[19] I. Rigoutsos and R. A. Hummel. Implementation of geometric hashing on the connection machine. In *IEEE Workshop on Directions in Aut. CAD-Based Vision*, Maui, Hawaii, June 1991.

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

[21] F. C. D. Tsai. A Statistical Approach to Affine Invariant Matching with Line Features. Technical Report TR 621, Computer Science Dept., Courant Inst., NYU, 1992.

[22] F. C. D. Tsai. *A Probabilistic Approach to Geometric Hashing using Line Features*. PhD thesis, Department of Computer Science, Courant Inst. of Math., NYU, 1993.

[23] F. C. D. Tsai. Robust Affine Invariant Matching with Application to Line Features. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 393–399, 1993.

[24] F. C. D. Tsai. Using Line Invariants for Object Recognition by Geometric Hashing. Technical Report TR 625, Computer Science Dept., Courant Inst., NYU, 1993.

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

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

References

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

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

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

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

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

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

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

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

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

[10] R. Hummel and Rigoutsos I. Geometric Hashing as a Bayesian Maximum Likelihood Object Recognition Method. *Int. J. of Computer Vision*, 1993. Submitted.

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

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

[13] Y. Lamdan. *Object Recognition by Geometric Hashing.* PhD thesis, Department of Computer Science, Courant Inst. of Math., NYU, 1989.

[14] Y. Lamdan and H. J. Wolfson. Geometric Hashing: A General and Efficient Model-Based Recognition Scheme. In *Proc. of the IEEE Int. Conf. on Computer Vision*, pages 238–249, Tampa, Florida, December 1988.

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

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

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

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

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

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

Appendix-II

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

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

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

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

7 Discussion

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

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

The method described above does not assume any grouping of features, which is expected to greatly expedite the recognition stage (at least for scene basis selection). Lowe [16] first explicitly discussed the importance of grouping to recognition. However, if reliable segmentation can not be available, intelligent grouping seems difficult. We note that in probing, some basis selected, though corresponding to none of model bases, may happen to result in false alarms which even pass the verification due to high noise in the scene. However, we point out here that statistically false alarms of such case disperse their votes for different skewing factors, while correct matches will accumulate their votes for the global skewing factor.

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

Acknowledgement

The author would like to thank Professor Jacob T. Schwartz and Robert Hummel for generous discussions and Professor Jiawei Hong for suggestions on best least-squares match procedures.

Appendix-I

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

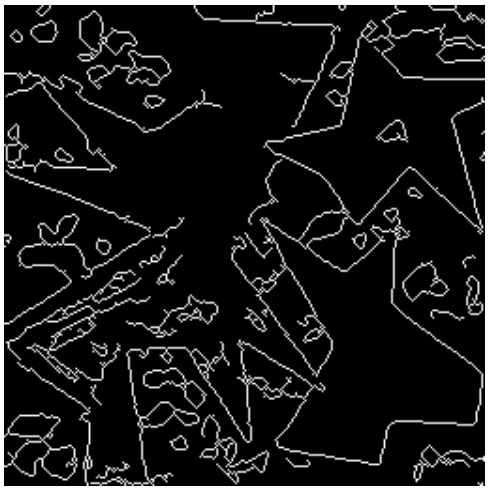
Let the first triplet of lines be $(\theta_i, r_i)^t_{i=1,2,3}$ and the second triplet of lines be $(\theta_i^{'}, r_i^{'})^t_{i=1,2,3}$. We have the closed-form formula for $\mathbf{T} = (\mathbf{A},\mathbf{b})$ as follows:

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

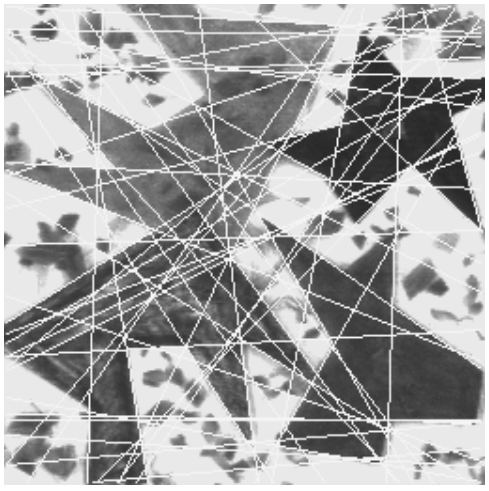
where

$$a_{11} = \cos\theta_3 \csc(\theta_1^{'} - \theta_2^{'}) \sin(\theta_1 - \theta_2)(r_1^{'} \sin\theta_2^{'} - r_2^{'} \sin\theta_1^{'}) +$$

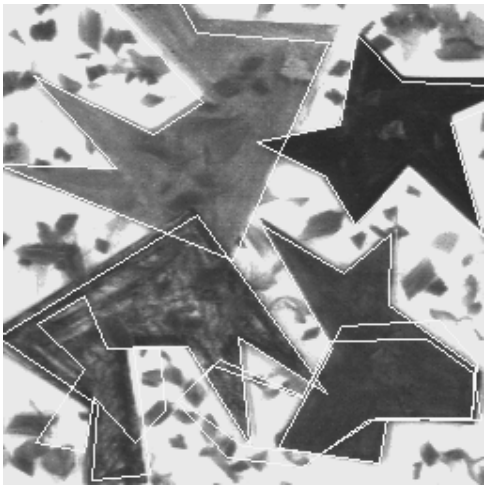
(a)



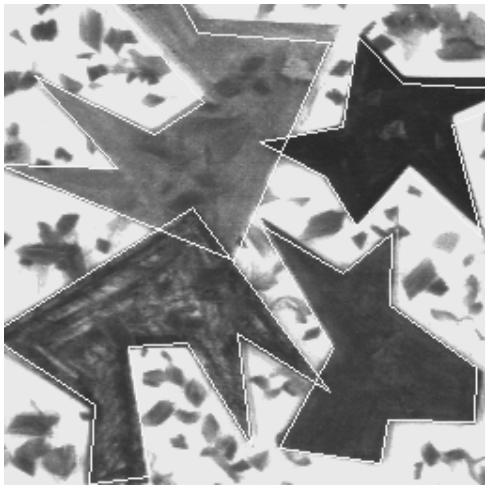
(b)



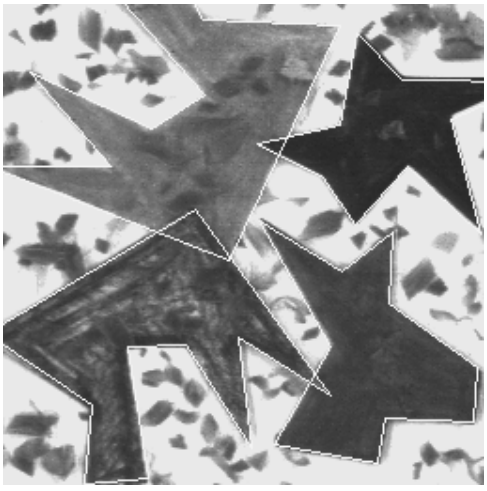
(c)



(d)

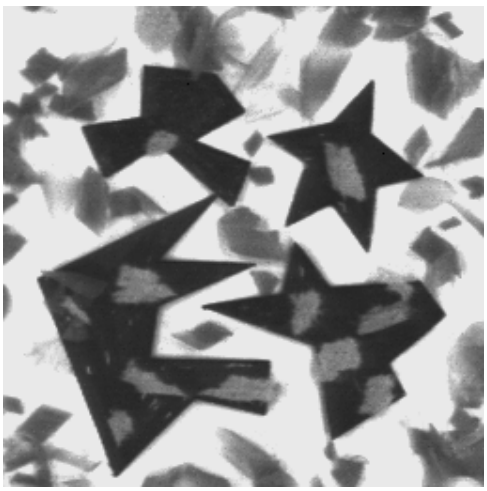


(e)

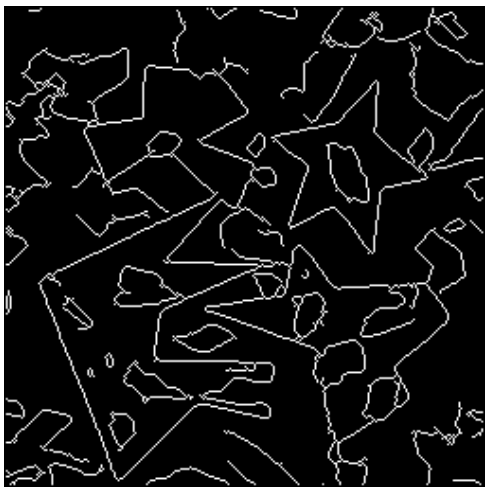


(f)

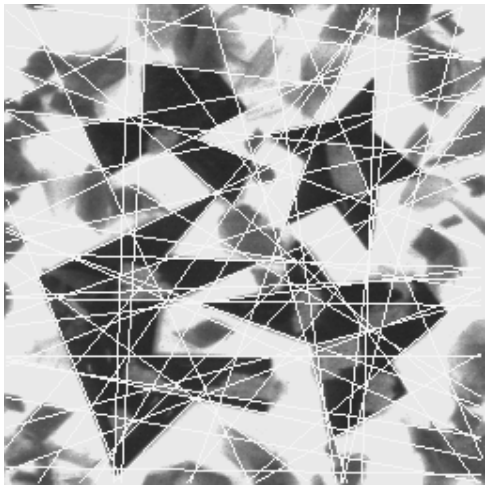
Figure 6: Example 5 of the experiments



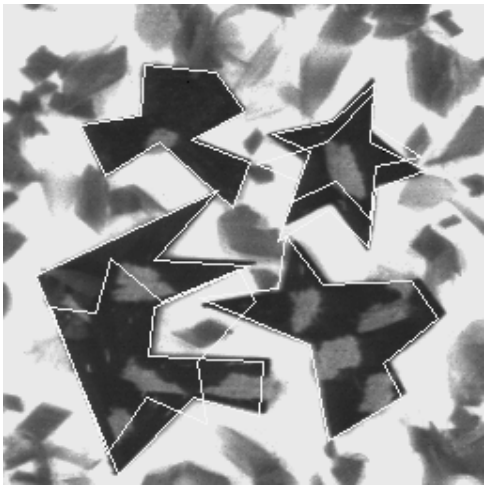
(a)



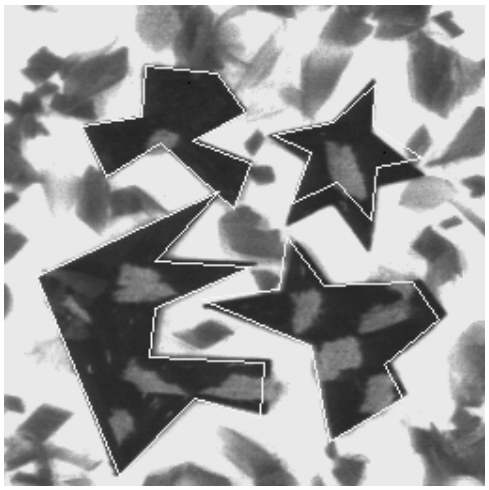
(b)



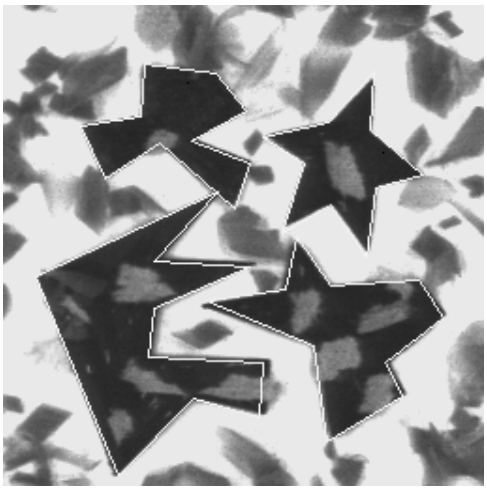
(c)



(d)

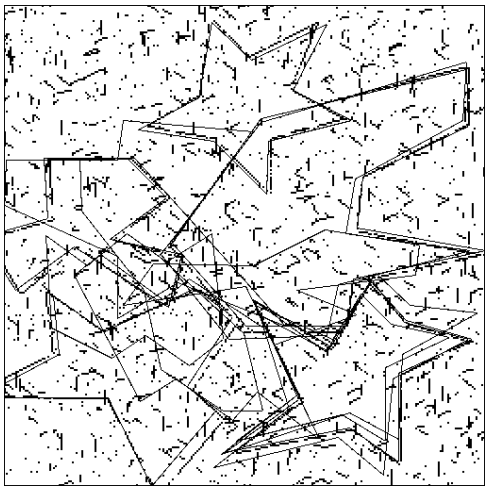


(e)

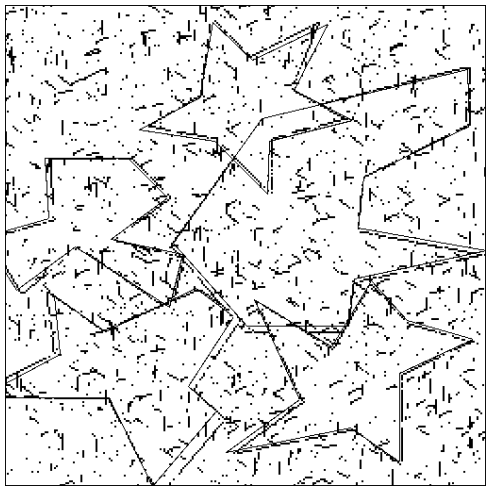


(f)

Figure 5: Example 4 of the experiments

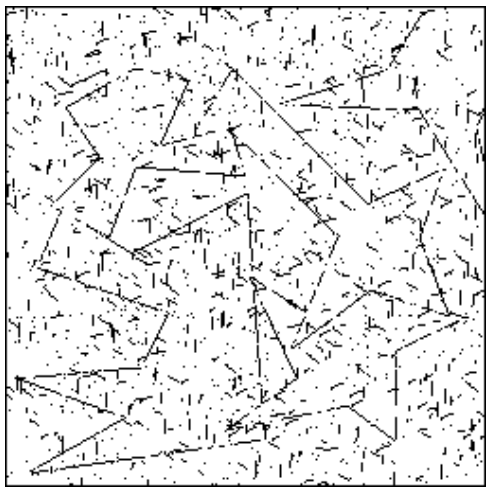


(c)

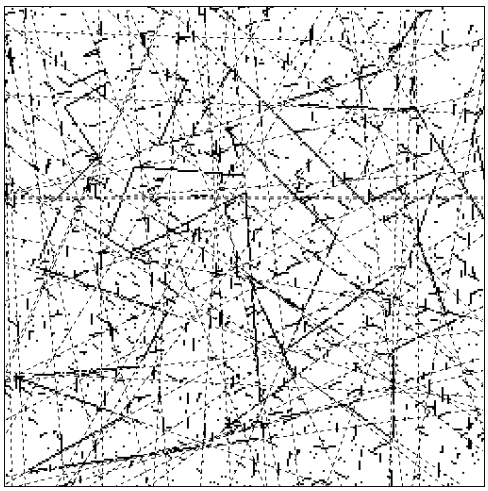


(d)

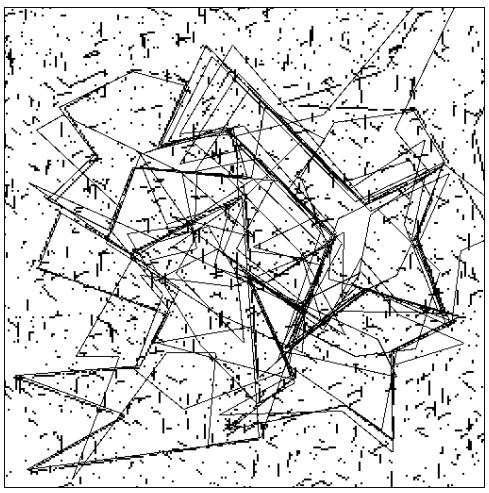
Figure 3: Example 2 of the experiments



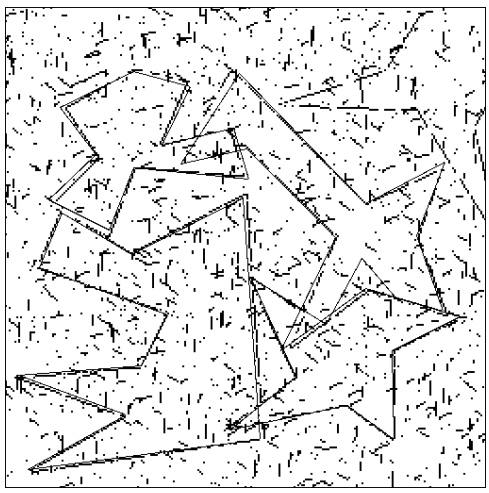
(a)



(b)

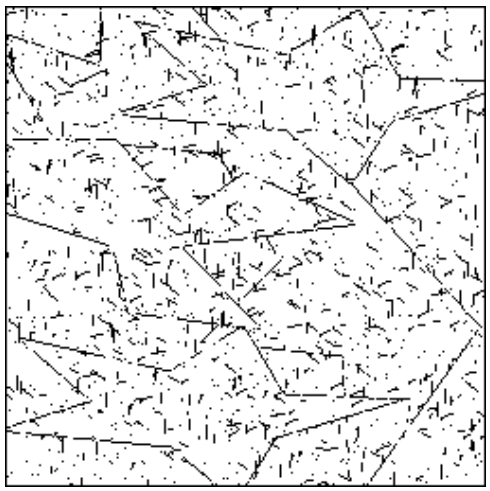


(c)

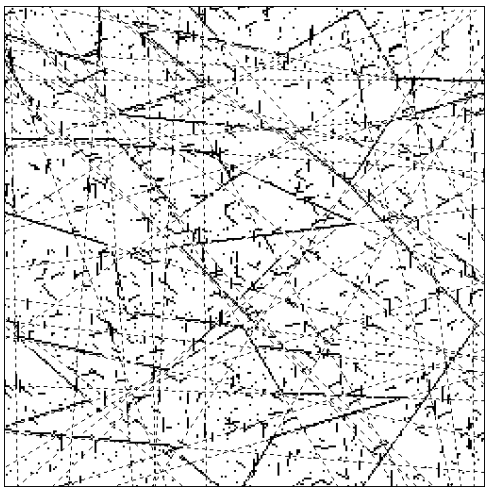


(d)

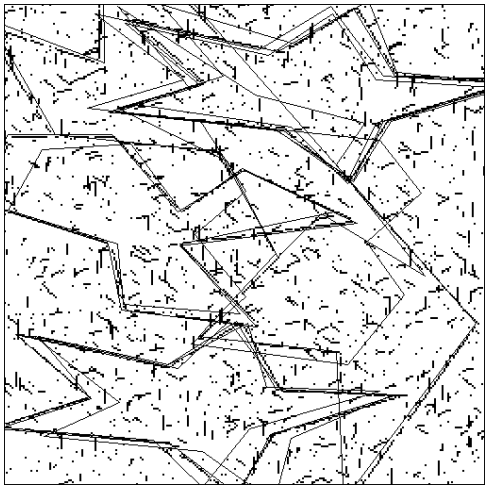
Figure 4: Example 3 of the experiments



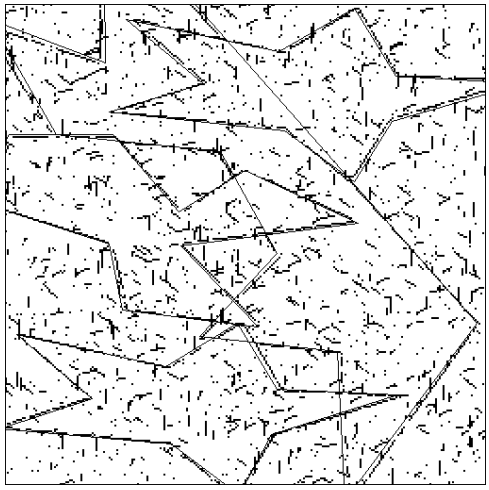
(a)



(b)

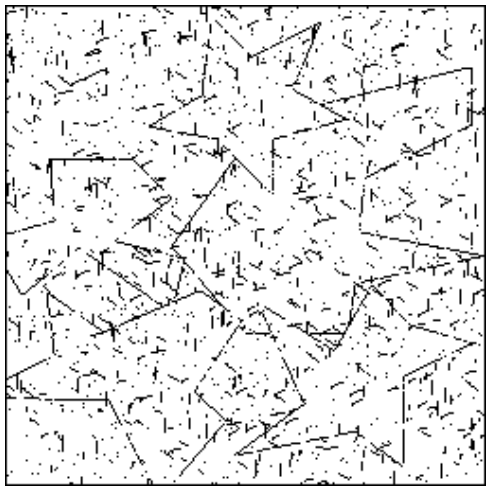


(c)

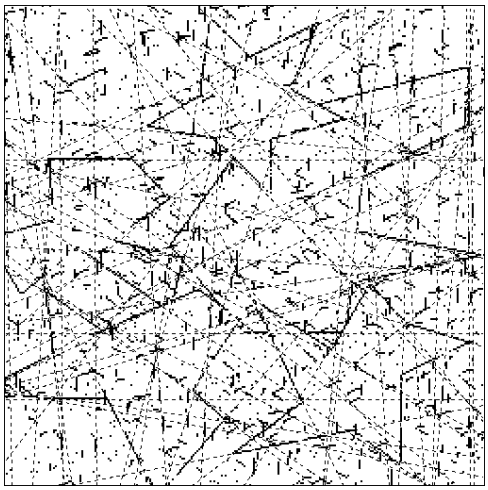


(d)

Figure 2: Example 1 of the experiments



(a)



(b)

Fig-3(a) shows a composite overlapping scene of model-0, 1, 2, 4 and 9. Fig-3(b) is the result of Hough analysis applied to this scene. 60 lines were detected. The covariance matrix used for the deviation of the detected lines from true lines is the same as in Fig-2. Fig-3(c) shows the model instances hypothesized in the recognition stage. 5000 probes were tried. Fig-3(d) shows the final result after disambiguation.

Fig-4(a) shows a composite overlapping scene of model-1, 3, 4, 13 and 19. Fig-4(b) is the result of Hough analysis applied to this scene. 60 lines were detected. The covariance matrix used for the deviation of the detected lines from true lines is the same as in Fig-2. Fig-4(c) shows the model instances hypothesized in the recognition stage. 5000 probes were tried. Fig-4(d) shows the final result after disambiguation. (Note that model-19 was not detected, since half of its boundary is invisible.)

Fig-5(a) shows a real image of model-0, 2, 3 and 4 with flakes scattered. Fig-5(b) is the result of edge detection using Boie-Cox edge detector [4]. Fig-5(c) shows the result of Hough analysis applied to this scene. 50 lines are detected. Fig-5(d) shows the recognition result before disambiguation. Fig-5(e) shows the recognition result after disambiguation. Fig-5(f) shows the result after best least-squares match.

Fig-6(a) shows a real image of model-1, 2 and 3 (note: model-3 appears twice) with flakes scattered. Fig-6(b) is the result of edge detection using also Boie-Cox edge detector. Fig-6(c) shows the result of Hough analysis applied to this scene. 50 lines are detected. Fig-6(d) shows the recognition result before disambiguation. Fig-6(e) shows the recognition result after disambiguation. Fig-6(f) shows the result after best least-squares match.

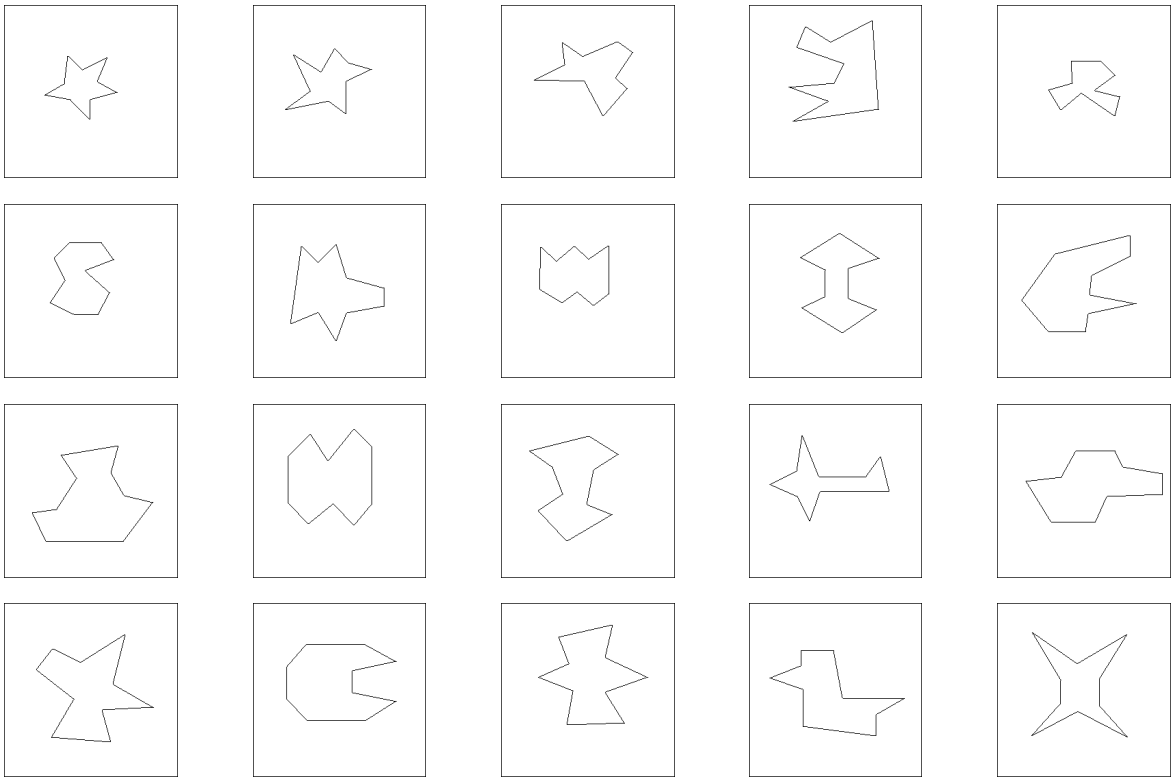


Figure 1: The twenty models used in our experiments. From left to right, top to bottom are model-0 to model-19.

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

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

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

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

The six unknowns can be solved by Cramer's rule (see p.89 of [15]) as follows:

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

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

Discussion

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

6.3 Experimental Results

We have done a series of experiments on synthesized images and real images containing polygonal objects, which are modeled by line features. Our model base consists of twenty models, each with ten lines (see Fig-1). For synthesized images, two kinds of noise are imposed onto the scene: positive and negative. Both kinds of noise are supposed to be introduced by occlusion, e.g. flakes in the milling process. Negative noise represents missing boundary pixels and is the result of occlusions on object boundary. Positive noise mainly includes edge pixels of those occluding objects and also random dot noise introduced in the imaging process.

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

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

Fig-2, Fig-3 and Fig-4 show three examples of the experiments on synthesized images. Fig-5 and Fig-6 show two examples of the experiments on real images, with best-least-squares match using Method-3 discussed before.

Fig-2(a) shows a composite overlapping scene of model-0, 3 and 4 (note: model-0 appears twice), which are significantly skewed. Fig-2(b) is the result of Hough analysis applied to this scene. 50 lines were detected. The covariance matrix used for the deviation of the detected lines from true lines was $\Sigma = \begin{pmatrix} 0.003384 & -0.00624 \\ -0.00624 & 2.5198 \end{pmatrix}$, obtained from statistics of an extensive simulations on images of different levels of degradation. Fig-2(c) shows the model instances hypothesized in the recognition stage. 5000 probes were tried. Fig-2(d) shows the final result after disambiguation.

Usually this distorted transformation transforms a model to match its scene instance with basis lines matching each other perfectly while the other lines deviating from their correspondences more or less. Knowledge of additional line correspondences between a model and its scene instance can be used to improve the accuracy of the computed transformation. In the following, we discuss several methods to minimize the errors.

Method 1

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

Discussion

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

Method 2

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

Discussion

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

Method 3

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

We derived in the following the closed-form formula for the case of affine transformations. The similar technique can be applied for the cases of other transformation group.

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

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

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

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

Then

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

and

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

And the components for the covariance matrix of the spread function $p(\Delta\Theta', \Delta R')$ is given by

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

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

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

Since we are dealing with highly occluded scenes, we have found that even if a hypothesis has passed verification (by verifying its boundary), it can still be a false alarm. By the viewpoint consistency principle [17], the locations of all object features in an image should be consistent with the projection from a single viewpoint. We may show that all 2-D objects lying on the same plane have an identical ratio of enlargement of area, if viewed from the same camera, assuming approximation of affine transformations to perspective transformations. The quantity of this ratio is $|\det(\mathbf{A})|$, where $\mathbf{A}$ is the skewing matrix of an affine transformation $\mathbf{T} = (\mathbf{A}, \mathbf{b})$. We call it as *skewing factor* and request that all the hypothesized instances have the same skewing factor. (Appendix-II provides the formula for computing $\mathbf{T}$ from a correspondence of a scene basis and a model basis.)

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

After sufficient number, which is determined by a statistical estimation procedure (see section 5.4), of bases are probed, a global skewing factor, which is taken to be the $|\det(\mathbf{A})|$ with highest number of votes, is obtained. Those hypothesized model instances voting for the global skewing factor are passed onto a disambiguation process we go on to describe below.

It is possible that a scene line is shared by many model instances, some of which are spurious. We could refine the result by classifying the shared lines to belong to that model instance most strongly supported by the evidence. One possible technique for disambiguation is *relaxation* [25]. However, relaxation is a complicated process. Since each model instance is associated with its quality measure, QM, as a measure of the strength of evidence supporting its existence (this definition of QM favors long segments of a model instance and is similar to that used in [1] with the same justification). The following straightforward technique proves to work well.

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

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

6.2 Best Least-Squares Match

The transformation between a model and its scene instance can be recovered by the correspondence of the model basis and the scene basis alone. However, scene lines detected by the Hough transform are usually somewhat distorted due to noise. This results in distortion in computing the transformation.

6.1 Implementation of Affine Invariant Matching

To apply the procedure described in section 5.3 to the case of affine group, a triplet of lines is necessary to form a basis. Eq. (1) and Eq. (2) are used to compute the invariant. Considering numerical stability, we have to prevent using small bases or large bases for invariant computing. Small bases are easily subject to perturbation; while large bases produce invariants with small values (thus without discriminativity). In our implementation, we use 256×256 images. If any side of the basis is less than 20 pixels or greater than 500 pixels, we consider it infeasible.

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

We also have to specialize Eq. (9), the spread function of the invariant, to the affine case (interested readers are referred to [22] for the cases of other transformations): 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 the line being encoded, (θ, r), the basis lines, $(\theta_i, r_i)_{i=1,2,3}$, and the computed invariant $(\theta^{'}, r^{'})$. We have

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

where the coefficients $c_{i,j}$'s are given by

$$\begin{aligned}
c_{11} &= \sin(\theta_1 - \theta_2)D, \\
c_{13} &= -\csc(\theta_1 - \theta_3)\sin(\theta - \theta_2)\sin(\theta - \theta_3)D, \\
c_{15} &= \csc(\theta_2 - \theta_3)\sin(\theta - \theta_1)\sin(\theta - \theta_3)D, \\
c_{17} &= -\csc(\theta_1 - \theta_3)\csc(\theta_2 - \theta_3)\sin(\theta_1 - \theta_2)\sin(\theta - \theta_1)\sin(\theta - \theta_2)D, \\
c_{21} &= [-C(\cos(\theta - \theta_1)\csc^2(\theta_1 - \theta_3)\sin(\theta - \theta_1) + \cos(\theta - \theta_2)\csc^2(\theta_2 - \theta_3)\sin(\theta - \theta_2))/B + \\
&\quad \csc(\theta_1 - \theta_2)(r_2\cos(\theta - \theta_1) - r_1\cos(\theta - \theta_2))]/E, \\
c_{22} &= 1/E, \\
c_{23} &= [-\csc(\theta_1 - \theta_2)(r_2\cos(\theta - \theta_1) + \cot(\theta_1 - \theta_2)(r_2\sin(\theta - \theta_1) - r_1\sin(\theta - \theta_2))) - \\
&\quad C((r_3\cos(\theta_1 - \theta_2) - r_2\cos(\theta_1 - \theta_3))/A - \\
&\quad \csc^2(\theta_1 - \theta_3)\sin(\theta - \theta_1)(\cos(\theta - \theta_1) + \cot(\theta_1 - \theta_3)\sin(\theta - \theta_1))/B) - \cot(\theta_1 - \theta_2)]/E, \\
c_{24} &= [-\sin(\theta_2 - \theta_3)AC - \csc(\theta_1 - \theta_2)\sin(\theta - \theta_2)]/E, \\
c_{25} &= [\csc(\theta_1 - \theta_2)(r_1\cos(\theta - \theta_2) + \cot(\theta_1 - \theta_2)(r_2\sin(\theta - \theta_1) - r_1\sin(\theta - \theta_2))) - \\
&\quad C((r_1\cos(\theta_2 - \theta_3) - r_3\cos(\theta_1 - \theta_2))/A - \\
&\quad \csc^2(\theta_2 - \theta_3)\sin(\theta - \theta_2)(\cos(\theta - \theta_2) + \cot(\theta_2 - \theta_3)\sin(\theta - \theta_2))/B) + \cot(\theta_1 - \theta_2)]/E, \\
c_{26} &= [\sin(\theta_1 - \theta_3)C/A + \csc(\theta_1 - \theta_2)\sin(\theta - \theta_1)]/E, \\
c_{27} &= [-C((r_2\cos(\theta_1 - \theta_3) - r_1\cos(\theta_2 - \theta_3))/A + \\
&\quad (\cot(\theta_1 - \theta_3)\csc^2(\theta_1 - \theta_3)\sin^2(\theta - \theta_1) + \cot(\theta_2 - \theta_3)\csc^2(\theta_2 - \theta_3)\sin^2(\theta - \theta_2))/B)]/E, \\
c_{28} &= [-\sin(\theta_1 - \theta_2)C/A]/E,
\end{aligned}$$

with

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

score value is set to 0. In computing $\log p(inv_{s_k}|[M_i, B_{M_i}, l_j])$, the background distribution function, compared with the spike induced by $[M_i, B_{M_i}, l_j]$ is usually negligible. We will use logarithm of Eq. (9) to approximate it. Typically this Gaussian $p.d.f.$ falls off rapidly. We thus approximately credit the same weighted score, $\log c$, to all those not in the neighborhood of the hash. Equivalently, we may instead credit $\log p(inv_{s_k}|[M_i, B_{M_i}, l_j]) - \log c$ to those nearby nodes accessed and keep intact those not in the neighborhood. After all the invariants are processed, the support for all the $[M_i, B_{M_i}]$ combinations are histogramed. Top few $[M_i, b_{M_i}]$'s with sufficiently high vote are verified by transforming and superimposing the model onto the scene. A quality measure, QM, defined as the ratio of visible boundary, is computed. If the QM is above a preset threshold (e.g. 0.5), the hypothesis passes the verification.

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

5.4 An Analysis of Probings Needed

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

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

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

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

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

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

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

Thus,

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

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

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

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

6 The Experiments

Since affine transformations are good substitutes for more general perspective viewing transformations. We choose to implement the case of affine transformations as follows. Similar approach can be applied to the cases of other transformations.

$$
\begin{aligned}
&= \frac{1}{p(inv_{s_1},\ldots,inv_{s_n})} \frac{P([M_i,B_{M_i}]|inv_{s_1})\,p(inv_{s_1})}{P([M_i,B_{M_i}])} \cdots \frac{P([M_i,B_{M_i}]|inv_{s_n})\,p(inv_{s_n})}{P([M_i,B_{M_i}])} \\
&= \frac{p(inv_{s_1}) \ldots p(inv_{s_n})}{p(inv_{s_1},\ldots,inv_{s_n})} \prod_{k=1}^{n} \frac{P([M_i,B_{M_i}]|inv_{s_k})}{P([M_i,B_{M_i}])} \\
&\propto \frac{p(inv_{s_1}) \ldots p(inv_{s_n})}{p(inv_{s_1},\ldots,inv_{s_n})} \prod_{k=1}^{n} \frac{\max_j p(inv_{s_k}|[M_i,B_{M_i},l_j]}{p(inv_{s_k})\,P([M_i,B_{M_i}])} \qquad \text{(by Eq. (10))} \\
&= \frac{1}{p(inv_{s_1},\ldots,inv_{s_n})} \prod_{k=1}^{n} \frac{\max_j p(inv_{s_k}|[M_i,B_{M_i},l_j]}{P([M_i,B_{M_i}])}.
\end{aligned}
$$

Thus

$$
P([M_i,B_{M_i}]|inv_{s_1},\ldots,inv_{s_n}) \propto \prod_{k=1}^{n} \max_j p(inv_{s_k}|[M_i,B_{M_i},l_j]), \tag{11}
$$

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

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

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

which means that the expected probability distribution of a scene invariant inv_{s_k} is independent of the other scene invariants. The intuition for this conditional independence to hold is that if the given inv_{s_k} does not belong to the embedded model M_i, it has nothing to do with $[M_i,B_{M_i}]$ and also other scene invariants; if the given inv_{s_k} belongs to the embedded model M_i, since geometric hashing applies transformation-invariant information, inv_{s_k} is destined to appear once B_{M_i} is selected and is not affected by other invariants. A more formal argument is given in [10].

In computing $P([M_i,B_{M_i}]|inv_{s_1},\ldots,inv_{s_n})$, we do not need to compute absolute probabilities but only relative probabilities. Since the logarithm function is monotonic, we can apply the logarithm to both sides of Eq. (11):

$$
\log\ P([M_i,B_{M_i}]|inv_{s_1},\ldots,inv_{s_n}) = \log C + \sum_{k=1}^{n} \log \max_j p(inv_{s_k}|[M_i,B_{M_i},l_j]), \tag{12}
$$

turning products into sums.

5.3 The Algorithm

The pre-processing stage parallels our description of the preprocessing stage of the geometric hashing method reviewed in section 2. For a specific transformation group considered, we use the formulae given in section 3 to compute the invariants. We also compute the spread functions of these invariants using Eq. (9). Both pieces of information, together with the identity of [model, basis, line], are stored in a node in the hash entry indexed by each invariant. Understanding that the spread function is Gaussian, we store its covariance matrix as a representation of the function. We note that each node records information about a particular [model, basis, line] combination necessary for our recognition stage. The hash table is so arranged that the position of each entry in fact represents a quantized coordinate (θ, r) of hash space. In this way, range query can be easily performed.

In the recognition stage, lines are detected by the Hough transform and a subset of these lines are chosen as a basis for associating invariants with all remaining lines. Eq. (12) is used to accumulate support from all these invariants ($\log C$ term, which is common to all hypotheses, can be ignored). We note that a straightforward use of Eq. (12) runs through all the [model, basis, line] combinations and thus the computational cost can be immense. By advantage of the geometric hash table prepared in the pre-processing stage, for each invariant inv_{s_k} hashed into the hash space, we access only its nearby nodes. For each node accessed, we credit $\log p(inv_{s_k}|[M_i,B_{M_i},l_j])$ to the node, whose initial

the bin being hashed. However, our analysis of noise effect on computed invariants in the preceding section shows that the perturbation of an invariant has a Gaussian distribution with non-zero correlation coefficient. This implies that the region of hash space that would need to be accessed is an ellipse instead of a circle centered around the "true" value of the invariant. Moreover, this perturbation also depends on the basis and the line being encoded, which means that different invariants should associate with different size, shape and orientation of hash space regions for error tolerance.

The nodes in hash entries more "closely match" value of the invariant should get higher weighted score than those which are far away, in a manner accurately representing the behavior of the noise which affects these invariants. Let inv_s be a scene invariant and let inv_{M_i} be a model invariant computed from a basis B_{M_i} and a line l_j of M_i. Bayes' theorem [7] allows us to precisely determine the probability that inv_s results from the presence of a certain $[M_i, B_{M_i}, l_j]$:

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

where

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

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

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

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

$$\begin{aligned} P([M_i, B_{M_i}])|inv_s) \quad &\approx \quad \max_j P([M_i, B_{M_i}, l_j]|inv_s) \\ &\propto \quad \frac{\max_j \; p(inv_s|[M_i, B_{M_i}, l_j])}{p(inv_s)}. \end{aligned} \tag{10}$$

5.2 Evidence Synthesis by Bayesian Reasoning

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

In order to synthesize support coming from different invariants, we have the following formula, based on probability derivation using Bayes' theorem [5] :

$$\begin{aligned} & \frac{P([M_i, B_{M_i}]|inv_{s_1}, \ldots, inv_{s_n})}{P([M_i, B_{M_i}])} \\ = \quad & \frac{p(inv_{s_1}, \ldots, inv_{s_n}|[M_i, B_{M_i}])}{p(inv_{s_1}, \ldots, inv_{s_n})} \\ = \quad & \frac{p(inv_{s_1}|[M_i, B_{M_i}]) \ldots p(inv_{s_n}|[M_i, B_{M_i}])}{p(inv_{s_1}, \ldots, inv_{s_n})} \qquad \text{(assuming conditional independence)} \end{aligned}$$

where

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

Since

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

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

$$p(\Delta\Theta', \Delta R') = \frac{1}{2\pi\sqrt{|\mathbf{\Sigma}'|}} \exp[-\frac{1}{2}(\Delta\Theta', \Delta R')\mathbf{\Sigma}'^{-1}(\Delta\Theta', \Delta R')^t] \tag{9}$$

and covariance matrix

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

where

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

For a detailed derivation of the above, the reader is referred to Appendix-I of [21]. Recall that in computing the invariant $(\theta', r')^t$, we may adjust the value of θ' by adding π or $-\pi$ with the sign of r' flipped accordingly such that θ' is in $[0..\pi)$. If this is the case, then the covariance matrix $\mathbf{\Sigma}'$ must be adjusted by flipping the sign of $\sigma_{\theta'r'}$.

5 Geometric Hashing with Weighted Voting

Our recognition scheme builds upon the geometric hashing method. Even though we propose to use line features in a degraded image, we still face the problem dealing with image noise and the resulting problem of noise in the invariants used for hashing, as discussed in the preceding section.

To minimize these noise effects, we need to derive a theoretically justified scheme for weighted voting among hash bins. This is the aim of the probabilistic procedure which we now go on to describe.

5.1 A Measure of Matching between Scene and Model Invariants

In the geometric hashing method described in section 2, during recognition a computed scene invariant is used to index the geometric hash table and tally one vote for the entry retrieved. However, considering the presence of noise, a scene invariant can not exactly hit the hash table entry where its matching model invariant is supposed to reside. Gavrila and Groen [8] assumes a bounded circular region around each hash bin and tally equal vote for all the bins within the region centered around

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

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

and is centered around (θ, r).

4.2 The Spread Function of the Invariants

In section 3, we have given formulae of the invariants (θ', r')'s for various transformations. These invariants are functions of the basis lines, $(\theta_i, r_i)^t_{i=1,\ldots,k}$, and the line being encoded, $(\theta, r)^t$:

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

Equivalently we may rewrite the above equation as follows:

$$\theta' = f'((\theta, r)^t, (\theta_1, r_1)^t, \cdots, (\theta_k, r_k)^t), \quad (3)$$
$$r' = f''((\theta, r)^t, (\theta_1, r_1)^t, \cdots, (\theta_k, r_k)^t). \quad (4)$$

The exact forms of Eq. (3) and Eq. (4), together with the value of k (i.e. the number of basis lines needed), depend on the viewing transformation under consideration and are given in section 3.1, 3.2, 3.3 and 3.4 respectively.

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

$$\theta' + \delta\theta' = f'((\theta + \delta\theta, r + \delta r)^t, (\theta_1 + \delta\theta_1, r_1 + \delta r_1)^t, \cdots, (\theta_k + \delta\theta_k, r_k + \delta r_k)^t), \quad (5)$$
$$r' + \delta r' = f''((\theta + \delta\theta, r + \delta r)^t, (\theta_1 + \delta\theta_1, r_1 + \delta r_1)^t, \cdots, (\theta_k + \delta\theta_k, r_k + \delta r_k)^t). \quad (6)$$

Expanding Eq. (5) and Eq. (6) in Maclaurin series and ignoring second and higher order perturbation terms and then subtracting Eq. (3) and Eq. (4) from them, we have

$$\begin{aligned}\delta\theta' &= \frac{\partial\theta'}{\partial\theta}\delta\theta + \frac{\partial\theta'}{\partial r}\delta r + \sum_{i=1}^{k}\left(\frac{\partial\theta'}{\partial\theta_i}\delta\theta_i + \frac{\partial\theta'}{\partial r_i}\delta r_i\right) \\ &= c_{11}\delta\theta + c_{12}\delta r + c_{13}\delta\theta_1 + c_{14}\delta r_1 + \ldots + c_{1,2k+1}\delta\theta_k + c_{1,2k+2}\delta r_k, \quad (7)\\ \delta r' &= \frac{\partial r'}{\partial\theta}\delta\theta + \frac{\partial r'}{\partial r}\delta r + \sum_{i=1}^{k}\left(\frac{\partial r'}{\partial\theta_i}\delta\theta_i + \frac{\partial r'}{\partial r_i}\delta r_i\right) \\ &= c_{21}\delta\theta + c_{22}\delta r + c_{23}\delta\theta_1 + c_{24}\delta r_1 + \ldots + c_{2,2k+1}\delta\theta_k + c_{2,2k+2}\delta r_k, \quad (8)\end{aligned}$$

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

Let $(\Delta\Theta, \Delta R)$, $(\Delta\Theta_i, \Delta R_i)_{i=1,\ldots,k}$ and $(\Delta\Theta', \Delta R')$ be stochastic variables denoting the perturbations of (θ, r), $(\theta_i, r_i)_{i=1,\ldots,k}$ and (θ', r') respectively. Eq. (7) and Eq. (8) can be rewritten as

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

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

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

where

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

3.4 Line Invariants under Projective Transformations

Five lines are necessary to define a shape signature under the projective group: Given an ordered quadruplet of lines $(\theta_i, r_i)^t_{i=1,2,3,4}$ such that no three lines are parallel to one another or intersect at a common point, any fifth line $(\theta, r)^t$ can be encoded in terms of these four lines in a projective-invariant way.

One way to encode the fifth line is to find a transformation $\mathbf{T}$ which maps the four basis lines to a canonical basis, say $x=0$, $y=0$, $x=1$ and $y=1$ (i.e. $(0,0)^t$, $(\frac{\pi}{2},0)^t$, $(0,1)^t$ and $(\frac{\pi}{2},1)^t$ respectively), then apply $\mathbf{T}$ to the fifth line. It can be shown that the invariant $(\theta', r')^t$ of $(\theta, r)^t$ with respect to $(\theta_i, r_i)^t_{i=1,2,3,4}$ is as follows:

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

where

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

4 The Effect of Noise on the Invariants

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

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

4.1 A Noise Model for Line Parameters

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

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

A convenient way to encode the third line is to find a transformation $\mathbf{T}$ which maps the first line to the x-axis and maps the second line to such that its intersection with the first line is the origin (i.e. maps them to $(\frac{\pi}{2}, 0)^t$ and $(\varphi, 0)^t$ respectively, where φ is fixed though unknown), then apply $\mathbf{T}$ to the third line. It can be easily shown that the resulting invariant $(\theta', r')^t$ of $(\theta, r)^t$ with respect to $(\theta_i, r_i)^t_{i=1,2}$ is as follows:

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

or

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

Note that we have two solutions due to the ambiguity discussed above. However, during recognition we need to compute only either of them, since we may store both invariants in the geometric hash table during preprocessing.

3.2 Line Invariants under Similarity Transformations

Four lines are necessary to define a shape signature under the similarity group: Given an ordered set of triplet of lines $(\theta_i, r_i)^t_{i=1,2,3}$, not all of which are parallel to each other and intersect at a common point, any fourth line $(\theta, r)^t$ can be encoded in terms of this triplet as a basis in a similarity-invariant way.

Without loss of generality, we may assume that the first line intersects the remaining two basis lines. One way to encode the fourth line is to find a transformation $\mathbf{T}$ which maps the first line to the x-axis, maps the second line to such that its intersection with the first line is the origin and maps the third line to such that its intersection with the first line has Euclidean coordinate $(1, 0)$ (i.e. maps them to $(\frac{\pi}{2}, 0)^t$, $(\varphi_1, 0)^t$ and $(\varphi_2, \sin|\theta_1 - \theta_3|)^t$ respectively, with φ_1 and φ_2 fixed though unknown), then apply $\mathbf{T}$ to the fourth line. It can then be shown that the resulting invariant $(\theta', r')^t$ of $(\theta, r)^t$ with respect to $(\theta_i, r_i)^t_{i=1,2,3}$ is as follows:

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

where

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

3.3 Line Invariants under Affine Transformations

Four lines are necessary to define a shape signature under the affine group: Given an ordered triplet of lines $(\theta_i, r_i)^t_{i=1,2,3}$, which are not parallel to one another and do not intersect at a common point, any fourth line $(\theta, r)^t$ can be encoded in terms of these three lines in an affine-invariant way.

One way to encode the fourth line is to find a transformation $\mathbf{T}$ which maps the three basis lines to a canonical basis, say $x = 0$, $y = 0$ and $x + y = 1$ (i.e. $(0, 0)^t$, $(\frac{\pi}{2}, 0)^t$ and $(\frac{\pi}{4}, \frac{1}{\sqrt{2}})^t$ respectively), then apply $\mathbf{T}$ to the fourth line. As before, it can be shown in this case that the resulting invariant $(\theta', r')^t$ of $(\theta, r)^t$ with respect to $(\theta_i, r_i)^t_{i=1,2,3}$ is as follows:

$$\begin{aligned} \theta' &= \tan^{-1}(\csc(\theta_1 - \theta_3) \sin(\theta_1 - \theta) \csc(\theta_1 - \theta_2) A / \\ &\qquad \csc(\theta_2 - \theta_3) \sin(\theta_2 - \theta) \csc(\theta_1 - \theta_2) A), \qquad (1) \\ r' &= \frac{1}{\tau}(r_1 \csc(\theta_1 - \theta_2) \sin(\theta_2 - \theta) - r_2 \csc(\theta_1 - \theta_2) \sin(\theta_1 - \theta) + r), \qquad (2) \end{aligned}$$

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

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

The recognition stage

Given a scene with n feature points extracted, we

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

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

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

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

(v) hypothesize the existence of an instance of model M_i in the scene if $[M_i, b_j]$, for some j, peaks in the histogram with sufficiently many hits;

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

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

3 Line Invariants under Various Transformation Groups

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

Potentially relevant transformation groups include rigid transformations, similarity transformations, affine transformations and projective transformations, depending on the manner in which an image is formed. Each of these classes of transformations is a subgroup of the full group of projective transformations. Since a projective transformation is a collineation (referring to p. 89 of [12]), all these transformations preserve lines. This allows us to use line features as inputs to the recognition procedures.

We will give the formulae of the invariants under various transformation groups in the following. For additional detail, the reader is referred to [24]. Throughout the following discussion, we represent a line by its normal parameterization (θ, r) with θ in $[0, \pi)$ and r in $\mathbf{R}$. If the computed invariant (θ', r') has θ' not in $[0..\pi)$, we adjust it by adding π or $-\pi$ and adjust the value of r' by flipping its sign.

3.1 Line Invariants under Rigid Transformations

Three lines are necessary to define a *shape signature* under the Euclidean group up to a 180°-rotation ambiguity: Given an ordered pair of non-parallel lines $(\theta_i, r_i)^t_{i=1,2}$, any third line $(\theta, r)^t$ can be encoded in terms of this pair in a rigid-invariant way up to a 180°-rotation ambiguity. This ambiguity can be easily broken if we know the position of a third line during verification.

1 Introduction

Visual object recognition 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.

Prior research (see surveys [2,3,6]) 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. Geometric hashing [14] is a model-based approach which precompiles redundant transformation-invariant information derived for object models into a hash table. Then during recognition, the same invariants are computed from features in a scene and used as indexing keys to retrieve from the hash table the possible matches with model features.

In noisy scenes, the locations of point features can be hard to detect and the analysis of geometric hashing on point sets [9] is much troubled by noise. Line features are more robust and can be extracted by the Hough transform method [11] with greater accuracy, since more image points are involved. Although point features can be obtained by intersecting lines, this increases uncertainty. Hence it can be advantageous to work directly on line features by computing the geometric invariants of lines using a combination of lines.

We examine line invariants under various transformations and investigate their statistical behavior under uncertainty. We impose minimal segmentation requirements: Only positional information of edgels is assumed to be available. The method presented combines use of the Hough transform, geometric hashing and weighted voting by Bayesian reasoning.

This paper is organized as follows. In section 2, we give a brief review of the original geometric hashing method to make the paper self-contained. In section 3, we give closed-form formulae for line invariants under various viewing transformations. Section 4 analyzes the perturbation of the computed invariant from the perturbations of the lines giving rise to this invariant. A Gaussian noise model is assumed. This analysis yields a measure of the closeness of two invariants, which can be used to pick out the most promising candidate set of model features from a pool of alleged sets. Section 5 describes the resulting improvement of the recognition procedure. Section 6 describes a system implementation for the case of affine transformations. We also discuss best least-squares match procedures for model registration. Experimental results are also presented. Section 7 discusses some observations of the approach and a possibility of parallel realizations.

2 Brief Review of the Geometric Hashing Method

The geometric hashing idea has its origins in work of Schwartz and Sharir [20]. Application of the geometric hashing idea for model-based visual recognition was described by Lamdan, Schwartz and Wolfson. This section outlines the method; a more complete description can be found in Lamdan's dissertation [13]; later developments of the technique are found in [18,19,23,24].

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

The pre-processing stage

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

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

A Probabilistic Approach to Geometric Hashing using Line Features

Frank Chee-Da Tsai

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

Technical Report No. 640

June 1993

Abstract

Most current object recognition algorithms assume reliable image segmentation, which in practice is often not available. We examine 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. Prior work on the performance analysis of geometric hashing has focused on point features and shown its noise sensitivity. This paper uses line features to compute recognition invariants in a potentially more robust way. We investigate the statistical behavior of these line features analytically. Various viewing transformations, which 2-D (or flat 3-D) objects undergo during image formation, are considered. For the case of affine transformations, which are often suitable substitutes for more general perspective transformations, we show experimentally that the technique is noise resistant and can be used in highly occluded environments.