Statistical Approach to Affine Invariant Matching with Line Features [1]

Frank Chee-Da Tsai

Robotics Research Laboratory,
Courant Institute of Mathematical Sciences,
New York University
715 Broadway, 12FL
New York, N.Y. 10003

Abstract

One of the most important goals of computer vision research is *object recognition*. Currently, most object recognition algorithms assume reliable quality of image segmentation, which in practice is often not the case. This report examines the combination of the *Hough Transform* with a variation of *Geometric Hashing* as a technique for model-based object recognition in *seriously degraded* single intensity images.

There is recently much focus on the performance analysis of geometric hashing. However, to our knowledge, all of them are focusing on applying the paradigm to point features and show that the technique is sensitive to noise. There is as yet no exploration of line features. In this report, we use lines as the primitive features to compute the *geometric invariant*s for fast indexing into the geometric hash table containing the pre-processed model information. In addition, we analytically determine the effect of perturbations of line parameters on the computed *invariant* for the case where models are allowed to undergo affine transformation.

We have implemented the system with a series of experiments on polygonal objects, which are modeled by lines. It shows that the technique is noise resistant and suitable in an environment containing many occlusions.

[1] The author would like to thank Prof. Jack Schwartz and Isidore Rigoutsos for generous discussion.

Contents

1 Introduction **3**

2 Line Encoding **4**
2.1 Creating the Canonical Coordinate System 4
2.2 Computing The Invariant 5

3 The Effect of Noise **6**
3.1 Noise Model 6
3.2 The Spread Function of the Invariant 7

4 Geometric Hashing with Weighted Voting **9**

5 The Algorithm and Its Complexity Analysis **10**
5.1 The Preprocessing Stage 10
5.2 The Recognition Stage 11
5.3 The Disambiguation Stage 12

6 Implementation and Experiments **13**

7 Discussion **19**

8 Future Work **20**

9 Appendix **21**
9.1 Appendix 1 21
9.2 Appendix 2 23

1 Introduction

Object recognition is a central task in computer vision. This humanlike visual capability would enable machines to sense their environment in their field of view, to understand what is being sensed and to take an appropriate action as desirable.

This task can be complicated by partial overlapping of the objects in the scene (e.g. piles of industrial parts) or possible existence of occluding (e.g. flakes generated during the milling process) or unfamiliar objects.

Research (see surveys [Binford 82]) has indicated that a *model-based* approach to object recognition can be very effective in overcoming occlusion, complication and inadequate or erroneous low level processing. Most commercial vision systems are model-based ones, in which recognition involves matching the input image with a set of predefined models of objects. The goal in this approach is to *precompile* a description of each of a known set of objects, then to use these object models to recognize in an image each instance of an object and to specify its position and orientation relative to the viewer. This allows the power of a model-based recognition system to be determined by the richness of the models as well as the ability of the system to correlate the models with the sensed data.

Geometric Hashing[LamSchWol 88] is a model-based approach based on *precompiling* redundant *transformation-invariant* information of models in a hash table *off-line* and using the *invariant*s computed from the scene for fast indexing into the hash table to hypothesize possible matches between scene objects and models during recognition.

In a highly degraded image, extracting of the locations of point features is inherently error-prone and the analysis of geometric hashing on point sets [GrimHutt 90] shows that it is very sensitive to noise. We apply the *Hough* transform to extract line features, which appear in most industrial parts, and work directly on lines as the primitive features to compute the *geometric invariant*s.

In this report, we consider the problem of 2-D (or, *flat* 3-D) object recognition under affine transformation. In particular, we are to work in the environment containing many occlusions. For affine transformation, it is quite a successful approximation to perspective transformation under suitable situation (see [LamSchWol 88]) for most industrial applications. For serious occlusion, this realm is as yet to be explored. The state of the art of most model-based recognition system is restricted among others to: (1) small model base (5 to 10 objects); (2) typical simple scene (high signal noise ratio); (3) slight level of occlusion allowed; (4) no exploitation of parallelism; (5) *segmentation is the limiting factor.*

This report concentrates on the recognition stage of the whole visual recognition process, while assumes minimal requirement on the segmentation — only positional information of edgels is assumed to be available, which can be obtained from any low level edge detection algorithm, say [BoieCox 87].

The method presented combines the techniques of the *Hough transform*, *geometric hashing* [LamWol 88] and weighted voting by *Baysian* reasoning. It is divided into 3 stages: *preprocessing*, *recognition* and *disambiguation.*

The preprocessing stage is executed *off-line.* It is applied to the model features, the invariants (under affine transformation) of which are computed and stored in the so-called *geometric hash table.*

The recognition stage attempts to find instances of models (which may be only partially visible) in the scene by finding matches of minimal scene feature subsets and minimal model feature subsets. This process is speeded up by advantage of the fast indexing into the geometric hash table prepared in the preprocessing stage.

The disambiguation stage breaking ties of alleged model instances sharing the same scene features. An alleged model with stronger evidence support will win the competition.

The domain of the problem involves objects which can be modeled by a collection of lines. Compared with point features (which are used in [LamWol 88]), line features can generally be extracted with greater accuracy since more image points are involved in such features. In particular, good results are obtained for parameter extraction using the *Hough* transform [Risse 89]. (In fact, we have enhanced Risse's method with a technique of perceptual grouping and obtained even better result.)

This report is organized as follows. In section 2, we give a closed-form formula for the invariant encoded by an ordered quadruplet of lines, using their line parameters (θ, r)s. Section 3 analyzes the perturbation of the computed invariant from the perturbations of the lines giving rise to the invariant. A *Gaussian* noise model is assumed (in fact, also shown by previous experiments done on the detection of lines by the *Hough* transform). This analysis provides a theoretically justified measure of closeness of two invariants, which lets us pick out the most promising candidate set of model features from a pool of alleged sets. Section 4 describes the index selectivity improvement on the geometric hashing methodology by weighted voting. Section 5 details the 3 stages of our algorithms. Section 6 describes its implementation and provides experimental results. Finally, section 7 discusses the number of random probes needed and the possibility of a parallel realization of the algorithm.

2 Line Encoding

Given an ordered triplet of 2-D lines $(\theta_i, r_i)^t_{i=1,2,3}$ in general position, any fourth line $(\theta, r)^t$ can be *encoded* in terms of these 3 lines in an *affine-invariant* way.

One way of encoding is to find a transformation $\mathbf{T}$, which maps $(\theta_i, r_i)^t_{i=1,2,3}$ to a *canonical* coordinate system, say $(0,0)^t$, $(\frac{\pi}{2}, 0)^t$ and $(\frac{\pi}{4}, \frac{1}{\sqrt{2}})^t$ (or $x = 0$, $y = 0$ and $x + y = 1$), then apply $\mathbf{T}$ to $(\theta, r)^t$[HeckBoll 91].

2.1 Creating the Canonical Coordinate System

Let a line be parameterized as $\cos\theta\, x + \sin\theta\, y - r = 0$ and let us represent the 3 *basis lines* by their parameters $\mathbf{a}_1$, $\mathbf{a}_2$ and $\mathbf{a}_3$, s.t.

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

which are to be mapped by $\mathbf{T}'$ to

$$\mathbf{e}_1 = (\cos 0, \sin 0, 0)^t = (1, 0, 0)^t,$$

$$\mathbf{e}_2 = (\cos\frac{\pi}{2}, \sin\frac{\pi}{2}, 0)^t = (0,1,0)^t,$$

$$\mathbf{e}_3 = (\cos\frac{\pi}{4}, \sin\frac{\pi}{4}, \frac{-1}{\sqrt{2}})^t = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}})^t.$$

Since $ax + by + c = 0$ and $\lambda ax + \lambda by + \lambda c = 0$, $\lambda \in \mathbf{R}$, represent the same line, we have

$$\lambda_1 \mathbf{T}'\mathbf{a}_1 = \mathbf{e}_1,$$

$$\lambda_2 \mathbf{T}'\mathbf{a}_2 = \mathbf{e}_2,$$

$$\lambda_3 \mathbf{T}'\mathbf{a}_3 = \mathbf{e}_3,$$

where λ_1, λ_2 and λ_3 are constants.

Solving the above system of linear equations, we obtain $\mathbf{T}' =$

$$\begin{pmatrix} \csc(\theta_1-\theta_2)\csc(\theta_2-\theta_3)\sin\theta_2 D & -\csc(\theta_1-\theta_2)\csc(\theta_2-\theta_3)\cos\theta_2 D & 0 \\ -\csc(\theta_1-\theta_2)\csc(\theta_3-\theta_1)\sin\theta_1 D & \csc(\theta_1-\theta_2)\csc(\theta_3-\theta_1)\cos\theta_1 D & 0 \\ \csc(\theta_1-\theta_2)(r_2\sin\theta_1 - r_1\sin\theta_2) & \csc(\theta_1-\theta_2)(r_1\cos\theta_2 - r_2\cos\theta_1) & 1 \end{pmatrix},$$

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

2.2 Computing The Invariant

Define a mapping $\mathbf{F}$

$$\mathbf{F} : \mathbf{R}^2 \mapsto \mathbf{R}^3$$

such that

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

Then, $\mathbf{F}^{-1}$ exists, if we restrict the domain of θ to be in $[0, \pi)$. In fact,

$$\mathbf{F}^{-1}((a,b,c)^t) = (tan^{-1}(b/a), -c/\sqrt{a^2+b^2})^t$$

Note that $\mathbf{F}^{-1}(\lambda\mathbf{v}) = \mathbf{F}^{-1}(\mathbf{v})$, where $\lambda \in \mathbf{R}$ and $\mathbf{v} \in \mathbf{R}^3$.

The invariant of $(\theta, r)^t$ is given by $\mathbf{T}((\theta, r)^t) = \mathbf{F}^{-1} \circ \mathbf{T}' \circ \mathbf{F}((\theta, r)^t)$ or

$$\mathbf{F}^{-1}(\lambda\mathbf{T}'(\cos\theta, \sin\theta, -r)^t) = \mathbf{F}^{-1}(\mathbf{T}'(\cos\theta, \sin\theta, -r)^t).$$

Computing and expanding $\mathbf{T}'(\cos\theta, \sin\theta, -r)^t$, we get

$$\begin{pmatrix} \csc(\theta_1-\theta_2)\csc(\theta_2-\theta_3)\sin(\theta_2-\theta) \\ (r_3\sin(\theta_1-\theta_2) + r_2\sin(\theta_3-\theta_1) + r_1\sin(\theta_2-\theta_3)) \\ \\ \csc(\theta_1-\theta_2)\csc(\theta_1-\theta_3)\sin(\theta_1-\theta) \\ (r_3\sin(\theta_1-\theta_2) + r_2\sin(\theta_3-\theta_1) + r_1\sin(\theta_2-\theta_3)) \\ \\ r_2\csc(\theta_1-\theta_2)\sin(\theta_1-\theta) - r_1\csc(\theta_1-\theta_2)\sin(\theta_2-\theta) - r \end{pmatrix}$$

Applying $\mathbf{F}^{-1}$ to the above vector, we obtain $(\theta^{'}, r^{'})^t$ with

$$\theta^{'} = \tan^{-1}(\csc(\theta_1 - \theta_3)\sin(\theta_1 - \theta)/\csc(\theta_2 - \theta_3)\sin(\theta_2 - \theta)) \tag{1}$$

and

$$r^{'} = \frac{1}{\tau}(r_1 \csc(\theta_1 - \theta_2)\sin(\theta_2 - \theta) - r_2 \csc(\theta_1 - \theta_2)\sin(\theta_1 - \theta) + r) \tag{2}$$

where

$$\tau = \frac{\csc(\theta_1 - \theta_2)(r_3 \sin(\theta_1 - \theta_2) + r_2 \sin(\theta_3 - \theta_1) + r_1 \sin(\theta_2 - \theta_3))}{\sqrt{\csc^2(\theta_1 - \theta_3)\sin^2(\theta_1 - \theta) + \csc^2(\theta_2 - \theta_3)\sin^2(\theta_2 - \theta)}}$$

3 The Effect of Noise

In an environment with serious noise and occlusion, line features are often detected by the *Hough* transform. Detected line parameters (θ, r)s usually deviate slightly from the true values. In the imaging process, the positions of the endpoints of a segment are usually randomly perturbed. As long as this segment is *not* too short, this induces only *small* perturbation on the line parameters of the segment. It is reasonable for us to assume (and in fact, we have done extensive simulations covering a range of line lengths, line orientations and positions under increasing noise levels) that detected line parameters differ from the "true" values by a small perturbation having a *Gaussian* distribution.

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

3.1 Noise Model

We make the following moderate assumption that the measured values of a set of line parameters (θ, r)s are

1. statistically independent,
2. distributed according to a *Gaussian* distribution centered at the true value of the parameter, and
3. with a fixed covariance $\Sigma = \begin{pmatrix} \sigma_\theta^2 & \sigma_{\theta r} \\ \sigma_{\theta r} & \sigma_r^2 \end{pmatrix}$.

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

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

centered around (θ, r).

3.2 The Spread Function of the Invariant

The invariant $(\theta^{'}, r^{'})$ is given by (1) and (2). Introducing *Gaussian* perturbation in the line parameters (θ, r), $(\theta_i, r_i)_{i=1,2,3}$, (1) and (2) can be rewritten as follows:

$$\begin{aligned}\theta^{'} + \delta\theta^{'} &= \tan^{-1}(\csc((\theta_1 - \theta_3) + (\delta\theta_1 - \delta\theta_3))\sin((\theta_1 - \theta) + (\delta\theta_1 - \delta\theta))/ \\ &\quad \csc((\theta_2 - \theta_3) + (\delta\theta_2 - \delta\theta_3))\sin((\theta_2 - \theta) + (\delta\theta_2 - \delta\theta))) \qquad (3)\end{aligned}$$

and

$$\begin{aligned}r^{'} + \delta r^{'} &= \frac{1}{\tau^{'}}((r_1 + \delta r_1)\csc((\theta_1 - \theta_2) + (\delta\theta_1 - \delta\theta_2))\sin((\theta_2 - \theta) + (\delta\theta_2 - \delta\theta)) \\ &\quad -(r_2 + \delta r_2)\csc((\theta_1 - \theta_2) + (\delta\theta_1 - \delta\theta_2))\sin((\theta_1 - \theta) + (\delta\theta_1 - \delta\theta)) \\ &\quad +(r + \delta r)) \qquad (4)\end{aligned}$$

where

$$\begin{aligned}\tau^{'} &= \csc((\theta_1 - \theta_2) + (\delta\theta_1 - \delta\theta_2)) \\ &\quad ((r_3 + \delta r_3)\sin((\theta_1 - \theta_2) + (\delta\theta_1 - \delta\theta_2)) + \\ &\quad (r_2 + \delta r_2)\sin((\theta_3 - \theta_1) + (\delta\theta_3 - \delta\theta_1)) + \\ &\quad (r_1 + \delta r_1)\sin((\theta_2 - \theta_3) + (\delta\theta_2 - \delta\theta_3))) \\ &\quad \sqrt{\csc^2((\theta_1 - \theta_3) + (\delta\theta_1 - \delta\theta_3))\sin^2((\theta_1 - \theta) + (\delta\theta_1 - \delta\theta)) + \\ &\quad \overline{\csc^2((\theta_2 - \theta_3) + (\delta\theta_2 - \delta\theta_3))\sin^2((\theta_2 - \theta) + (\delta\theta_2 - \delta\theta))}}\end{aligned}$$

Expanding (3) and (4) in *Maclaurin* series and ignoring second and higher order perturbation terms and then subtracting (1) and (2) from them, we have

$$\begin{aligned}\delta\theta^{'} &= \frac{\partial\theta^{'}}{\partial\theta}\delta\theta + \sum_{i=1}^{3}\frac{\partial\theta^{'}}{\partial\theta_i}\delta\theta_i \\ &= c_{11}\delta\theta + c_{13}\delta\theta_1 + c_{15}\delta\theta_2 + c_{17}\delta\theta_3 \qquad (5) \\ \delta r^{'} &= \frac{\partial r^{'}}{\partial\theta}\delta\theta + \frac{\partial r^{'}}{\partial r}\delta r + \sum_{i=1}^{3}\frac{\partial r^{'}}{\partial\theta_i}\delta\theta_i + \sum_{i=1}^{3}\frac{\partial r^{'}}{\partial r_i}\delta r_i \\ &= c_{21}\delta\theta + c_{22}\delta r + c_{23}\delta\theta_1 + c_{24}\delta r_1 + \\ &\quad c_{25}\delta\theta_2 + c_{26}\delta r_2 + c_{27}\delta\theta_3 + c_{28}\delta r_3 \qquad (6)\end{aligned}$$

where

$$\begin{aligned}c_{11} &= D[\sin(\theta_1 - \theta_2)] \\ c_{13} &= D[-\csc(\theta_1 - \theta_3)\sin(\theta_2 - \theta)\sin(\theta_3 - \theta)] \\ c_{15} &= D[\csc(\theta_2 - \theta_3)\sin(\theta_1 - \theta)\sin(\theta_3 - \theta] \\ c_{17} &= D[-\csc(\theta_1 - \theta_3)\csc(\theta_2 - \theta_3)\sin(\theta_1 - \theta_2)\sin(\theta_1 - \theta)\sin(\theta_2 - \theta)]\end{aligned}$$

$$
\begin{aligned}
c_{21} &= [C\sin(\theta_1-\theta_2)(\cos(\theta_1-\theta)\sin(\theta_1-\theta)\csc^2(\theta_1-\theta_3)+ \\
&\qquad \cos(\theta_2-\theta)\sin(\theta_2-\theta)\csc^2(\theta_2-\theta_3))/B^2 - \\
&\quad (r_1\cos(\theta_2-\theta)-r_2\cos(\theta_1-\theta))]/AB \\
c_{22} &= [\sin(\theta_1-\theta_2)]/AB \\
c_{23} &= [C(\cos(\theta_1-\theta_2)+\sin(\theta_1-\theta_2)(r_2\cos(\theta_1-\theta_3)-r_3\cos(\theta_1-\theta_2))/A + \\
&\qquad \sin(\theta_1-\theta_2)\csc^2(\theta_1-\theta_3)\sin(\theta_1-\theta) \\
&\qquad (\cot(\theta_1-\theta_3)\sin(\theta_1-\theta)-\cos(\theta_1-\theta))/B^2) - \\
&\quad (r_2\cos(\theta_1-\theta)+\cot(\theta_1-\theta_2)(r_1\sin(\theta_2-\theta)-r_2\sin(\theta_1-\theta)))]/AB \\
c_{24} &= [-C\sin(\theta_1-\theta_2)\sin(\theta_2-\theta_3)/A+\sin(\theta_2-\theta)]/AB \\
c_{25} &= [C(-\cos(\theta_1-\theta_2)-\sin(\theta_1-\theta_2)(r_1\cos(\theta_2-\theta_3)-r_3\cos(\theta_1-\theta_2))/A + \\
&\qquad \sin(\theta_1-\theta_2)\csc^2(\theta_2-\theta_3)\sin(\theta_2-\theta) \\
&\qquad (\cot(\theta_2-\theta_3)\sin(\theta_2-\theta)-\cos(\theta_2-\theta))/B^2) + \\
&\quad (r_1\cos(\theta_2-\theta)+\cot(\theta_1-\theta_2)(r_1\sin(\theta_2-\theta)-r_2\sin(\theta_1-\theta)))]/AB \\
c_{26} &= [C\sin(\theta_1-\theta_2)\sin(\theta_1-\theta_3)/A-\sin(\theta_1-\theta)]/AB \\
c_{27} &= [C\sin(\theta_1-\theta_2)((r_1\cos(\theta_2-\theta_3)-r_2\cos(\theta_1-\theta_3))/A - \\
&\qquad (\cot(\theta_1-\theta_3)\csc^2(\theta_1-\theta_3)\sin^2(\theta_1-\theta)+ \\
&\qquad \cot(\theta_2-\theta_3)\csc^2(\theta_2-\theta_3)\sin^2(\theta_2-\theta))/B^2)]/AB \\
c_{28} &= [-C\sin^2(\theta_1-\theta_2)/A]/AB \\
A &= r_3\sin(\theta_1-\theta_2)+r_2\sin(\theta_3-\theta_1)+r_1\sin(\theta_2-\theta_3) \\
B &= \sqrt{\csc^2(\theta_1-\theta_3)\sin^2(\theta_1-\theta_4)+\csc^2(\theta_2-\theta_3)\sin^2(\theta_2-\theta_4)} \\
C &= r_1\csc(\theta_1-\theta_2)\sin(\theta_2-\theta)-r_2\csc(\theta_1-\theta_2)\sin(\theta_1-\theta)+r \\
D &= \csc(\theta_1-\theta_3)\csc(\theta_2-\theta_3)/[\csc^2(\theta_1-\theta_3)\sin^2(\theta_1-\theta)+\csc^2(\theta_2-\theta_3)\sin^2(\theta_2-\theta)]
\end{aligned}
$$

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

Substituting $\Delta\Theta^{'}$ and $\Delta R^{'}$ for $\delta\theta^{'}$ and $\delta r^{'}$ in (5) and (6) respectively, we have

$$
\begin{aligned}
\Delta\Theta^{'} &= c_{11}\Delta\Theta+c_{13}\Delta\Theta_1+c_{15}\Delta\Theta_2+c_{17}\Delta\Theta_3 \\
\Delta R^{'} &= c_{11}\Delta\Theta+c_{12}\Delta R+c_{13}\Delta\Theta_1+c_{14}\Delta R_1+ \\
&\qquad c_{15}\Delta\Theta_2+c_{16}\Delta R_2+c_{17}\Delta\Theta_3+c_{18}\Delta R_3
\end{aligned}
$$

Since $(\Delta\Theta,\Delta R)$ and $(\Delta\Theta_i,\Delta R_i)_{i=1,2,3}$ are *Gaussian* and independent, we may show that $p(\Delta\Theta',\Delta R')$ also has a *Gaussian* distribution with *p.d.f.*

$$
p(\Delta\Theta',\Delta R') = \frac{1}{2\pi\sqrt{|\Sigma'|}}exp[-\frac{1}{2}(\Delta\Theta',\Delta R')\Sigma'^{-1}(\Delta\Theta',\Delta R')^t] \tag{7}
$$

where

$$\Sigma' = \begin{pmatrix} (c_{11}^2 + c_{13}^2 + c_{15}^2 + c_{17}^2)\sigma_\theta^2 & (c_{11}c_{21} + c_{13}c_{23} + c_{15}c_{25} + c_{17}c_{27})\sigma_\theta^2 + \\ & (c_{11}c_{22} + c_{13}c_{24} + c_{15}c_{26} + c_{17}c_{28})\sigma_{\theta r} \\ (c_{11}c_{21} + c_{13}c_{23} + c_{15}c_{25} + c_{17}c_{27})\sigma_\theta^2 + & (c_{21}^2 + c_{23}^2 + c_{25}^2 + c_{27}^2)\sigma_\theta^2 + \\ (c_{11}c_{22} + c_{13}c_{24} + c_{15}c_{26} + c_{17}c_{28})\sigma_{\theta r} & (c_{22}^2 + c_{24}^2 + c_{26}^2 + c_{28}^2)\sigma_r^2 + \\ & 2(c_{21}c_{22} + c_{23}c_{24} + c_{25}c_{26} + c_{27}c_{28})\sigma_{\theta r} \end{pmatrix} \quad (8)$$

is the covariance matrix of $(\Delta\Theta', \Delta R')$. (For details, see Appendix 1.)

4 Geometric Hashing with Weighted Voting

Our approach builds upon the geometric hashing method. Briefly speaking, this method pre-computes and stores redundant, transformation-invariant model information in a *hash table*. Then during recognition, the same invariants are computed from scene features in the image and used as indexing keys to retrieve the possible matches with the models from the hash table. If a model in the hash table scores enough *hit*s, one hypothesizes the existence of an instance of the model in the image.

Drawbacks [GrimHutt 90] of the method lie in its sensitivity to image noise and to quantization noise introduced in constructing the hash table; also index selectivity problem and problems arising from non-uniform distribution of invariants in the hash table (see Fig-1).

Hash space density can be equalized by *rehashing*. A drawback here is that invariants mutually close to each other can be far apart after rehashing. This can be disadvantageous in the presence of noise.

Therefore, instead of rehashing, we adopt an approach similar to [RigouHum 91] by analytically determining the effect of noise on the computed invariants. This information is then used to obtain a formula which gives the probability for a "(model, basis, line)" combination to appear in the scene, given a scene invariant. This probability is to be used as the *weighted vote* a hash node, which represents a certain (model, basis, line), would receive when hit. Obviously, hash nodes *near* the "true" value of the invariant will get more weighted vote than those which are far away.

According to *Bayes*' theorem [Duda 73], given a scene invariant, inv_s, the probability for inv_s to be induced by a certain (model, basis, line) is

$$P((\text{model, basis, line})|inv_s) = \frac{p(inv_s|(\text{model, basis, line}))P((\text{model, basis, line}))}{p(inv_s)} \quad (9)$$

where

- $P((\text{model, basis, line}))$ is the *a priori* probability of a certain model being embedded in the scene and a certain triplet of lines of that model being selected as a basis. We will assume that each (model, basis, line) is equally likely.
- $p(inv_s)$ is the $p.d.f.$ of observing inv_s.
- $p(inv_s|(\text{model, basis, line}))$ is the conditional $p.d.f.$ of observing inv_s given that (model, basis, line) appears in the scene. Eq.s (7) and (8) provide us with this $p.d.f.$

In computing (9), P((model, basis, line)) can be dropped since we do not need to compute the very probability but the *relative* probability. $p(inv_s)$ can be obtained by table look-up and interpolation. The table can be approximated by simulation: using random number generator randomly generate 5 points over the image; using the lines formed by the first 3 points as the basis to encode the line formed by the 4-th and 5-th points; tallying 1 count in the entry the computed invariant hits; trying, say, 1 million such quintuplets, then each entry is replaced by the value count/1000000.

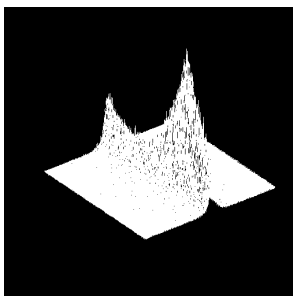

Fig-1. A density histogram of the hash table obtained by simulation. We see that there is a very high concentration of invariants (θ, r), with r values in $[-1..1]$ and a fast drop as one moves away from this area.

5 The Algorithm and Its Complexity Analysis

Our algorithm operates in 3 stages, the function of which have been explained in section 1. In the following, we sketch each stage, giving complexity analyses where available.

5.1 The Preprocessing Stage

Models are processed one by one. New models added to the model base can be processed and embedded into the hash table *independently*.

For each model M and for every 3 lines (sides), say l_1, l_2 and l_3 with parameters (θ_1, r_1), (θ_2, r_2) and (θ_3, r_3), of the model, we

1. By eq.s (1) and (2), compute the invariants, (θ', r')s, of all the rest lines w.r.t. the above 3 lines as the basis.

2. Use (θ', r'), which is invariant under affine transformation, to index the hash entry of the 2-D geometric hash table, where we record a "node" $(model_m, basis_b, side_s, inv_m, cov_{inv_m})$ consisting of

   - $model_m$, the model identifier;
   - $basis_b$, the basis identifier;
   - $line_l$, the identifier of the line being encoded;
   - inv_m, the invariant of $line_l$ w.r.t $basis_b$; and
   - cov_{inv_m}, the covariance matrix associated with inv_m, which provides the information about the *spread* of inv_m over the hash space and is given by eq. (8).

Note that all the permutations (up to 3!) of l_1, l_2 and l_3 have to be explored as the basis to be used to encode the other lines.

Also note that care has to be taken that all 3 lines used are feasible as a basis. If the triangle formed by the 3 lines are too small or too large, the invariants encoded by them will be numerically unstable and should be avoided.

The complexity of this stage is $O(l^4)$ per model, where l is the number of lines extracted from a model. Since this stage is executed *off-line*, its complexity is of little significance.

5.2 The Recognition Stage

First a collection of n lines $\cos\theta_i\, x + \sin\theta_i\, y = r_i$, $i = 1, .., n$ are detected by the *Hough* transform. (Note that usually more lines than necessary are detected to guarantee that no true lines are missing.)

A triplet of lines is then chosen randomly. If it is feasible as a basis, we do the following:

1. Compute the invariants, (θ', r')s, of all the rest lines w.r.t. the above triplet of lines as the basis by eq.s (1) and (2).

2. Each such computed invariant inv_s is used to index the 2-D hash table entries. (The position of each entry in fact represents a quantized coordinate (θ, r) of the hash table.)

3. An appropriate-sized area of neighboring entries are considered (theoretically, we might have to consider all the entries because the "spread" of the invariant extends to infinity; yet, practically, those too far away result in a large *Mahalanobis* distance and thus a negligible weighted vote.)

4. If the *Mahalanobis* distance between an entry considered and the entry indexed by inv_s lies above a threshold, ignore considering the entry; otherwise, retrieve the nodes in that entry, if any.

5. For every such node, $(model_m, basis_b, line_l, inv_m, cov_{inv_m})$, we compute the weighted vote by eq. (9) and credit it to $(model_m,\ basis_b)$. We also tag that $line_l$ has a possible match in the scene (for later verification, if necessary.)

6. The $(model_{m_i},\ basis_{b_j})$ with highest votes is a candidate model feature set, with minimum error rate of classification, appearing in the scene. Practically we may wish to consider more $(model_m,\ basis_b)$s whose votes are above a preset threshold.

7. For each such alleged model feature set, verify its existence in the scene by alignment:

   (a) Compute the affine transformation $\mathbf{T} = (\mathbf{A}, \mathbf{b})$ that transforms the model basis to match scene triplet. (see Appendix 2 for the formula.)

   (b) Superimpose the model onto the scene and verify the existence of those model lines (segments) which are tagged to have a possible match in step 5. (More precisely, verifying the existence of a scene line segment is by checking if the edgel density of the line segment is above some threshold.)

   (c) Compute a *quality measure* QM, defined as the ratio of the total length of all *existent* segments over the total length of all segments of the transformed model.

8. if QM is below a preset threshold (e.g. 0.5), that means the verification fails. Otherwise, vote QM for $|det(A)|$.

N randomly chosen triplets are probed as bases. N must be large enough for this statistical approach to work (i.e. to detect all the objects in the scene *simultaneously*.)

After all the N bases are tried, the *global* skewing factor, which is taken to be the $|det(A)|$ with the highest number of votes, is obtained. (We may show that all the 2-D objects *lying on the same plane* have the same skewing factor, if viewed from the same tilted camera, assuming affine approximation to perspective transformation.)

Those alleged model feature sets which voted for a wrong skewing factor are thrown away, while those which voted for the global skewing factor are the models which were hypothesized and have passed verification.

The complexity of this stage is $O(n \times c)+O(t)$ per probe, where n is the number of lines detected by the *Hough* transform; c is the size of the area of neighboring entries considered and t is the complexity of verifying the model.

5.3 The Disambiguation Stage

The disambiguation stage follows the recognition stage to further refine the result.

We now have a list of candidate model instances produced by the recognition stage. It is possible that a scene line (segment) is shared by many model instances, some of which are false alarms. We might wish to classify the shared line (segment) to uniquely belong to the model instance with stronger support.

One possible technique for disambiguation is *relaxation* [ZukHumRos 77]. However, *relaxation* is a complicated process and the following simple-minded method proves to work well.

Recall that each model instance is associated with a *quality measure*, QM, computed in the recognition stage and acting as a measurement of the strongness of evidence supporting its existence. This definition of QM favors long segments of a model instance and is similar to that used in [Ayache 86]. The justification is that long sides are usually less numerous and therefore more discriminant.

A *Z-buffer-Algorithm*-like [Roger 85] methodology is exercised. Here the QM acts as its counterpart of "depth" in the *Z-buffer-Algorithm*. Similarly, each candidate model instance is given a unique label or *id*. Two similar data structures, *z-buffer* and *frame buffer*, are maintained. This algorithm, after initializing z-buffer and frame buffer, processes in arbitrary order each candidate model instance as follows:

1. superimpose the candidate model instance onto the original image (by "original image", we mean the edgel image);

2. trace on the original image the visible segments of the model instance – for each edgel $P(x, y)$ traced,

   (a) compare the "depth" $z(x, y)$ (i.e. this model instance's QM) with the value stored in the z-buffer, $Zbuffer(x, y)$;

   (b) if $z(x, y) > Zbuffer(x, y)$, then write the *id* of this model instance to $Framebuffer(x, y)$ and replace $Zbuffer(x, y)$ with $z(x, y)$;

(c) if $z(x,y) = Zbuffer(x,y)$, put the *id* of this model instance into the *clash list* of the entry $Framebuffer(x,y)$;

(d) Otherwise, no action is taken.

After all candidate model instances have been processed, the algorithm re-processes each candidate model instance as follows:

1. trace the visible segments of the model instance *on the frame buffer* – for $Framebuffer(x,y)$ encountered, count only those with the same *id* of this model instance. In this way, the *density of edgels* along the line-segment can be obtained. If the density is below a preset threshold, that segment is deemed to be non-existent.

2. recompute the QM of this model instance. If it is less than a preset threshold (e.g. 0.5), remove it from the list of candidate model instances.

This stage costs very little running time. If k model instances appearing in the scene are detected in recognition stage, the complexity is $O(k \times t)$, where t is the complexity of tracing line segments of a model (boundary).

6 Implementation and Experiments

We have done a series of experiments on synthesized images containing polygonal objects, which are modeled by line features, from the model base consisting of 15 models as shown in Fig-2. Fig-3, 4, 5, and 6 are 4 examples among them. The perturbation of the lines detected by the *Hough* transform is obtained from statistics of an extensive simulations on images of different levels of degradation.

Several practical problems arise in the system implementation. Our formulae involve a lot of trigonometric evaluations. To speed up the process, a fine-grained look-up table of trigonometric functions was pre-computed. Also, the boundary of the hash table has to be treated specially, since we have to consider a neighborhood of a hash entry when it is retrieved. The θ-axis of the 2-D hash table should be viewed as *round-wrapped* with *flip*, since invariant (θ, r) is mathematically equivalent to $(\theta - \pi, -r)$. (e.g., $(179^o, 2)$ is equivalent to $(-1^o, -2)$, thus neighboring to, say, $(0^o, -2)$). Care should also be taken to reject *reflexion* case. Reflexion transformation is a subgroup of affine transformation. However, we would like to exclude it because a 2-D object, say a triangle with vertex-$1, 2, 3$ in clockwise sequence, preserves this sequence, even its shape being seriously skewed when viewed from a tilted camera. This problem is tackled by detecting the *orientation* of a basis (to be clockwise or counterclockwise) by *cross product*. Thus no such false match between scene basis and model basis will be hypothesized.

We also found from the experiments that when doing probing, if the probed triplet of lines happen to correspond to a basis of a certain model (i.e., a correct match), then this "(model, basis)" pair usually accumulates highest weighted vote among others. Even if sometimes it does not rank the highest, its weighted vote is, from our observation, still no less than 1/3 of the highest. Moreover, the high weighted vote of a false alarm (i.e. matching this scene basis to a wrong "(model, basis)") is usually resulted from the accumulation of only 1 or 2 scene invariants which happen to be very close to their corresponding model invariants in the hash table (thus getting high weighted vote.) Such

false alarms are usually rejected by verification since too few lines (segments) of the models are visible (thus resulting in a small quality measure QM.) We conclude that this weighted voting scheme is quite effective.

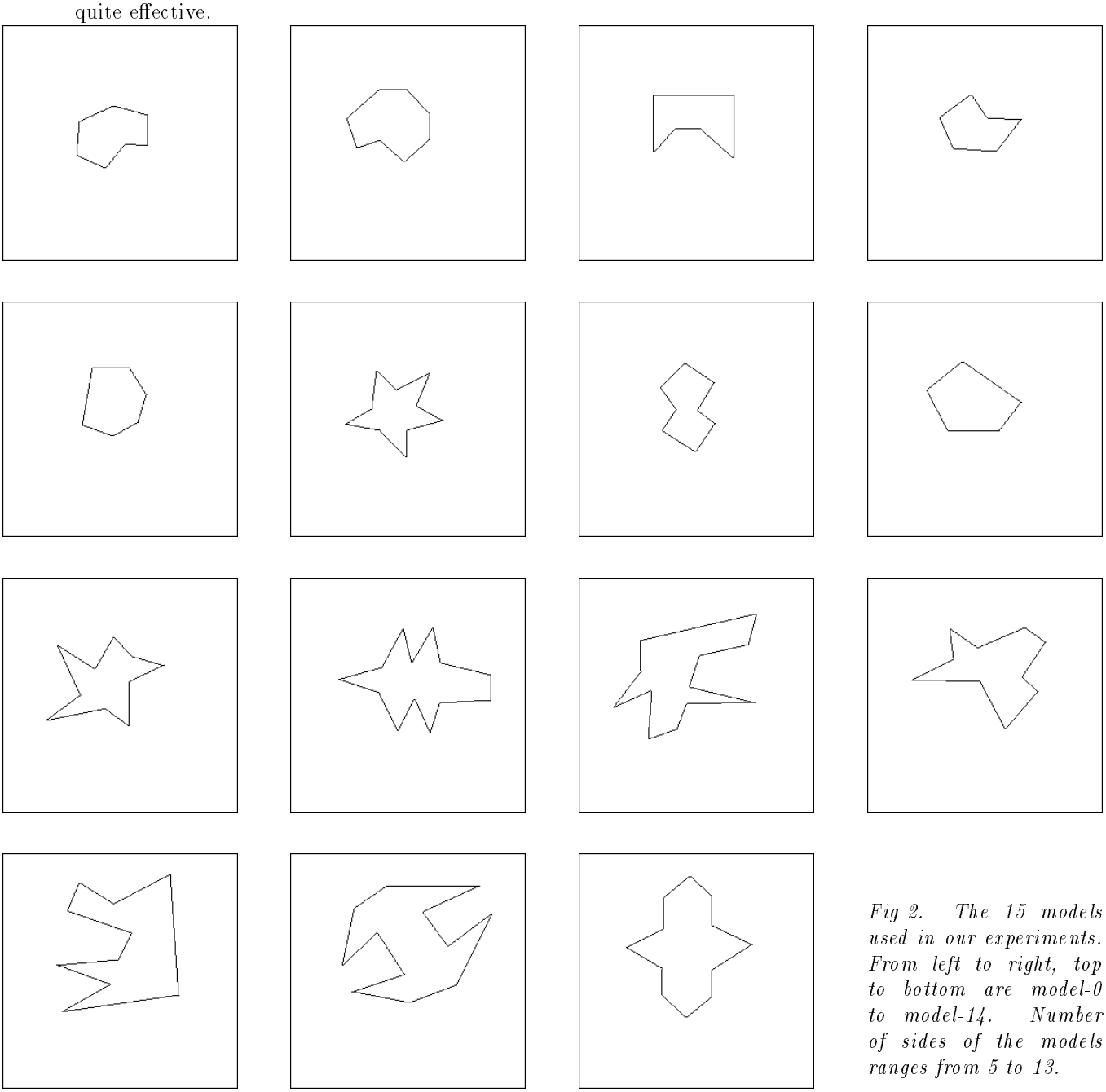

Fig-2. The 15 models used in our experiments. From left to right, top to bottom are model-0 to model-14. Number of sides of the models ranges from 5 to 13.

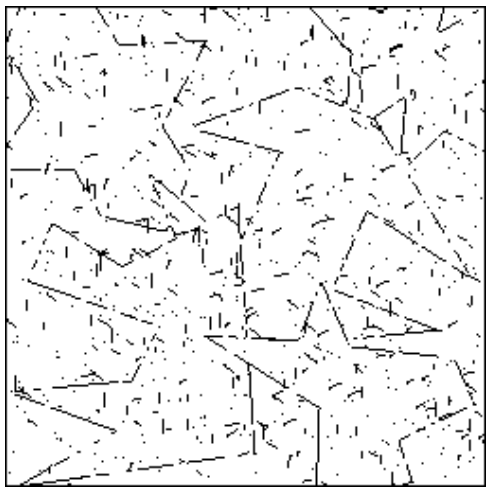

Fig-3(a)

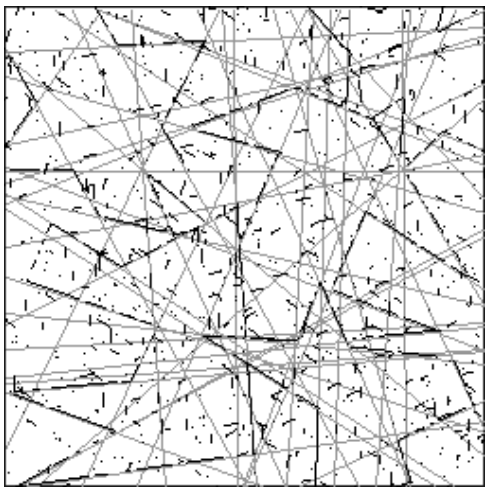

Fig-3(b)

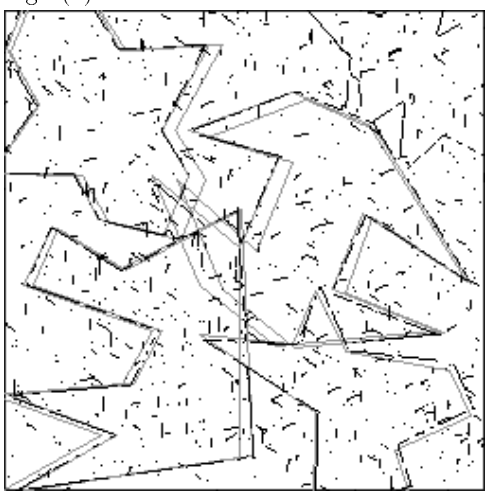

Fig-3(c)

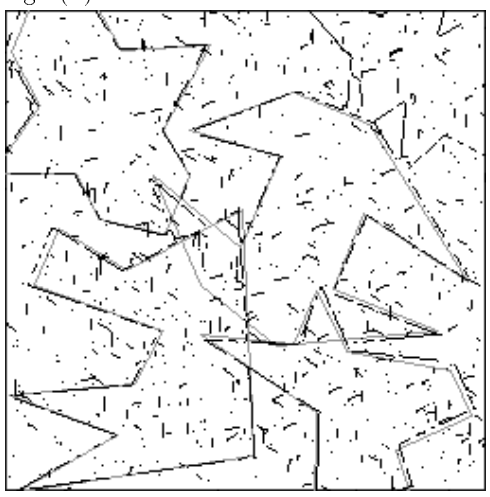

Fig-3(d)

Fig-3(a) shows a composite overlapping scene of model-9, 11, 12, 13 and 14. Fig-3(b) is the result of Hough *transform.* 50 *lines were detected. The covariance matrix used for the deviation of the detected lines from true lines was* $\Sigma = \begin{pmatrix} 0.00071 & 0.001 \\ 0.001 & 3.765 \end{pmatrix}$. *Fig-3(c) shows the model instances hypothesized and passing verification after the recognition stage.* 5000 *probes were tried. The threshold used for the existence of a segment was its pixel density over 0.6. Fig-3(d) shows the final result after* disambiguation. *Note that the instance of model-9 at the upper right corner was not detected. (We reject a hypothesized object instance if half of its boundary is invisible.)*

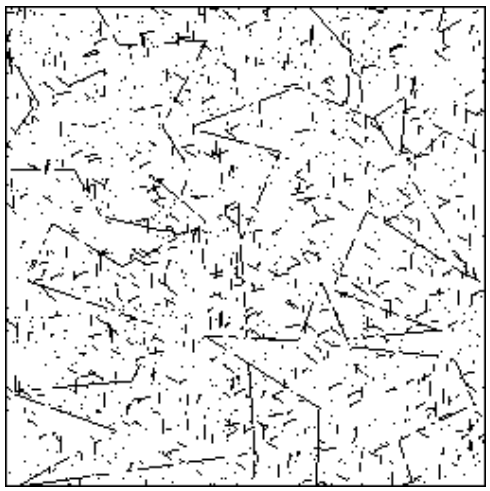

Fig-4(a)

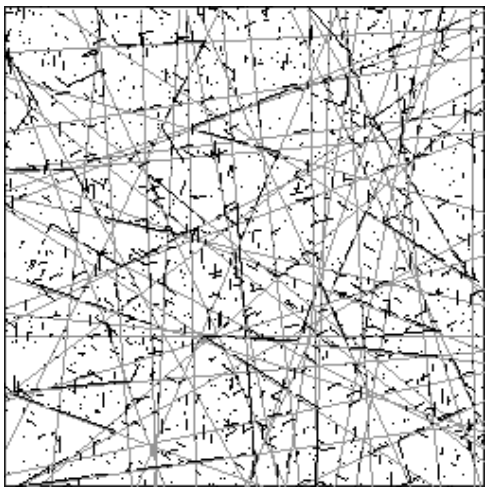

Fig-4(b)

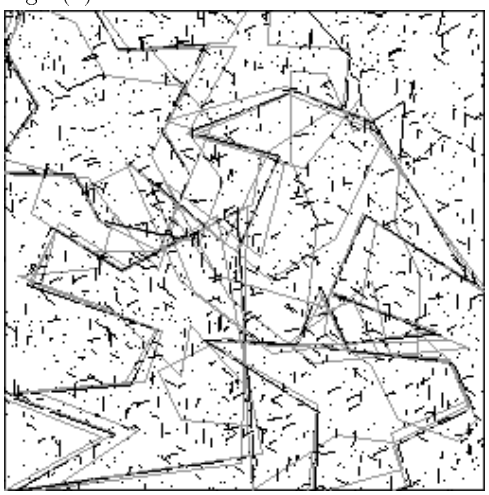

Fig-4(c)

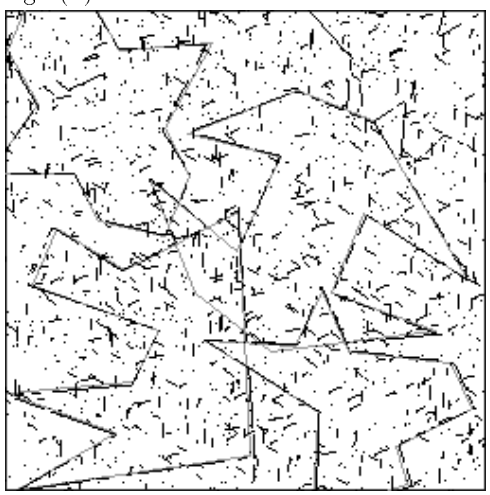

Fig-4(d)

Fig-4(a) is basically the same scene as Fig-3(a) but with more (roughly twice) segment and random dot noise. Fig-4(b) is the result of Hough *transform,* 55 *lines were detected. The covariance matrix used for the deviation of the detected lines from true lines was* $\Sigma = \begin{pmatrix} 0.00145 & -0.01061 \\ -0.01061 & 4.506 \end{pmatrix}$. *Fig-4(c) shows the model instances hypothesized and passing verification after the recognition stage. We can see that more false alarms occurred than in Fig-3(c).* 5000 *probes were tried. The threshold used for the existence of a segment was its pixel density over 0.6. Fig-4(d) shows the final result after* disambiguation. *Note that the instance of model-9 at the upper right corner was not detected as in Fig-3(d).*

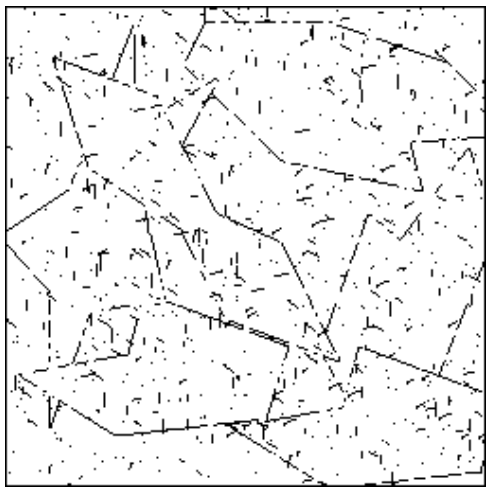

Fig-5(a)

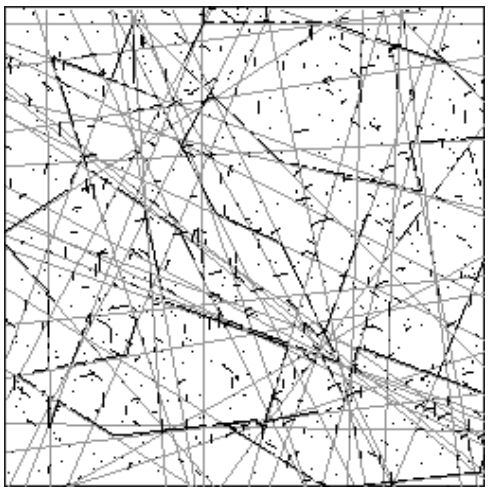

Fig-5(b)

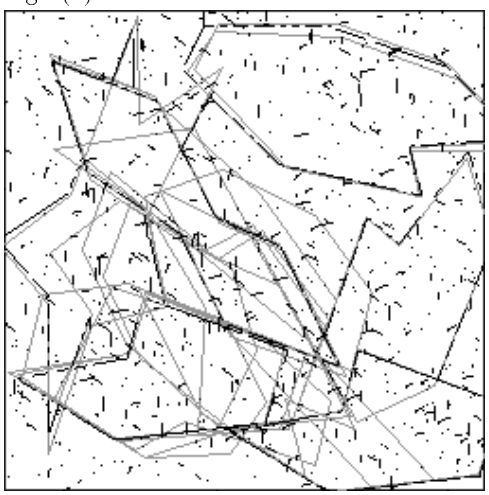

Fig-5(c)

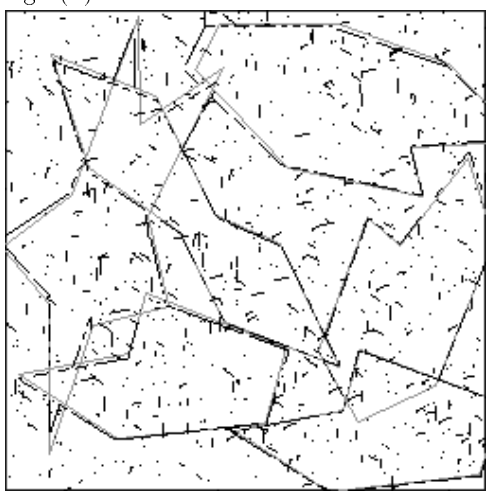

Fig-5(d)

Fig-5(a) shows a composite overlapping scene of model-0, 1, 3 and 5 (note: model-3 appears twice), which were significantly skewed. *Fig-5(b) is the result of* Hough *transform,* 55 *lines were detected. The covariance matrix used for the deviation of the detected lines from true lines is* $\Sigma = \begin{pmatrix} 0.00071 & 0.001 \\ 0.001 & 3.765 \end{pmatrix}$. *Fig-5(c) shows the model instances hypothesized and passing verification after the recognition stage.* 5000 *probes were tried. The threshold used for the existence of a segment was its pixel density over 0.5. We may see that many false alarms occurred with this low threshold. Fig-5(d) shows the final result after* disambiguation.

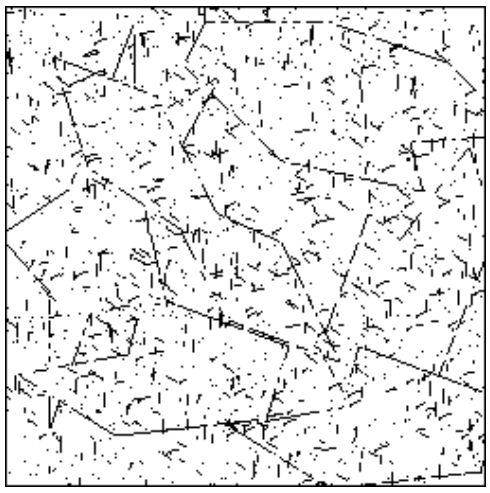

Fig-6(a)

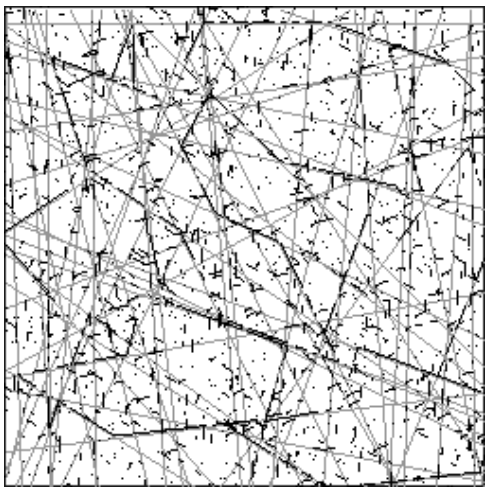

Fig-6(b)

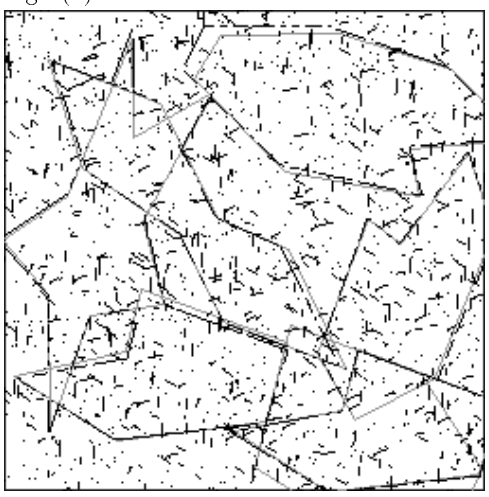

Fig-6(c)

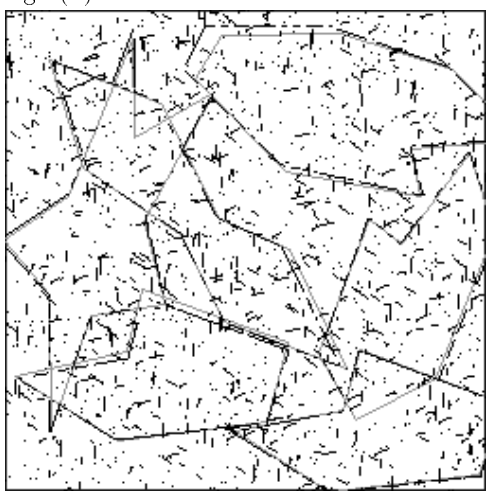

Fig-6(d)

Fig-6(a) is basically the same scene as Fig-5(a) but with more (roughly twice) segment and random dot noise. Fig-6(b) is the result of Hough *transform,* 60 *lines were detected. The covariance matrix used for the deviation of the detected lines from true lines is* $\Sigma = \begin{pmatrix} 0.00145 & -0.01061 \\ -0.01061 & 4.506 \end{pmatrix}$. *Fig-6(c) shows the model instances hypothesized and passing verification after the recognition stage.* 5000 *probes were tried. The threshold used for the existence of a segment was its pixel density over 0.7. With this high threshold, after recognition stage, only one false alarm occurred (at the right lower corner of the picture.) Fig-6(d) shows the final result after* disambiguation.

7 Discussion

The method described above does not assume any *grouping* of features, which is expected to greatly expedite the recognition stage (at least for scene basis selection). Lowe [Lowe 87] first explicitly discussed the importance of grouping to recognition. However, if reliable segmentation can not be available, intelligent grouping seems difficult.

As we pointed out in the recognition stage that a large number N of triplets have to be probed to detect all the objects. The following is a probability analysis of this operation, giving the order of magnitude of N where available.

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

The probability of choosing a scene triplet which happens to correspond to a basis of a specific model is

$$p = (\frac{\frac{n_1}{k}}{n})(\frac{\frac{n_1}{k}-1}{n-1})(\frac{\frac{n_1}{k}-2}{n-2})$$

If we are to detect all k model objects at once, in best case we have to probe only k triplets from the scene. This happens when each time a triplet is probed, it happens to correspond to a basis of a certain model and next time another triplet is probed, it corresponds to a basis of another model. However, the probability for this to happen is p^k, which is obviously small.

The probability of detecting exactly i *different models* after t trials (i.e. among the t trials, $t = i' + i'', i' \geq i$ triplets cover bases of each of the i models and i'' triplets correspond to no bases of any model) is

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

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

$$t(t-1)\cdots(t-i+1) \geq \epsilon/p^i \tag{10}$$

Thus,

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

The probability of at least j *different model*s are detected is

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

Let us consider some practical case to get an idea of the order of the magnitude of N.

For the special case when $k = 1$, i.e., there is only 1 model object embedded in the scene, the probability of choosing a model basis is

$$p = (\frac{n_1}{n})(\frac{n_1-1}{n-1})(\frac{n_1-2}{n-2})$$

- when half of the lines in the scene are spurious, $p = (n-4)/8(n-1)$;

- when one third of the lines in the scene are spurious, $p = 4(2n-3)(n-3)/27(n-1)(n-2)$;
- when there is no spurious lines in the scene, $p = 1$.

In a simple scene with only one model object embedded with $n_1 = 10$,

- when $n = 20$ with 10 spurious lines, $p = 0.105$. We need roughly try 10 probes.
- when $n = 15$ with 5 spurious lines, $p = 0.264$. We need roughly try 4 probes.

From this observation, we know that as long as there is only 1 (or few) object(s) in the scene, it is easy to probe a feasible basis. Thus, a correct model feature can be easily indexed, even when model base contains a lot of models. (Note that number of models in the model base is independent of probing a scene basis.)

For a typical scene when $k = 5$ (i.e. 5 models appearing in the scene) and $n_1 = 50$ (i.e. averagely each model object has 10 lines appearing in the scene)

- when $n = 100$ with 50 spurious lines, we have $p = 0.000742$;
- when $n = 75$ with 25 spurious lines, we have $p = 0.001778$;
- when $n = 50$ without spurious lines, we have $p = 0.00612$;

Note that p is just the probability of choosing a triplet belonging to *a specific* model appearing in the scene. If we want to detect all $k = 5$ models in t trials with probability ≥ 0.9, t will be bounded from buttom by 1322, 350 and 162 respectively, accroding to eq. (10).

In practice, we have to try more probes than theoretically predicted, because: (1) some triplets, though corresponding to some model bases, are *bad* (e.g., too small, too big or too much perturbed) bases and do not result in stable transformation that transforms model to properly fit scene object to pass verification; (2) some triplets, though corresponding to none of model bases, happen to result in false alarms which even pass the verification due to high noise. (However, we point out here that statistically, false alarms of case (2) disperse their votes for different skewing factors, while correct matches will accumulate their votes for the global skewing factor.)

It is obvious that the recognition stage can be trivially parallelized by assigning each basis to a *processing element* in a parallel computer, since each basis can be processed *independently*.

Parallel realization of image processing algorithms has become more and more popular with the advance of hardware techniques and the decrease of cost [Rosenfeld 87].

A possible candidate of parallel computer is the *Connection Machine*, which is equipped with $16K - 64K$ processors (for model CM-2). When the number of processors simultaneously needed exceeded the maximum number of physical processors in the machine, the machine can operate in a *virtual processor* mode. In this mode, a single processor emulates the work of several virtual processors, by serializing operations in time and partitioning the local memory associated with that processor.

8 Future Work

Since the initial experiments have been successful, we will create more models and add them to the model base in the future (this requires more memory and disk space). Experiments on real imagery is currently under way.

The transformation between a model object and its scene instance is uniquely determined by the correspondence of their bases (corresponding pair of triplets of lines). By transforming and superimposing the model onto the scene, we often find that despite the basis lines perfectly fit each other, other lines do not mutually fit quite well. A scene object can have more than one basis being probed. Theoretically, the affine transformations derived from different basis correspondences should be the same. However, they are often slightly different, because noisy computation of the scene triplet of lines. Knowledge of additional matching lines may help us improving the accuracy of the computed transformation. We will have to formulate criteria of minimizing matching errors between sets of lines.

Another interesting aspect of the research direction is to investigate the methodology's applicability to recognizing 3-D objects from 2-D images, assuming the affine approximation to perspective transformation. We will have to deal with the problem of the reduced dimension of the image space compared with the model space.

9 Appendix

9.1 Appendix 1

In the following, we derive the closed-form formula of the perturbation of the invariant from the perturbations of the quadruplet of lines giving rise to this invariant.

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

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

Corollary: Let $\mathbf{Y}^T = (Y_1, Y_2)$ s.t. $\mathbf{Y} = \mathbf{CX}$, where $\mathbf{X}^T = (X_1, X_2, ..., X_n)$ and $\mathbf{C} = \begin{pmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & c_{2n} \end{pmatrix}$, a real $2 \times n$ matrix. Then the random variable $\mathbf{Y}$ is $N(\mathbf{Cu}, \mathbf{C\Sigma C}^T)$.

$<$ *proof* $>$ The *moment-generating function* of the distribution $\mathbf{Y}$ is given by

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

Apply the above theorem, we have

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

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

Let $(\Delta\Theta, \Delta R)$ and $(\Delta\Theta_i, \Delta R_i)_{i=1,2,3}$ be *Gaussian* stochastic variables denoting the *perturbations* of (θ, r) and $(\theta_i, r_i)_{i=1,2,3}$ respectively.

Assuming that they have a fixed *covariance matrix*

$$\Sigma = \begin{pmatrix} \sigma_\theta^2 & \sigma_{\theta r} \\ \sigma_{\theta r} & \sigma_r^2 \end{pmatrix}$$

and also assuming that $(\Delta\Theta, \Delta R)$ and $(\Delta\Theta_i, \Delta R_i)_{i=1,2,3}$ are independent, we have that *p.d.f.* $p(\Delta\Theta, \Delta R, \Delta\Theta_1, \Delta R_1, \Delta\Theta_2, \Delta R_2, \Delta\Theta_3, \Delta R_3)$ is *Gaussian*. In particular,

$$\begin{aligned}
& p(\Delta\Theta, \Delta R, \Delta\Theta_1, \Delta R_1, \Delta\Theta_2, \Delta R_2, \Delta\Theta_3, \Delta R_3) \\
= \; & p(\Delta\Theta, \Delta R)\, p(\Delta\Theta_1, \Delta R_1)\, p(\Delta\Theta_2, \Delta R_2)\, p(\Delta\Theta_3, \Delta R_3) \\
= \; & (\frac{1}{2\pi\sqrt{|\Sigma|}})^4 exp[-\frac{1}{2}((\Delta\Theta, \Delta R)\Sigma^{-1}(\Delta\Theta, \Delta R)^T + (\Delta\Theta_1, \Delta R_1)\Sigma^{-1}(\Delta\Theta_1, \Delta R_1)^T + \\
& \qquad (\Delta\Theta_2, \Delta R_2)\Sigma^{-1}(\Delta\Theta_2, \Delta R_2)^T + (\Delta\Theta_3, \Delta R_3)\Sigma^{-1}(\Delta\Theta_3, \Delta R_3)^T)] \\
= \; & (\frac{1}{(2\pi)^4\sqrt{|\mathbf{\Sigma}'|}}) exp[-\frac{1}{2}\mathbf{t}^T \mathbf{\Sigma}'^{-1}\mathbf{t}]
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{t}^T &= (\Delta\Theta, \Delta R, \Delta\Theta_1, \Delta R_1, \Delta\Theta_2, \Delta R_2, \Delta\Theta_3, \Delta R_3) \\
\mathbf{\Sigma}' &= \begin{pmatrix}
\sigma_\theta^2 & \sigma_{\theta r} & 0 & 0 & 0 & 0 & 0 & 0 \\
\sigma_{\theta r} & \sigma_r^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \sigma_\theta^2 & \sigma_{\theta r} & 0 & 0 & 0 & 0 \\
0 & 0 & \sigma_{\theta r} & \sigma_r^2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \sigma_\theta^2 & \sigma_{\theta r} & 0 & 0 \\
0 & 0 & 0 & 0 & \sigma_{\theta r} & \sigma_r^2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \sigma_\theta^2 & \sigma_{\theta r} \\
0 & 0 & 0 & 0 & 0 & 0 & \sigma_{\theta r} & \sigma_r^2
\end{pmatrix}
\end{aligned}$$

Let $(\Delta\Theta^{'}, \Delta R^{'})$ be the stochastic variables denoting the *perturbations* of the computed invariant $(\theta^{'}, r^{'})$. Suppose

$$\begin{pmatrix} \Delta\Theta' \\ \Delta R' \end{pmatrix} = \mathbf{C}\mathbf{t}$$

where

$$\mathbf{C} = \begin{pmatrix}
c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & c_{16} & c_{17} & c_{18} \\
c_{21} & c_{22} & c_{23} & c_{24} & c_{25} & c_{26} & c_{27} & c_{28}
\end{pmatrix}$$

From the above **corollary**, we have that $(\Delta\Theta', \Delta R')$ also has a *Gaussian* distribution with *p.d.f.*

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

where

$$\mathbf{\Sigma}'' = \mathbf{C}\mathbf{\Sigma}'\mathbf{C}^T$$

$$= \begin{pmatrix} (c_{11}^2 + c_{13}^2 + c_{15}^2 + c_{17}^2)\sigma_\theta^2 & (c_{11}c_{21} + c_{13}c_{23} + c_{15}c_{25} + c_{17}c_{27})\sigma_\theta^2 + \\ (c_{12}^2 + c_{14}^2 + c_{16}^2 + c_{18}^2)\sigma_r^2 + & (c_{12}c_{22} + c_{14}c_{24} + c_{16}c_{26} + c_{18}c_{28})\sigma_r^2 + \\ 2(c_{11}c_{12} + c_{13}c_{14} + c_{15}c_{16} + c_{17}c_{18})\sigma_{\theta r} & (c_{11}c_{22} + c_{13}c_{24} + c_{15}c_{26} + c_{17}c_{28} + \\ & c_{12}c_{21} + c_{14}c_{23} + c_{16}c_{25} + c_{18}c_{27})\sigma_{\theta r} \\ \\ (c_{11}c_{21} + c_{13}c_{23} + c_{15}c_{25} + c_{17}c_{27})\sigma_\theta^2 + & (c_{21}^2 + c_{23}^2 + c_{25}^2 + c_{27}^2)\sigma_\theta^2 + \\ (c_{12}c_{22} + c_{14}c_{24} + c_{16}c_{26} + c_{18}c_{28})\sigma_r^2 + & (c_{22}^2 + c_{24}^2 + c_{26}^2 + c_{28}^2)\sigma_r^2 + \\ (c_{11}c_{22} + c_{13}c_{24} + c_{15}c_{26} + c_{17}c_{28} + & 2(c_{21}c_{22} + c_{23}c_{24} + c_{25}c_{26} + c_{27}c_{28})\sigma_{\theta r} \\ c_{12}c_{21} + c_{14}c_{23} + c_{16}c_{25} + c_{18}c_{27})\sigma_{\theta r} & \end{pmatrix}$$

is the covariance matrix of $(\Delta\Theta', \Delta R')$.

9.2 Appendix 2

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

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

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

where

$$\begin{aligned} a_{11} &= \cos\theta_3 \csc(\theta_1^{'} - \theta_2^{'}) \sin(\theta_1 - \theta_2)(r_1^{'} \sin\theta_2^{'} - r_2^{'} \sin\theta_1^{'}) + \\ &\quad \cos\theta_1 \csc(\theta_2^{'} - \theta_3^{'}) \sin(\theta_2 - \theta_3)(r_2^{'} \sin\theta_3^{'} - r_3^{'} \sin\theta_2^{'}) + \\ &\quad \cos\theta_2 \csc(\theta_3^{'} - \theta_1^{'}) \sin(\theta_3 - \theta_1)(r_3^{'} \sin\theta_1^{'} - r_1^{'} \sin\theta_3^{'}) \\ a_{21} &= \cos\theta_3 \csc(\theta_1^{'} - \theta_2^{'}) \sin(\theta_1 - \theta_2)(r_1^{'} \cos\theta_2^{'} - r_2^{'} \cos\theta_1^{'}) + \\ &\quad \cos\theta_1 \csc(\theta_2^{'} - \theta_3^{'}) \sin(\theta_2 - \theta_3)(r_2^{'} \cos\theta_3^{'} - r_3^{'} \cos\theta_2^{'}) + \\ &\quad \cos\theta_2 \csc(\theta_3^{'} - \theta_1^{'}) \sin(\theta_3 - \theta_1)(r_3^{'} \cos\theta_1^{'} - r_1^{'} \cos\theta_3^{'}) \\ a_{12} &= \sin\theta_3 \csc(\theta_1^{'} - \theta_2^{'}) \sin(\theta_1 - \theta_2)(r_1^{'} \sin\theta_2^{'} - r_2^{'} \sin\theta_1^{'}) + \\ &\quad \sin\theta_1 \csc(\theta_2^{'} - \theta_3^{'}) \sin(\theta_2 - \theta_3)(r_2^{'} \sin\theta_3^{'} - r_3^{'} \sin\theta_2^{'}) + \\ &\quad \sin\theta_2 \csc(\theta_3^{'} - \theta_1^{'}) \sin(\theta_3 - \theta_1)(r_3^{'} \sin\theta_1^{'} - r_1^{'} \sin\theta_3^{'}) \\ a_{22} &= \sin\theta_3 \csc(\theta_1^{'} - \theta_2^{'}) \sin(\theta_1 - \theta_2)(r_1^{'} \cos\theta_2^{'} - r_2^{'} \cos\theta_1^{'}) + \\ &\quad \sin\theta_1 \csc(\theta_2^{'} - \theta_3^{'}) \sin(\theta_2 - \theta_3)(r_2^{'} \cos\theta_3^{'} - r_3^{'} \cos\theta_2^{'}) + \\ &\quad \sin\theta_2 \csc(\theta_3^{'} - \theta_1^{'}) \sin(\theta_3 - \theta_1)(r_3^{'} \cos\theta_1^{'} - r_1^{'} \cos\theta_3^{'}) \end{aligned}$$

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

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

References

[Ayache 86] Ayache, N. and O. D. Faugeras, 1986. *HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects*, IEEE Trans. on Pattern Analysis and Machine Intelligence, vol.8(1),pp.44-54.

[Binford 82] Binfold, T. O., 1982 *Survey of Model-Based Image Analysis System*, Int. J. of Robotics Research,vol.1 (1) pp.18-64.

[BoieCox 87] Boie, R and I. Cox, 1987. *Two Dimensional Optimum Edge Recognition using Matched and Weiner Filter for Machine Vision*, Proceedings of the International Conference on Computer Vision, London, England, December 1987.

[Duda 73] Duda, R. O. and P. E. Hart, 1973. *Pattern Classification and Scene Analysis*, New York, Wiley.

[GrimHutt 90] Grimson, W. E.L. and D. P. Huttenlocher, 1990. *On the Sensitivity of Geometric Hashing*, Proc. of the 3rd Int. Conf. on Computer Vision, pp.334-338.

[HeckBoll 91] Hecker, Y. C. and R. M. Bolle, 1991. *Invariant Feature Matching in Parameter Space with Application to Line Features*, IBM Computer Science Research Report, January, 1991

[LamSchWol 88] Lamdan, Y., J. T. Schwartz and H. J. Wolfson, 1988. *Object Recognition by Affine Invariant Matching*, Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition,pp.335-344.

[LamWol 88] Lamdan, Y. and H. J. Wolfson, 1988. *Geometric Hashing: a General and Efficient Model-Based Recognition Scheme*, Proc. of the 2nd Int. Conf. on Computer Vision, pp.238-249.

[Lowe 87] Lowe, D., 1985. *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers.

[RigouHum 91] Rigoutsos, I. and R. Hummel, 1991. *Robust Similarity Invariant Matching in the Presence of Noise*, Proc. of the 8th Israeli Conf. on Artificial Intelligence and Computer Vision.

[Risse 89] Risse, T., 1989. *Hough Transform for Line Recognition: Complexity of Evidence Accumulation and Cluster Detection*, Computer Vision, Graphics and Image Processing, vol. 46, pp.327-345.

[Roger 85] Roger, D., 1985. *Procedural Elements for Computer Graphics*, McGraw-Hill Book Company. pp.265-272.

[Rosenfeld 87] Rosenfeld, A. 1987. *Computer Architecture for Machine Vision*, In H. Freeman (Ed.), Machine Vision – Algorithms, Architectures and Systems, Academic Press, Inc, pp.97-102.

[ZukHumRos 77] Zucker, S. W., R. Hummel and A. Rosenfeld, 1977. *An Application of Relaxation Labeling to Line and Curve Enhancement*, IEEE Trans. on Computers, vol.26, no.4, pp.394-403.