only difference between this experiment and the original demo was the distribution of the clutter points, thus isolating the cause of the discrepancy.

(C) Lastly, we tested to make sure that model pose did not affect the results by running the same test as (B), but translating the model points to the very top of the image (Figure 6-11). This did affect the weight histogram slightly, but in the other direction — that is, it served to make the P_F prediction an overestimate, not an underestimate, of the clutter effects.

6.3.2 Finding a Workaround

Density Correction Factor

Our first attempt at fixing the problem is to determine the effective image density. If we can do this, then we can maintain a single ROC and threshold per image. Let us define a quantity which we will name the *density correction factor*. This is an empirically derived number which serves as a kind of amplification factor, in that it scales the actual number of image features to yield the effective number of image features. The procedure for finding it is simple: we subdivide the image into a 16×16 grid of regions, each one with approximately uniformly distributed image clutter. During the demo, every time an error disk is projected into the image, a

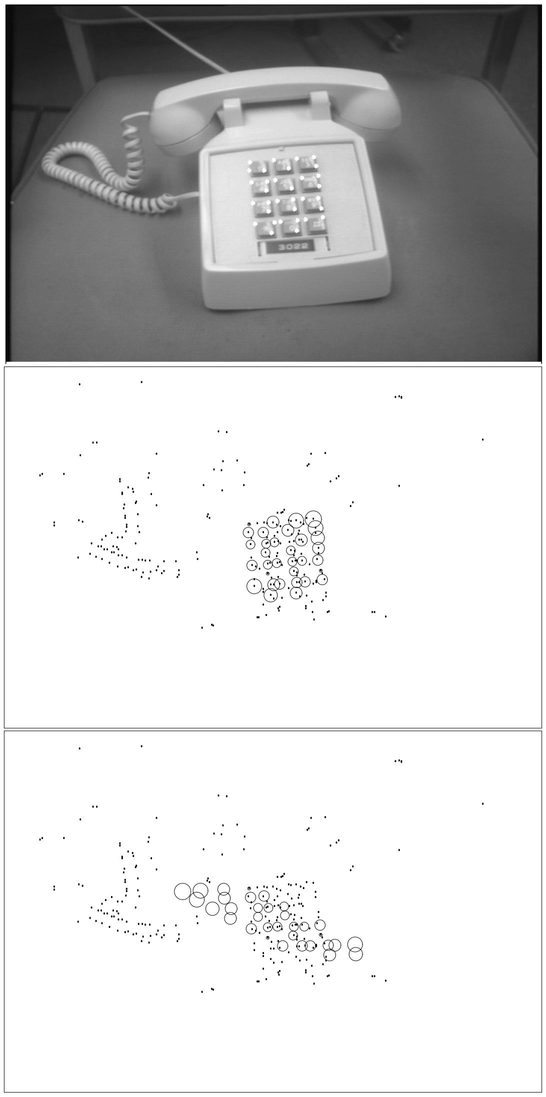


Figure 6-7: The top figure shows the original image, with the feature points chosen for the model indicated in white. The middle figure is a correct hypothesis that exceeded the predicted threshold, and the bottom shows an incorrect one. The points indicate image feature points, and the circles indicate projected weight disks.

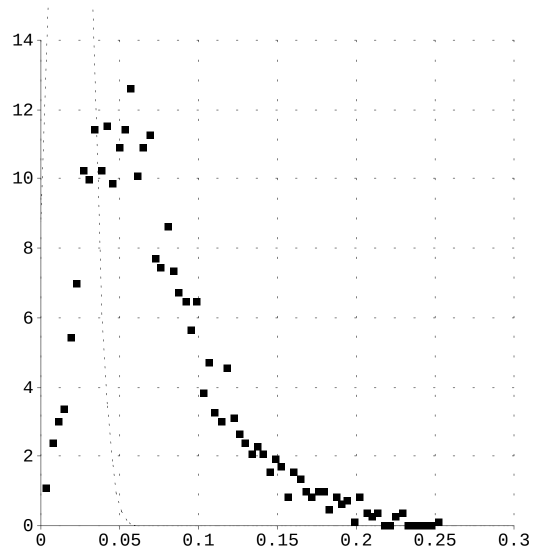


Figure 6-8: The histogram for the weights of incorrect hypotheses chosen from the original image. Note that the mean and variance greatly exceeds those of the predicted density of $W_{\overline{H}}$ for $m = 30, n = 248$.

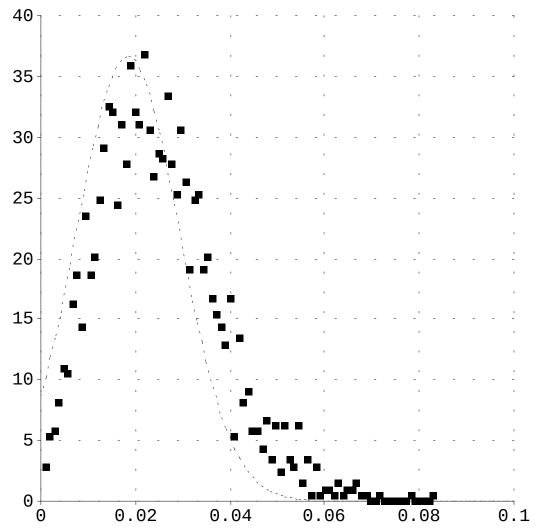
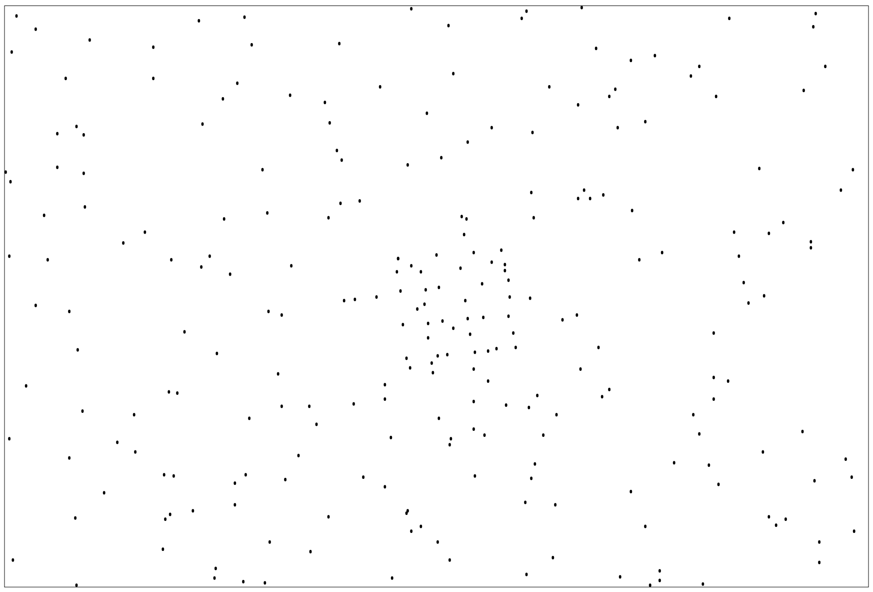


Figure 6-9: The histogram of $W_{\overline{H}}$ for 2500 randomly chosen hypotheses For this model and image, $m = 30$, $n = 248$, when the clutter is uniform and the model does not appear in the image. The prediction closely matches the empirical curve.



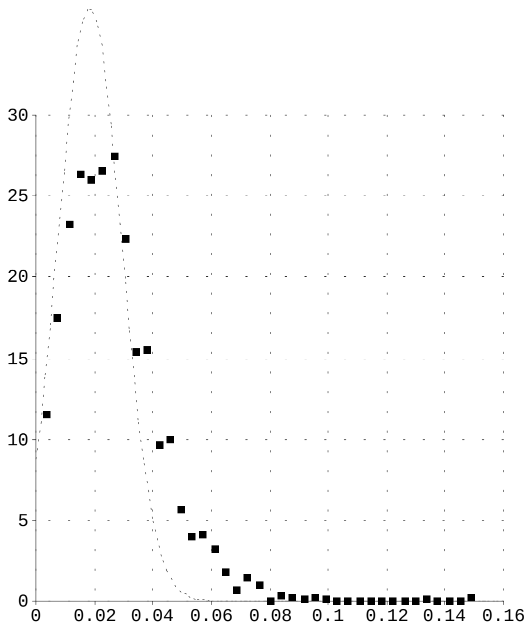
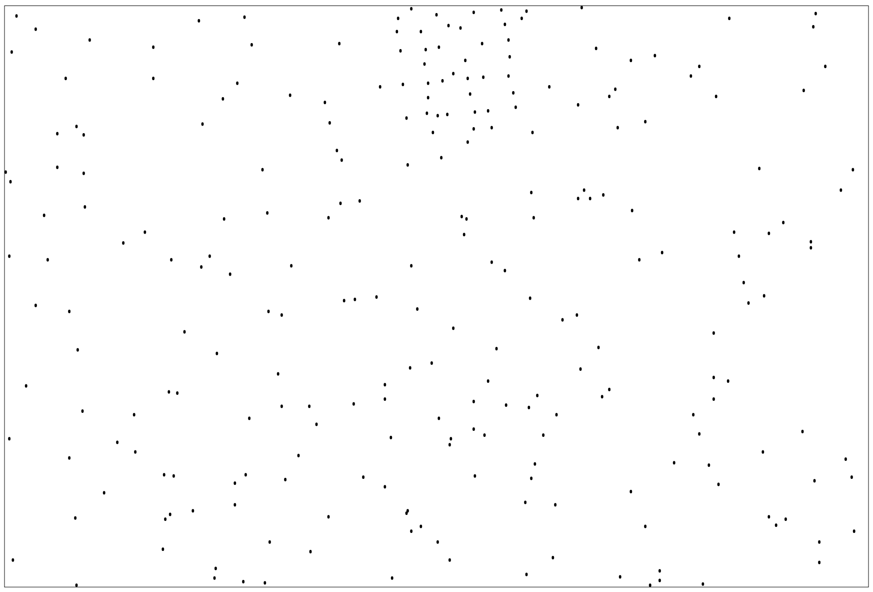


Figure 6-10: The top picture shows the model present in the image, but with the clutter points redistributed uniformly over the image. The actual density of $W_{\overline{H}}$ closely matches the predicted density.



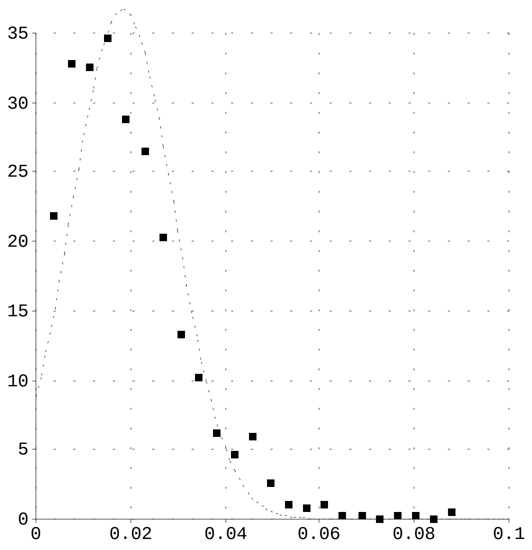


Figure 6-11: When the model points are displaced to the top of the image, the resulting density of $W_{\overline{H}}$ is affected, but in the other direction. That is, the noise effects are now slightly overestimated instead of underestimated.

count for that region is incremented. Finally, the histogram is normalized (turning it into a $2D$ probability density function for the probability of being hit by an error disk), and from this we calculate the expected number of image points per region by multiplying the normalized histogram by the number of image points per region. Finally this is turned into the density correction factor by multiplying it by (number of regions / number of image points).

We used the density correction factor to get an idea of the difference between the number of clutter points that we are using in our calculation versus the effective amount of clutter. We would expect the demo to produce a correction factor of ≈ 1 when run on a completely uniformly distributed image, with a higher number indicating a higher variability in density. As expected, the original image resulted in a correction factor of ≈ 4.57, while experiment A (completely uniform image) yielded a correction factor of ≈ 1.13. Experiments B (model with uniform noise), and C (displaced model with uniform noise) yielded intermediate values of ≈ 2.03 and ≈ 1.75, respectively. Note that knowing this number doesn't directly suggest a solution, since a correction factor of > 1 doesn't imply that the method breaks down. Rather, it simply confirms that regions of high clutter density are actually hit more often than the low clutter regions, as we suspected.

Threshold per Hypothesis

It is clear from our work up until this point that the uniform clutter assumption does not adequately model the clutter in real images, and we cannot fix the method by amplifying the number of image points in a naive way. Instead, we have modified the method to work on a per-hypothesis basis. That is, we use the same grid of uniformly distributed density regions to estimate the effective image density every time we project the model into the image, and calculate the ROC curve and threshold, assuming a fixed certainty level. This implies that we cannot predict the overall probability of a miss at the outset of the demo, since it changes for every hypothesis, but we can set the threshold to maintain a fixed probability of false alarm.

Implications of the Uniform Clutter Assumption

We have definitively shown that the assumption of uniformly distributed clutter underestimates the negative effects of clutter for this recognition algorithm. Though the number of images we have examined is not enormous, it is quite safe to say that one cannot assume that the feature points in an image will be so distributed, and so any analysis which depends on this assumption will underestimate the effects of clutter. To our knowledge, all error analyses that have been done until now in the field of computer vision have used this assumption.

6.4 Demo

We revised the demo to take into account the density correction per hypothesis. The demo runs as before, with two changes. First, the user is able to specify either a P_F or P_D level. If a P_D level is specified, then only correct correspondences are tested. These correct correspondences are passed to the demo through the parameters SUB-MODEL and SUB-IMAGE. Second, the user is not told at the outset what the implied P_F rate (or P_D rate) will be for the specified level, since it changes for every hypothesis.

In this section we present the output of the demo in action. In theory the demo should work for any planar object in an arbitrary pose, but in practice the fact that we are working under perspective instead of orthographic projection will lead to errors that we do not expect will be adequately modeled by a Gaussian. For this reason we cannot vary the pose of the object in our experiment (except for translations in the x and y direction), and so for this limited case we can include $3D$ models in our domain.

For the demo, then, we can use all the $3D$ objects that we have been working with until now, namely, the telephone, fork, and army knife. We work with a single object at a time. A model of an object is constructed from a single image of it by first processing the image to find all the feature locations, displaying their $2D$ locations, and then mousing on points which we want to be in the model. To test the validity of the error analysis, we run the demo on a different image than the one from which the model was constructed.

6.4.1 Telephone

In this test, we used the telephone model that was shown in Figure 6-7. For every hypothesis, the effective density is calculated, and the ROC curve for that model size and image density determined. The threshold associated with the ROC point (P_F, P_D) on the curve is found, and if the weight exceeds the threshold, the hypothesis is displayed.

Table 6.1 shows the results of experiments in which a particular rate of either false alarm or true detection was given, then the threshold was dynamically set per hypothesis to maintain the specified rate. The same three experiments were performed for four different values of σ_0 including that found in Section 6.1.2. The first column is the σ_0 value that was assumed for the experiment. The second and third columns contain the user specified P_F or P_D. In the fourth column is the total number of hypotheses tested. The fifth, sixth and seventh columns contain the expected number of hypotheses of those tested that should pass the threshold, the actual number of hypotheses that pass the threshold, and the error bar for the experiment (we show one standard deviation $= \sqrt{tP_F(1-P_F)}$, t = number of trials. The actual P_F (or P_D) is shown in the eighth column. The last column shows the average distance of all

σ_0	P_F	P_D	Total	Expect	Actual	Error	Act P_F/P_D	E[D]
0.5	.01		1081	11	15	3.27	.014	-2.1
	.001		1121	1	0	1.06	0	
		.9	512	461	503	6.8	.98	5.1
0.65	.01		1082	11	29	3.27	.027	-3.7
	.001		1080	1	0	1.04	0	
		.9	510	459	510	6.8	1.0	2.7
1.0	.01		1073	11	23	3.26	.021	-3.24
	.001		1094	1	0	1.05	0	
		.9	514	463	501	6.8	.97	5.5
2.0	.01		1113	11	21	3.32	.019	-1.95
	.001		1091	1	0	1.05	0	
		.9	508	457	496	6.8	.98	4.2

Table 6.1: Results of experiments for the telephone. The first column is the σ_0 value that was assumed for the experiment. The second and third columns contain the user specified P_F or P_D. In the fourth column is the total number of hypotheses tested. The fifth, sixth and seventh columns contain the expected number of hypotheses of those tested that should pass the threshold, the actual number of hypotheses that pass the threshold, and the error bar for the experiment (we show one standard deviation $= \sqrt{tP_F(1-P_F)}$, $t =$ number of trials). The actual P_F (or P_D) is shown in the eighth column. The last column shows the average distance of all the hypotheses that passed the threshold from E[W_H].

the hypotheses that passed the threshold from E[W_H], The distance is given in terms of the standard deviation of W_H, that is:

$$D = \frac{\mathrm{E}[W_H] - w}{\sqrt{\mathrm{Var}\,(W_H)}}$$

where w is the weight of the hypothesis which crossed the threshold.

This model and image contained 33 and 231 features, respectively. In our experiments we tested several values of σ_0 to see how varying that value would affect the accuracy of our predictions. In Section 6.1.2 when we measured the noise associated with the maximum curvature feature type for the phone image group, we determined that $\sigma_0 = 0.65$. As we can see from the table, the results were not significantly different for the different values of σ_0, though a value of $\sigma_0 = 0.5$ seemed to be most accurate for the experiment $P_F = 0.01$. Oddly enough, using the measured value for σ_0 resulted in the worst predictions.

For all of the experiments, we see that the threshold predicted to maintain a specified P_F of .01 did not achieve the desired false detection rate. The reason for this is that our assumption that the density function of $W_{\overline{H}}$ is Gaussian is false; in fact, the upper tail of the actual distribution of $W_{\overline{H}}$ contains more of the distribution than a Gaussian with the same mean and variance would. Despite this, the predicted thresholds for

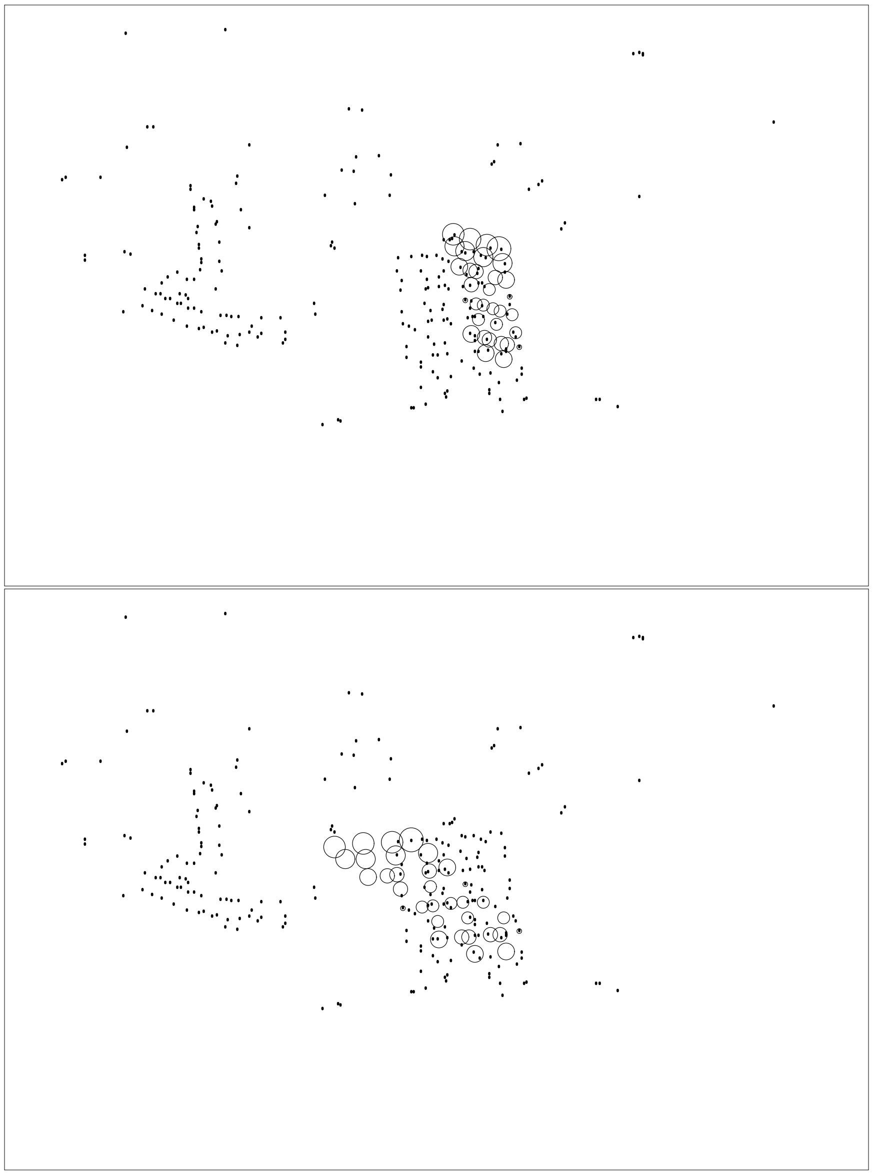


Figure 6-12: The incorrect hypotheses that fell above the threshold chosen to maintain a P_F of 0.01. For these experiments, $\sigma_0 = 2.0$. The circles show the locations of the projected weight disks, while the points show the feature locations.

σ_0	P_F	P_D	Total	Expect	Actual	Error	Act P_F/P_D	E[D]
1.0	.01		1157	12	44	3.38	.038	-1.6
	.001		1139	1	0	1.02	0	
		.9	514	463	0	6.8	0	
1.8	.01		1075	11	51	3.26	.047	-.98
	.001		1017	1	1	1.01	.001	-2.77
		.9	512	461	318	6.8	.62	-.048
2.0	.01		1126	11	40	3.34	.036	-.93
	.001		1100	1	0	1.05	0	
		.9	541	487	423	7.0	.78	-.2
3.0	.01		1020	10	29	3.18	.028	-.44
	.001		1044	10	2	1.0	.002	-1.06
		.9	517	465	431	6.8	.83	.5

Table 6.2: Experimental results for the army knife. Though σ_0 for this image group was determined to be 1.8, we see that the predictions for a value of $\sigma_0 = 2$ or 3 are much better. The columns indicate the σ_0 used for the experiment, either P_F or P_D , the total number of hypotheses tested, the expected number of hypotheses to score above the threshold, the actual number that scored above the threshold, and the error bar for this value (we show one standard deviation $= \sqrt{tP_F(1-P_F)}$, t = number of trials). The actual P_F (or P_D) is shown in the next column, and the last column shows the average distance of all the hypotheses that passed the threshold from E[W_H]

an even lower probability of false alarm (*i.e.*, $P_F = 0.001$) work well.

Lastly, we note that on the average, even those false hypotheses which passed the threshold had weights which were still significantly below the mean of W_H, while the weights of true hypotheses passing the threshold were significantly above. Though not justified by the analysis, this information might also be used to discriminate between true and false hypotheses passing the threshold.

6.4.2 Army Knife

The same experiment was done with the army knife. This example differs from the previous example in that we used far fewer model points, 14 versus 33. This brings the value of E[W_H] much closer to E[$W_{\overline{H}}$], and in general we found that the system behaved less well due to this. For this experiment, the number of model and image features were 14 and 162 respectively. The model plus two examples of hypotheses which fell above the threshold are shown in Figure 6-13.

The σ_0 for this feature type was determined in Section 6.1.2 to equal 1.8. Referring to Table 6.2, we see that using this value for the sensor noise results in a very poor prediction for P_D. For example, for a specified P_D value of 0.9 we can see that the actual percentage of true hypotheses that passed the threshold was only 0.62.

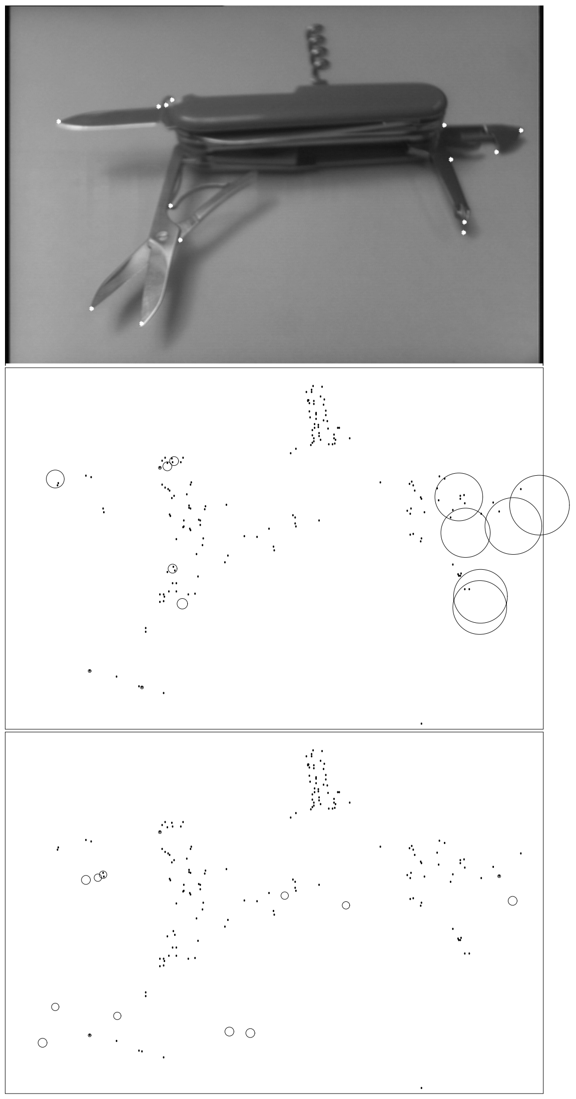


Figure 6-13: The top figure shows the original image, with the feature points chosen for the model indicated in white. The middle figure is a correct hypothesis that exceeded the predicted threshold, and the bottom shows an incorrect one.

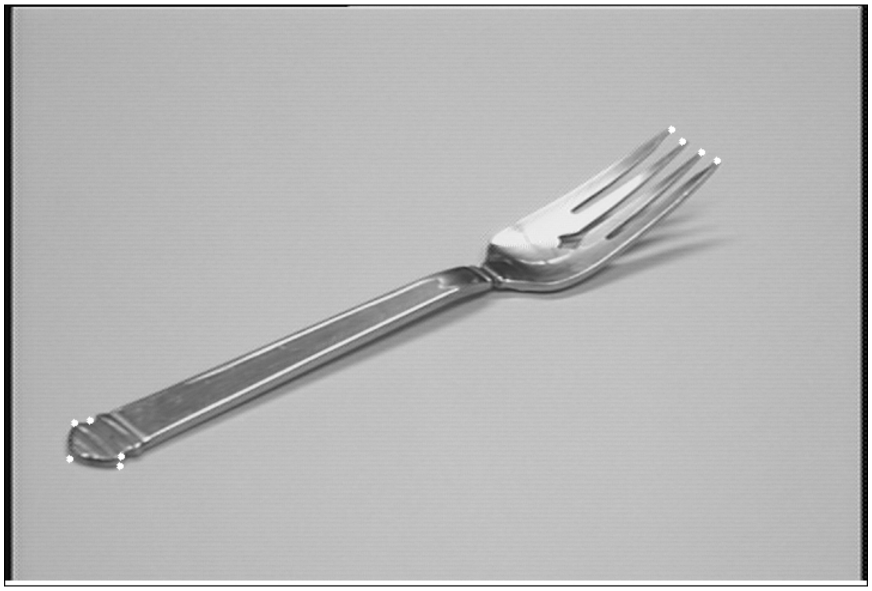


Figure 6-14: The model of the fork superimposed in white onto the original image.

Raising the value of σ_0 improved the prediction for both P_F and P_D, with the best performance prediction resulting from using a value of $\sigma_0 = 3.0$, with $\sigma_0 = 2.0$ a close second. However, even the predictions for these σ_0 values results in too high a false alarm rate when a value of $P_F = 0.01$ is specified — again, using a Gaussian approximation for the density of $W_{\overline{H}}$ causes us to underestimate the extent of the upper tail of the actual distribution. As in the previous model, the predicted performance closely matched actual performance for a P_F value of 0.001.

Unlike in the previous set of experiments with the telephone, there is not much difference between the average distance from $\mathrm{E}[V_M]$ of true and false hypotheses which pass the threshold. Whereas before there was a chance that this extra information might further help discriminate between true and false hypotheses which pass the threshold, for this model and image the extra information is no help.

6.4.3 Fork

The same group of experiments for the fork image group are depicted in Table 6.3. The model contained 9 feature points and is shown in Figure 6-14, while the image contained 170. The table indicates that the predictions for the $\sigma_0 = 1.7$ and 2.0 give the best results of the group, though the prediction for $\sigma_0 = 1.7$ is worse for a specified $P_F = 0.01$, and the prediction for $\sigma_0 = 2.0$ is worse for $P_F = 0.001$. The former value, $\sigma_0 = 1.7$, was the value determined for this image group in Section 6.1.2. Generally, performance predictions were not quite as successful for this model as for the first two.

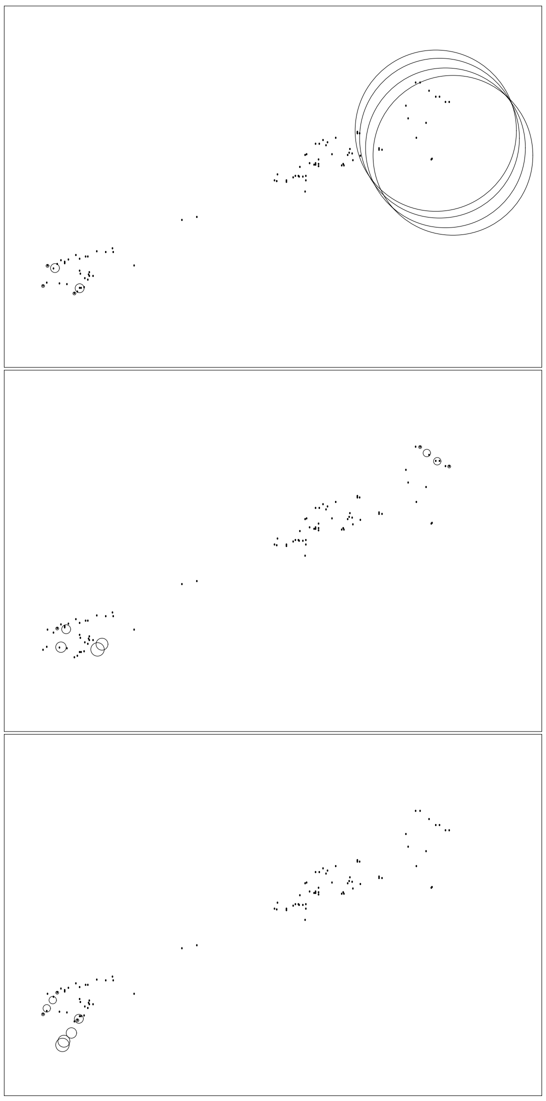


Figure 6-15: Three kinds of false positives that occurred for a highly symmetric model.

σ_0	P_F	P_D	Total	Expect	Actual	Error	Act P_F/P_D	E[D]
1.0	.01		1093	11	36	3.28	.033	-.058
	.001		1070	1	1	1.03	.001	-1.84
		.9	508	457	468	6.8	.92	2.48
1.7	.01		1052	11	56	3.23	.053	.27
	.001		1065	1	1	1.03	.001	1.77
		.9	529	476	475	6.9	.90	2.55
2.0	.01		1011	11	43	3.16	.043	-.28
	.001		1020	1	7	1.01	0.007	1.82
		.9	523	471	467	6.9	.89	2.4
3.0	.01		1043	10	59	3.21	.056	.67
	.001		1057	10	17	1.03	.016	1.5
		.9	512	461	444	6.8	.87	1.6

Table 6.3: Experimental results for the fork model.

6.4.4 The Effect of Model Symmetry

While performing the previous experiments, it was noted that incorrect hypotheses which roughly aligned the model along an axis of symmetry in its image projection would be more likely to get a high score and pass the threshold. The second hypothesis in Figure 6-12 is an example of this phenomenon, as well as the first two false hypotheses shown in Figure 6-15. These "symmetric" hypotheses are more likely to be sampled when the three image points in the basis actually arise from the model (while not correctly corresponding to the model basis tested). To test this effect, we ran the above experiments for some sample values of σ_0. The three points in the image basis used for the random correspondence was restricted to those arising from the model.

The results in Table 6.4 show that on the whole, the restriction to hypotheses using image bases arising from the model causes a higher false positive rate than the same experiment without the restriction. This does not prove that the model symmetry is entirely causing this effect, especially since the knife model, which is not symmetric, shows the same tendency towards a higher false positive rate, while the fork, which is highly symmetric, does not (at least for the $P_F = .01$ experiment). Nonetheless, we suspect that model symmetry may is a contributing factor, though more tests would have to be done to settle the matter conclusively.

6.4.5 Comparison to Results Using the Uniform Clutter Assumption

In the previous sections we performed experiments in which we dynamically set the threshold per hypothesis, depending on which region of the image the model projected

Model	σ_0	P_F	Total	# Found	This P_F	Previous P_F	Error
Phone	1.0	.01	1101	37	.034	.021	3.3
		.001	1126	2	.002	0	1.06
Knife	3.0	.01	1096	66	.06	.028	3.29
		.001	1015	11	.011	.002	1.01
Fork	2.0	.01	1070	46	.043	.043	3.25
		.001	1067	22	.021	.007	1.03

Table 6.4: Experimental results when all the points in the image bases tested come from the model. The first and second columns contain the model and σ_0 tested. The next columns contain the specified P_F , total number of hypotheses tested, and number of hypotheses that passed the threshold. The next two columns contain the actual P_F for this experiment and the value for the same experiment in which the tested image bases are not constrained to come from the model (this value was taken from the previous group of experiments. Finally the last column is the error bar for the experiment, which we took to be one standard deviation = $\sqrt{tP_F(1-P_F)}$, t = number of trials.

Model	σ_0	P_F	Total	# Found	This P_F	Previous P_F
Phone	1.0	.01	1080	418	.39	.021
Knife	3.0	.01	1114	528	.47	.028
Fork	2.0	.01	1061	490	.46	.043

Table 6.5: Experimental results when uniform clutter is assumed. The first and second columns contain the model and σ_0 tested. The next columns contain the specified P_F , total number of hypotheses tested, and number of hypotheses that passed the threshold. The next two columns contain the actual P_F for this experiment and the value for the same experiment in which the effective density is calculated per hypothesis, and the threshold dynamically reset.

to under the tested hypothesis. We have shown the method working reasonably well despite a slightly higher false positive rate than expected for some cases. One source of the problem may possibly be that the image was indiscrimately broken into a 16×16 grid for the effective density calculation, in which the clutter was assumed to be uniformly distributed within a rectangle of the grid. This approximation may not be quite correct for the images used.

Lest the reader question the advantage of using a dynamic threshold, we illustrate some experiments for the case when clutter is assumed to be uniform. That is, a single ROC curve and threshold is calculated for the entire image, and any hypothesis which falls above it is counted. The results are shown in Table 6.5. The table clearly shows the necessity of using dynamic thresholds, and one can appreciate how well our method actually performs when compared with these results.

6.5 Conclusion

In this chapter we have demonstrated the applicability of many of the ideas developed so far. First we argued that the positional error of features due to effects such as lighting and smoothing are well modeled by a Gaussian approximation, and showed how to determine the size of the Gaussian for different feature types. Finally we showed an example of automatic threshold setting applied to the problem of finding a correct correspondence between model and image features. It was found that in two of the three image groups, the better predictions were achieved when using a σ_0 for each image group that was slightly higher than that found in our measurements.

It was demonstrated early on that the assumption that clutter is uniformly distributed over an image greatly underestimates the effects of clutter on the algorithm we are using when applied to a real recognition problem. Because of this, we were not able to apply exactly the same approach that we demonstrated on simulated images in the Chapter 4; rather, we had to adjust the threshold for every model pose hypothesis, depending on the clutter levels of the regions that the model projected to. This meant that when a hypothesis projected to a region of high density, it needed far more evidence to be considered a possible detection than otherwise. We showed this approach working reasonably well for a small group of images. When given false alarm rates of .01 and .001, the system was able to recalculate the threshold per hypothesis to achieve close to the specified performance.

One effect that we noted was that when the image basis used in the correspondence was constrained to come from the model points in the image, the false positive rate tended to be higher. We suspect this may be related to the effect of model symmetry, since incorrect correspondences that happened to project the model to a position that was relatively symmetric to the actual pose would often pass the threshold. This event is more likely to occur when the features in the image basis come from the model.

In the next chapter we will discuss the implications of our findings to other existing recognition techniques.

Chapter 7

Implications for Recognition Algorithms

The error analysis we have presented applies not only to alignment, but to geometric hashing as well. We will briefly discuss the original geometric hashing algorithm and explain the modifications that are required to be able to apply our error analysis. Finally we will discuss possible applications and extensions of our work.

Because we used affine coordinates as the model representation and limited the Gaussian weight disk to a radius of $2\sigma_e$, our method for threshold and performance prediction applies equally well to both alignment and geometric hashing, provided the original geometric hashing algorithm is modified to take error into account in a particular way. First we will discuss the original algorithm, and then we will describe the modifications required for the error analysis to apply.

7.1 Geometric Hashing

The geometric hashing method was introduced by Lamdan, Schwartz and Wolfson in [LSW87], and Hummel and Wolfson in [HW88]. The algorithm consists of two stages, a preprocessing stage in which a lookup table is created, and a run time stage in which small groups of image are features used to access the lookup table for potential matches.

In the preprocessing stage, the hash table is constructed as follows: Every ordered triple of model points is used as a basis, and the affine coordinates (α, β) of all other model points are computed with respect to each basis. Thus, if $\vec{m}_0, \vec{m}_1$ and $\vec{m}_2$ are basis points, then we represent any other feature point by

$$\mathbf{m}_i = \mathbf{m}_0 + \alpha_i(\mathbf{m}_1 - \mathbf{m}_0) + \beta_i(\mathbf{m}_2 - \mathbf{m}_0)$$

The basis $(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2)$ is entered into the hash table at each (α_i, β_i) location. Intuitively, the invariance of the affine coordinates of a model with respect to 3 of its own points as basis is being used to "precompute" all possible views of the model in an image. The precise algorithm is:

```
• for every ordered model triplet $B_k = (\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}_2)$,
    – for every other model point $m_j$
        (i) find coordinates $\mathbf{m}_j = (\alpha_j, \beta_j)$ with respect to basis $B_k$
        (ii) enter basis $B_k$ at location $(\alpha_j, \beta_j)$ in the hash table.
```

The running time for this stage is $O(m^4)$, where m=number of model points.

At recognition time, the image is processed to extract $2D$ feature points. Every image triple is then taken as a basis, and the affine coordinates of all other image points are computed with respect to the basis to index into the hash table and "vote" for all bases found there. We will use the term "random image basis" to refer to an image basis which contains at least one point not arising from the model. Intuitively we are searching for any three image points which come from the model, and using the hash table to verify hypothesized triples of image points as instances of model points. Such an image triple will yield a large number of votes for its corresponding model basis. The precise algorithm is:

```
• for every unordered image triplet $(i_0, i_1, i_2)$
    (a) for every other image point $i_j$
        (i) find coordinates $i_j = (\alpha_j, \beta_j)$ with respect to basis $(i_0, i_1, i_2)$
        (ii) Index into the hash table at location $(\alpha_j, \beta_j)$ and increment a histogram count for all bases found there.
    (b) If the weight of the vote for any basis $B_k$ is greater than some threshold $\theta$, stop and output the correspondence between triple $(i_0, i_1, i_2)$ and basis $B_k$ as a correct hypothesis.
```

A single pass of the algorithm corresponds to testing a single image basis for a correspondence to any model basis. In some versions of the algorithm, the hypothesis that is output subsequently undergoes a verification stage before being accepted as correct. The termination condition for accepting a correspondence of bases (and hence a pose of the object) and the implied probability of true detection and false alarm are exactly the issues that our error analysis addresses.

7.2 Comparison of Error Analyses

The first error analysis of the geometric hashing technique was done by Grimson, Huttenlocher and Jacobs [GHJ91]. They used a uniform model for sensor error, and concluded several things: first, that when sensor error is taken into account and a particular image triplet is chosen in the recognition stage, then the regions in the hash table that are consistent with the sensed position of any fourth image point (step (ii)) are ellipses whose center and axes are dependent on configuration of the image basis. Thus, the error regions themselves cannot be taken into account at the preprocessing stage, but rather must be computed in step (ii) of the recognition stage, and all model bases in the region incremented.

Second, they derived the probability that a single random image basis would match any model basis as follows: Suppose the probability that a single random image point will be in a region consistent with any model point is μ on average. Let us fix the model basis that we are interested in. The probability that a single image point will fall in any region consistent with this particular model basis is

$$p = 1 - (1-\mu)^m$$

since there are m places in the index table where this basis appears, and the image point must avoid all of them. However, there are n image points, so the probability that this particular model basis gets at least k votes is

$$w_k = 1 - \sum_{i=0}^{k-1} \binom{n}{k} p^k (1-p)^{n-k}$$

This is the probability that a single random image basis matches a fixed model basis. There are $m(m-1)(m-2)$ bases in the hash table (we will use $m_{\langle 3 \rangle}$ to denote this expression), so probability that this image basis will contribute at least k votes to *any* model basis is

$$\begin{aligned} & 1 - \mathrm{P}\{\text{image basis contributes} \geq k \text{ to no model basis}\} \\ = \; & 1 - (1-w_k)^{m_{\langle 3 \rangle}} \\ = \; & 1 - \left[\sum_{i=0}^{k-1} \binom{n}{k} p^k (1-p)^{n-k} \right]^{m_{\langle 3 \rangle}} \end{aligned}$$

This is the probability that in a single pass through the recognition stage of the geometric hashing algorithm, the image basis being tested will find a match of at least size k at random. They presented an analogous analysis for alignment, which is identical except the roles of n and m are switched. Thus they conclude that the probability of an overall false positive was greater for the geometric hashing case than for alignment, because $n > m$ prevails rather generally.

The difference in the positions of n and m in their analysis was based on the assumption that alignment counts at most one image point per model disk whereas geometric hashing counts all image points that appear in the model disk. This is equivalent to the distinction between Schemes 1 and 3 that was discussed in Chapter 5. However, the geometric hashing scheme can be easily modified to use either collection method by keeping track of whether a point has already been collected from that particular location; in fact, this is the method generally used. The alignment method can also easily use either collection scheme. Therefore, the probability of error is equivalent for either method when using comparable collection schemes.

Interestingly enough, we have shown the opposite of the conclusion of [GHJ91] with regard to the probability of error as a function of collection method. We concluded in Chapter 5 that the performance of Scheme 1 (counting all image points that fall in a weight disk) is better than that of Scheme 3 (at most one image point per weight disk); this is because even when the clutter is very high, the expected number of points falling inside a correctly hypothesized weight disk is always greater than the expected number of points falling inside a random weight disk regardless of the clutter level. Therefore, the expected value of the sum of points appearing in the disk will always be higher if the disk is correct. The other collection scheme saturates with noise once there is a high probability that at least one image point will appear per weight disk. This finding does not contradict the analysis in [GHJ91] since the weighting scheme used in that analysis was based on a uniform model for sensor noise.

To apply our error analysis we would have to project entire error ellipses into the hash table as described in [GHJ91], but in the Gaussian error model case, the ellipses would be smaller and we would increment weights for model bases instead of votes. Now we can appreciate why it was important to limit the Gaussian weight disk to a finite size. If the distribution were unbounded, we would have to go through the entire table and contribute some small weight to every basis, thus changing the run time of the original geometric hashing algorithm. Applying our method to this domain results in being able to derive triples of (θ, P_F, P_D) for the termination step of the geometric hashing algorithm (step (b)).

Furthermore, we can easily calculate the probability that a particular image basis will match *any* model basis, as was done in [GHJ91]. We already mentioned that the geometric hashing technique can be considered a "filtering" step which provides candidate model to image basis correspondences to some more expensive verification step. Then the technique would be considered to break down once the number of matches it offers up is too high.

Suppose we are willing to verify (by alignment or any other verification technique) all bases that pass our threshold, as long as there are $\leq k$ of them. Then, an overall false positive is the combined event that the three image points being tested do not arise from the model, yet more than k model bases "look good". An overall true positive is the combined event that the three image points do arise from the model, that $\leq k$ model bases pass the test, and of these, one of them is the correct one. We will call these combined events Ω_F and Ω_D, and

$$\begin{aligned} \mathrm{P}\{\Omega_F\} &= 1 - \textstyle\sum_{i=0}^{k} \binom{m_{\langle 3 \rangle}}{i} P_F^i (1 - P_F)^{m_{\langle 3 \rangle} - i} \\ \mathrm{P}\{\Omega_D\} &= P_D * \textstyle\sum_{i=0}^{k-1} \binom{m_{\langle 3 \rangle}}{i} P_F^i (1 - P_F)^{m_{\langle 3 \rangle} - i} \end{aligned}$$

7.3 Summary

The error analysis and threshold prediction method that we derived in this thesis are directly translatable to the geometric hashing algorithm, provided the latter is modified to take Gaussian error into account.

Chapter 8

Conclusion

The kind of analysis we have done in this thesis is crucial in order to build robust systems. One way of making a system more robust is to incorporate several different sensors or processing modules for the same feature. Each sensor or module has a weighting attached to its output which is related to its reliability. The process of integrating the information from all the sensors is dependent on correctly assessing the reliability of the sensors under what conditions.

What we have done in this thesis is to begin to assess the reliability of a vision sensor for a fixed algorithm. We expect that such error analyses will be necessary for integrating vision into any automated system, with or without multiple sensors.

8.1 Extensions

To conclude, we have demonstrated an error analysis for alignment or geometric hashing, and a companion method that predicts triples of *(threshold,* P_F*,* P_D*)* for a fixed number of model and image features. The method as presented is limited to planar models solely due to our ignorance, as yet, of an analytical expression for the size of a projected error disk as a function of sensor error in the $3D$ case. We showed the method working well first in the domain of simulated models and images, and subsequently in real images.

The application to real images was problematic in that our model for clutter was not accurate, and the disparity initially resulted in an unexpectedly high false alarm probability. We modified the basic method to take the non-uniformity of the clutter distribution into account, and subsequently demonstrated a method to dynamically reset the threshold used for accepting a hypothesis to maintain a fixed probability of detection or false alarm.

There are several areas for extending the initial work presented in this thesis:

- The most significant improvement would result from a more sophisticated model for clutter. Though we assumed a uniform model, which is a standard in the field, this model vastly underpredicts the negative effects of clutter on the probability of false positives.

- When a model is symmetric, often a pose which aligns the model in the image along an axis of symmetry will appear very good. Information about model symmetry could be incorporated into the method such that when we are testing such a case, the threshold would be raised accordingly.
- Though we used simple $2D$ features for the analysis, there is nothing inherent in the method preventing extensions to more complex features.
- The method can easily be tailored to a particular model database by representing the score distributions for correct hypotheses for each model, and using these distributions instead of the generic distributions to improve performance for the given database.

It is our belief that model based vision algorithms will not be useful unless and until we can know how much faith we can place in the interpretations given by them. The work presented in this thesis is a step towards addressing the question.

Appendix A

Glossary, Conventions and Formulas

The notation that I use in the thesis generally follows the conventions used in Van Trees [VT68] except where no confusion would result by abbreviation.

A.1 Conventions

Random variables are denoted by capital letters, and their values are generally denoted by the same letter in lower case.

Vectors (such as $2D$ image and model features) are denoted in bold-face lower case, and matrices are denoted in bold-face upper case.

$\mathrm{P}\{\cdot\}$ denotes the probability of the event in parentheses.

$F_X(x)$ is the probability that random variable X is less than or equal to x.

$f_X(x)$ is the probability density function of the random variable X.

A vertical line in an expression means "given that". So for example, $f_X(x \mid E)$ is the probability density function of X given event E. If the event being conditioned upon is that the value of a random variable $A = a$, then we write $f_{X|A}(x \mid a)$

$\mathrm{E}[\cdot]$ is the expected value of the random variable in brackets.

$\mathrm{Var}\,(\cdot)$ is the variance of the random variable in parentheses. $\mathrm{Cov}\,(X, Y)$ is the covariance between the random variables X and Y.

$X \sim N(m, \sigma^2)$ denotes that the random variable X is normally distributed with mean m and variance σ^2.

A.2 Symbols and Constants

$m+3$	number of model features
$n+3$	number of image (sensor) features
$\mathbf{m}_i$	ith model feature
$\mathbf{s}_i$	ith image feature
θ	threshold used in recognition algorithm
P_F	probability of false positive (false detection)
P_D	probability of true positive (true detection)
σ_0	standard deviation of sensor noise for the Gaussian error model. This is considered to be a constant whose value must be determined empirically.
ϵ_0	radius of sensor noise for the uniform error model.
A	image area
ϕ,ψ	angles
i, j, k	indices
H	the event "three feature correspondence is correct"
$\overline{H}$	the event "three feature correspondence is incorrect".
M	the event "image feature arises from model"
$\overline{M}$	the event "image feature does not arise from model", or alternatively, "image feature is clutter"

A.3 Random Variables

ρ_e ranges over the values of the expression

$$\alpha^2 + \beta^2 + (1 - \alpha - \beta)^2 + 1$$

for all model points, where (α, β) is a model point's affine coordinates in the coordinate frame established by the three model points used in the correspondence.

σ_e describes the standard deviation of projected Gaussian error disks and is defined as

$$\sigma_0 \rho_e.$$

$V_M(m, n, c, \sigma_0)$ describes the weight or score distribution of a single point arising from the model for the specified values of m, n, c and σ_0.

$V_{\overline{M}}(m, n, c, \sigma_0)$ describes the weight or score distribution of a single clutter point for the specified values of m, n, c and σ_0.

$W_H(m, n, c, \sigma_0)$ describes the weight or score distribution of an entire correct hypothesis for the specified values of m, n, c and σ_0.

$W_{\overline{H}}(m, n, c, \sigma_0)$ describes the weight or score distribution of an entire incorrect hypothesis for the specified values of m, n, c and σ_0.

For simplicity, the last four random variables will be referred to as V_M, $V_{\overline{M}}$, W_H and $W_{\overline{H}}$, since the values of the parameters m, n, c and σ_0 are constant for the scope of the discussion.

A.4 Functions of Random Variables

Let X be a random variable with probability density $f_X(x)$, and let Y be a random variable which arises as a function of X, specifically, $Y = g(X)$. Assuming the function g is monotonically increasing and differentiable, the probability density function for the random variable Y is given as follows [BRB89]:

$$f_Y(y) = \frac{f_X(g^{-1}(y))}{g'(g^{-1}(y))} \tag{A.1}$$

For a monotonically decreasing function, the formula is the negation of the above expression.

Suppose X and Y are jointly random variables. Then the mean and variance of X can be found by conditioning on the value of Y [Ros84]:

$$\mathrm{E}[X] = \mathrm{E}[\mathrm{E}[X \mid Y]] \tag{A.2}$$

$$\mathrm{Var}(X) = \mathrm{E}[\mathrm{Var}(X \mid Y)] + \mathrm{Var}(\mathrm{E}[X \mid Y]) \tag{A.3}$$

Bibliography

[AB86] Haruo Asada and Michael Brady. The Curvature Primal Sketch. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1), 1986.

[AF86] N. Ayache and O. Faugeras. HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1), 1986.

[Agg93] J.K. Aggarwal, editor. *Multisensor Fusion for Computer Vision*, volume 99 of *NATO ASI Series F: Computer and Systems Sciences.* Springer-Verlag, Berlin, 1993.

[Bai84] Henry S. Baird. *Model Based Image Matching Using Location.* ACM Distinguished Dissertation. MIT Press, Cambridge, 1984.

[Bal81] D.H. Ballard. Generalizing the Hough Transform to Detect Arbitrary Shapes. *Pattern Recognition*, 13(2), 1981.

[BC82] R. Bolles and R. Cain. Recognizing and Locating Partially Visible Objects: The Local Feature-Focus Method. *International Journal of Robotics Research*, 1(3), 1982.

[Bha84] Bir Bhanu. Representation and Shape Matching of 3D Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(3), 1984.

[BQFW78] R.C. Bolles, L.H. Quam, M.A. Fischler, and H.C. Wolf. The SRI Road Expert: Image-To-Database Correspondence. In *Proc. of the Image Understanding Workshop*, pages 163–174, November 1978.

[BRB89] K. Baclawski, G.C. Rota, and S. Billey. An Introduction to the Theory of Probability. Second Preliminary Edition, MIT, 1989.

[Bre93] Thomas M. Breuel. Higher Order Statistics in Object Recognition. In *Proc. Computer Vision and Pattern Recognition*, page 707, June 1993.

[Cas90] Todd A. Cass. Feature Matching for Object Localization in the Presence of Uncertainty. Technical Report 1133, MIT AI Laboratory, 1990.

[CHS90] M. Costa, R.M. Haralick, and L.G. Shapiro. Optimal Affine-Invariant Point Matching. In *Proc. 6th Israel Conf. Art. Intell.*, pages 35–61, 1990.

[CJ91] David T. Clemens and David W. Jacobs. Space and Time Bounds on Indexing 3D Models from 2D Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10), October 1991.

[Cle91] David T. Clemens. Region-Based Feature Interpretation for Recognizing 3D Models in 2D Images. Technical Report 1307, MIT AI Laboratory, 1991.

[DW88] Hugh F. Durrant-Whyte. Sensor Models and Multisensor Integration. *International Journal of Robotics Research*, 7(6), 1988.

[Ett87] Gil J. Ettinger. Hierarchical Object Recognition Using Libraries of Parameterized Model Sub-Parts. Technical Report 963, MIT AI Laboratory, 1987.

[FB80] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Technical Report 213, SRI International, March 1980.

[GH90] W. Eric L. Grimson and Daniel P. Huttenlocher. On the Sensitivity of the Hough Transform for Object Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(3), 1990.

[GHA92] W.E.L. Grimson, D.P. Huttenlocher, and T.D. Alter. Recognizing 3D Objects from 2D Images: An Error Analysis. Technical Report 1362, MIT AI Lab, 1992.

[GHJ91] W.E.L. Grimson, D.P. Huttenlocher, and D.W. Jacobs. Affine Matching with Bounded Sensor Error: A Study of Geometric Hashing and Alignment. Technical Report 1250, MIT AI Lab, 1991.

[GLP84] W. Eric L. Grimson and Tomas Lozano-Perez. Model-Based Recognition and Localization from Sparse Range or Tactile Data. *International Journal of Robotics Research*, 3(3), 1984.

[GLP87] W.E.L. Grimson and T. Lozano-Perez. Localizing Overlapping Parts by Searching the Interpretation Tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4), 1987.

[Gri90] W. Eric L. Grimson. *Object Recognition By Computer: The Role of Geometric Constraints*, chapter 10. MIT Press Series in Artificial Intelligence. MIT Press, Cambridge, 1990.

[HU87] D.P. Huttenlocher and S. Ullman. Object Recognition Using Alignment. In *1st Int. Conf. Comp. Vis.*, pages 102–111, 1987.

[Hut88] Daniel P. Huttenlocher. Three-Dimensional Recognition of Solid Objects from a Two-Dimensional Image. Technical Report 1045, MIT AI Lab, 1988.

[HW88] R. Hummel and H. Wolfson. Affine Invariant Matching. In *DARPA Image Understanding Workshop*, 1988.

[Jac88] David W. Jacobs. The Use of Grouping in Visual Object Recognition. Technical Report 1023, MIT AI Lab, 1988.

[Jac92] David W. Jacobs. Recognizing 3D Objects Using 2D Images. Technical Report 1416, MIT AI Lab, 1992.

[KJ86] Thomas Knoll and Ramesh Jain. Recognizing Partially Visible Objects Using Feature Indexed Hypotheses. *IEEE Journal of Robotics and Automation*, RA-2(1), 1986.

[Low86] David Lowe. Three Dimensional Object Recognition from Single Two Dimensional Images. Technical Report 202, N.Y.U. Dept. of Comp. Sci, 1986.

[LSW87] Y. Lamdan, J.T. Schwartz, and H.J. Wolfson. On Recognition of 3-D Objects from 2-D Images. Technical Report 322, N.Y.U. Dept. Comp. Sci., 1987.

[LW91] Yehezkel Lamdan and Haim J. Wolfson. On the Error Analysis of 'Geometric Hashing'. In *IEEE Comp. Vis. Patt. Recog.*, 1991.

[Ols93] Clark Olson. Fast Alignment Using Probabilistic Indexing. In *IEEE Comp. Vis. Patt. Recog.*, 1993.

[RH91] I. Rigoutsos and R. Hummel. Robust Similarity Invariant Matching in the Presence of Noise. In *8th Israeli Conf. Art. Intell. Comp. Vis.*, 1991.

[Ros84] Sheldon Ross. *A First Course in Probability.* Macmillan, 1984.

[Sar92] K.B. Sarachik. Limitations of Geometric Hashing in the Presence of Gaussian Noise. Technical Report 1395, MIT AI Lab, 1992.

[SU88] A. Shashua and S. Ullman. Structural Saliency: The Detection of Globally Salient Structures Using A Locally Connected Network. In *Int. Conf. Comp. Vision*, pages 321–327, Tampa, FL, December 1988. Also in MIT AI-memo 1061.

[Swa90] Michael Swain. Color Indexing. Technical Report 360, University of Rochester, November 1990.

[TM87] D.W. Thompson and J.L. Mundy. Three Dimensional Model Matching from and Unconstrained Viewpoint. In *IEEE Robotics and Automation*, 1987.

[Tsa93] Frank Chee-Da Tsai. A Probabilistic Approach to Geometric Hashing Using Line Features. Technical Report 640, Robotics Research Laboratory, N.Y.U., June 1993.

[UB89] Shimon Ullman and Ronen Basri. Recognition by Linear Combinations of Models. Technical Report 1152, MIT AI Laboratory, August 1989.

[VT68] Harry L. Van Trees. *Detection, Estimation, and Modulation Theory*, volume I: Detection, Estimation, and Linear Modulation Theory, chapter 2. John Wiley and Sons, 1968.

[Wel91] William Wells. MAP Model Matching. In *IEEE Comp. Vis. Patt. Recog.*, pages 486–492, 1991.

[Wel92] William Wells. Statistical Object Recognition. Technical Report 1395, MIT AI Lab, 1992.