Geometric Hashing with Attributed Features

Jyh-Jong Liu * Robert Hummel †

Computer Science Department
Courant Institute of Mathematical Science
New York University
New York, NY 10012

Abstract

Geometric hashing systems for object recognition have typically made use of point features in order to describe models and objects. When lines have been included as primitive features, they have been used to generate collections of points from pairwise intersections. In the experiments described in this paper, we use line features that include location and orientation information. These features, for which the orientation information is an attribute, are incorporated into a geometric hashing system using weighted voting in order to effect a Bayesian-based maximum likelihood object recognition system. We show results of this system which is the first example of the use of attributed features (features with more than coordinate position information) in a geometric hashing application.

1 Introduction

Geometric hashing is a method that uses combinations of feature vectors to index into databases of models. As an object recognition system, geometric hashing can be an effective and efficient way of using spatial arrangements of features to locate instances of models. A description of the geometric hashing method is given by Lamdan and Wolfson [11], and early prototype systems have given good results [9,10]. Numerous systems based an recognition of point patterns and space curves have been demonstrated [13,3,7,8,5]. A parallel implementation on a CM-2 is reported in [14].

The purpose of a geometric hashing system for object recognition is to locate instances of models in a scene. A model is a pattern of features that may be embedded in a scene after a transformation from a restricted class of allowable transformations. A scene is a collection of features extracted from an image. The search problem of locating instances of models embedded in a scene is confounded by the existence of an unknown transformation, by obscuration that can occur, and by noise that can cause deviations of the features in the scene. Geometric hashing uses a transformation-invariant hash functions to speed the search.

An error analysis of possible configurations of embedded, noisy models in scenes show that hash functions on minimal collections of features may not provide much in the way of discrimination power, and thereby may lead to failure of the method in practical object recognition systems [4,2]. These difficulties can be circumvented by (1) using similarity invariance instead of affine or perspective invariance, (2) using weighted voting, as developed in the thesis of Rigoutsos [12], and (3) using *attributed* features, i.e., features endowed with additional information that can be used to filter the matches, as advocated by Califano [1]. In this paper, we use all these strategies to reduce false alarms. Our emphasis, however, is on the use of the third strategy: placing attributes on features to enable matching filtered by an extra criterion. In our case, the features are still point locations, but will carry the additional information of an orientation attribute.

2 Formulation

We will describe a particular formulation of a geometric hashing system using point features and orientation attributes. This system will be used to find objects in realistic images using models consisting of line segments. We will use similarity transformations (rotation, translation, and scale) as the invariance class for recognition.

A single feature consists of a point location (x, y) and an orientation θ. Orientations are not directed,

*E-mail: jyhjong@slinky.nyu.edu
†E-mail: hummel@cs.nyu.edu

0-8186-5310-8/94 $03.00 © 1994 IEEE

so $0 \leq \theta < \pi$. In practice, θ is stored as a cosine/sine pair to avoid trigonometric evaluations.

A model consists of a prototype collection of features $\{(x_i, y_i, \theta_i)\}_{i=1}^{n}$. We will have many models, $M_1, \ldots, M_n$, so that $M_k = \{(x_{k,i}, y_{k,i}, \theta_{k,i})\}_{i=1}^{n_k}$. Different models may have different number of features.

If T is a similarity transformation, then T can be described as a translation by some vector (a, b), followed by a rotation angle ϕ, followed by a scaling by s. Although T is a transformation of point features, it extends to an operator $\tilde{T}$ on attributed features (x, y, θ) by transforming (x, y) to $T(x, y)$, and rotating θ to $\theta + \phi$. This is because a line through the point (x, y) having orientation θ will be transformed under T to a line through $T(x, y)$ having orientation $\theta + \phi$. The new orientation should be regarded as being modulo π, so that

$$\tilde{T}(x, y, \theta) = (T(x, y), \theta + \phi \bmod \pi).$$

The object recognition problem can be stated very simply. Given a collection of models $M_1, \ldots M_n$, with each model formed by a set of features $M_k = \{(x_{k,i}, y_{k,i}, \theta_{k,i})\}_{i=1}^{n_k}$ and given a scene $S = \{(\hat{x}_j, \hat{y}_j, \hat{\theta}_j)\}_{j=1}^{s}$, find *instances* of models in the scene. An instance is defined to be a model number k, a transformation T, and a subset S' of S such that for each feature $(\hat{x}_j, \hat{y}_j, \hat{\theta}_j) \in S'$ in the subset of scene features, some transformed model feature approximates it: $\tilde{T}(x_{k,i}, y_{k,i}, \theta_{k,i}) \approx (\hat{x}_j, \hat{y}_j, \hat{\theta}_j)$. In order to be a valid instance, the number of features in the subset S' must be some large fraction of the number of features n_k in model M_k.

3 Geometric Hashing

Geometric hashing may be used to organize the search for instances of models in scenes. Our presentation will be specific to the problem of similarity-invariant matching of models composed of point features attributed with orientation information. A much more general presentation can be found in [6, in preparation]. .

In a preprocessing phase, information about the models are stored in a hash table. Models are redundantly encoded. Every model is encoded once for every reasonable basis set within the model. For our purpose, a basis set is simply a pair of points. As a hash function, we use a similarity-invariant computation of coordinates of a feature relative to a basis set.

Specifically, consider model M_k and basis points $(x_{k,\mu}, y_{k,\mu}, \theta_{k,\mu})$, $(x_{k,\nu}, y_{k,\nu}, \theta_{k,\nu})$ from model k, which

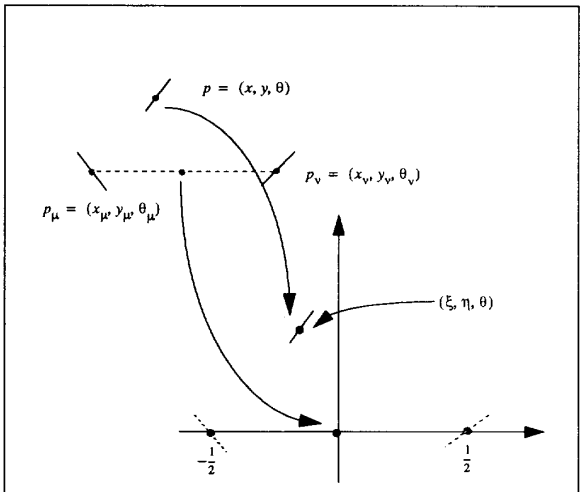

Figure 1: The construction of the hash function, $h(p_\mu, p_\nu, p)$.

we will denote by p_μ and p_ν respectively. For every other point $p = (x_{k,i}, y_{k,i}, \theta_{k,i})$ in M_k, we define

$$h(p_\mu, p_\nu, p) = (\xi, \eta, \psi),$$

where (ξ, η, ψ) gives the feature value of $\tilde{T}(p)$, where T is defined to be the similarity transformation such that

$$T(x_{k,\mu}, y_{k,\mu}) = (-1/2, 0)$$

$$T(x_{k,\nu}, y_{k,\nu}) = (1/2, 0)$$

Figure 1 displays graphically the definition of the hash function h. The hash function yields a vector with these components, and unlike the typical computer science notion of a hash function, h is continuous in all its variables.

The hash space consists of a set of entries. For the model M_k, and for basis pair p_μ and p_ν, there are n_k entries, each associated with $h(p_\mu, p_\nu, p)$ for $p \in M_k$. An entry is *tagged* with the information (M_k, p_μ, p_ν) (which can be encoded using the indices as pointers), and has a location in three-dimensional hash space given by the hash value.

In the recognition phase, features are extracted from a scene, yielding a set of attributed points $S = \{\hat{p}_j = (\hat{x}_j, \hat{y}_j, \hat{\theta}_j)\}_{j=1}^{s}$. A pair of points are chosen as a *trial basis*, say p_α and p_β. Using the trial basis, we perform a set of *probes*, which together constitute a single *trial*. A probe consists of the computation of

a hash value $h(p_\alpha, p_\beta, p) = (\xi, \eta, \theta)$, based on a point $p \in S$ and then a determination of all entries that lie close to (ξ, η, θ) in hash space. For each such entry, we can compute a distance from its location to (ξ, η, θ), and we then increment an accumulator associated with its tag (a model number k and a pair of point indices in M_k, say (μ, ν)), by an amount related to that distance. This is done for every entry near (ξ, η, θ) and is repeated for all possible probes as p varies over S. The Bayesian reasoning interpretation, given in the next section, allows us to specify the metrics and functions.

4 Bayesian Reasoning

Let us suppose that model M_k is embedded in the scene, and that after renumbering the scene points, the points $\hat{p}_1, \ldots, \hat{p}_{n_k} \in S$ correspond to an instance of the model. For the moment, we are assuming that all features of model M_k are present in the scene. Let us also suppose that scene points $\hat{p}_1$, $\hat{p}_2$ are chosen as a trial basis. If there is no noise, then each of the probes $h(\hat{p}_1, \hat{p}_2, \hat{p}_j)$, for $j = 1, \ldots n_k$, should land on entry with tag (k, μ, ν), where M_k is the embedded model and p_μ, p_ν is the pair of points in M_k corresponding to $\hat{p}_1$, $\hat{p}_2$ in S. Indeed, the set of entries with tag (k, μ, ν) are formed by computing $h(p_\mu, p_\nu, p)$, for $p \in M_k$, and thus constitute an image of M_k normalized by (p_μ, p_ν). Likewise, with the scene points S are normalized with respect to $h(\hat{p}_1, \hat{p}_2)$, and the embedded model $\hat{p}_j, j = 1, \ldots, n_k$, will map to the same image. Indeed, not only will all locations match, the orientation information should also match in order for the instance to be a valid embedding. In fact, even the orientation information of the transformed basis points should match.

In the presence of noise, there will be in the positions of the normalized features in the hash space. We will assume that individual features in the scene domain are subject to Gaussian perturbations. Thus a position of feature (x, y) belonging to a model in the scene can be perturbed by a distance in the Euclidean plane with standard deviation σ, and the orientation θ can be rotated by an angle whose standard deviation is τ radians. In practice, we measure the angle deviation using the sine of the angle difference, rather than the actual angle difference. For small angular perturbations, the two are the same.

At this point, we must assume independence of the perturbations. The validity of this assumption will depend on the models and choice of the features. Independence means the following: Under the assumption that a particular model M_k is embedded in the scene, and under the assumption that basis pair p_μ and p_ν in M_k match a particular basis of scene points, say $\hat{p}_1$ and $\hat{p}_2$ then the density distribution function in (x, y, θ) space for any collection of t features, $t \leq s$, is simply the product of t density distribution functions in (x, y, θ)-space for single features. We discuss the validity of the independence assumption for various feature representations (using orientation-attributed features) in the next section.

Using this assumption, we find that the joint density distribution of the features of the embedded model are Gaussianly-distributed. In order to compute the probability that a particular configuration of n_k scene features matches a model M_k with respect to a particular set of correspondence, it suffices to sum the relative Gaussian perturbations, in (x, y, θ) space, measured as a fraction of the individual standard deviations, and to normalize the result relative to all other hypotheses. In practice, using a maximum likelihood classifies dictates that we wish to minimize the sum of the individual deviations, measured in terms of standard deviations.

While we will not give the detail here, it turns out that the same holds true in the hash space. That is, for a given trial basis pair in the scene space, the projected image of the model M_k in the hash space will have (an approximate) Gaussian distribution, both for individual hashes, and for the joint density distribution of the collection of features.

Accordingly, for any given model M_k and basis pair p_μ, p_ν in M_k, the probability that M_k is present in the scene with p_μ, p_ν matching $\hat{p}_1$, $\hat{p}_2$ is related to a sum of deviations. Each deviation is computed by finding the closest hash from the scene, i.e., $h(\hat{p}_1, \hat{p}_2, \hat{p})$, over $\hat{p} \in S$, to a given entry associated with model M_k and basis μ, ν, and measuring the distance relative to the expected covariance about that entry.

This is precisely what the geometric hashing algorithm, using weighted voting, accomplishes. Instead of performing a sum over all entries, we use scene hashes, and only accumulate scores for nearby entries, which amounts to the same thing. Although we again omit all details and further justifications, the following formulas are used in our experiments.

Let $h(\hat{p}_1, \hat{p}_2, \hat{p}) = (\xi, \eta, \phi)$ represent a hash of a scene configuration. Suppose an entry at location (x, y, θ) is nearby, and is associated with model M_k, basis pair p_μ, p_ν. The vote given to model/basis (k, μ, ν) is

$$\log\left(1 + \frac{(4(\xi^2 + \eta^2) + 3)^2 \|\hat{p}_2 - \hat{p}_1\|^2}{12\sigma^2\sqrt{2\pi}\tau(s - n_k)(4(x^2 + y^2) + 3)}\right).$$

$$\exp\left(-\left(\frac{((\xi-x)^2+(\eta-y)^2)\|\hat{p}_2-\hat{p}_1\|^2}{\sigma^2(4(x^2+y^2)+3)}+\frac{\sin^2(\phi-\theta)}{2\tau^2}\right)\right).$$

In this formula, s is the total number of features in the scene, and n_k is the number of features in model M_k and $\|\hat{p}_2-\hat{p}_1\|$ is the separation distance of the trial basis point in the scene, measured as Euclidean distance in (x,y)-space. To initialize the process, model/basis (k,μ,ν) begins with a bias of

$$(s-q)\log\left[\frac{s-q-\alpha(n_i-q)}{s-q-\alpha(n_{max}-q)}\right].$$

We have used the assumption that all model/basis combinations are a priori equally likely. A major complication in deriving these formulas, and an innovative contribution of this work that has not been used before in geometric hashing, is that the formulas account for the fact that the number of features n_k in a model can vary.

5 Feature Extraction

The experiments done by Lamdan and Rigoutsos [11,12] use *interest points* such as high curvature points as the features. Line features are used in Tsai's approach [15], represented by the parameters (r,θ). We are also interested in line features, but our approach is different from Tsai's approach. The idea is that we still use interest points, although now the interest points are attached with attributes. For a detected line feature, we use as the attribute the direction of the line. By using the attributed point features, the false alarm rate is reduced.

For a line segment detected in the image, we can use (1) the endpoints of the line segment, or (2) the midpoint of the line segment as the point features. The direction of the line segment is used as its attribute. In the case of (1) above, there will then often be two or more features at the same location with different orientations, since line segment endings often occur at vertices.

For simplicity, we assume that the directional information is independent of the positional information, so that the probability density functions are separable. Thus the orientation attribute is used to modify the weight computed by the positional information of the point features, by adding on a term that penalizes for orientation inaccuracies.

If we use the endpoints of the line segments (option (1)) as the attributed point features, the two features are not really independent of each other. However, in our experiments, this feature also works fine. The main disadvantage of this case is that the number of features is doubled compared to utilizing midpoints as the features.

Theoretically speaking, the use midpoints as the features (option (2)) is more reasonable, since given the information of one midpoint, we cannot predict the appearance of the other midpoints. However, the features will not be as stable in practice.

When using attributed point features, there is extra information attached to the basis points. To form a hypothesis, we only make use of the positional information of the selected basis. We may use the directional information of the selected basis to filter out unlikely hypotheses.

6 Experimental results

This section contains three sets of experiments. The first set of the experiment is based on synthesized images. Fifteen polygonal objects are stored in the model database. We compare the recognition results using three different methods:

1. High curvature points as features, without orientation attributes;
2. Endpoints of the line segments as the attributed point features, with a separate entry for each oriented segment emanating from point;
3. Midpoints of the line segments as the attributed point features, with a single entry using the line orientation as the attribute.

For each method, a hash table containing entries is built and recognition results are assessed. In the case of method (2), we ignore the lack of independence of the features. A test scene with an embedded model is generated. For each method, features are extracted from the test scene.

The program generates 25 trials randomly for each data set. The number of features in the model polygons varies from five to fourteen for method (1), five to thirteen for method (2), and ten to 26 for method (3). The size of the hash table for each method is shown in Table 1. Note that the size of hash table for method (2) becomes almost eight times as the size of the hash table for method (3).

Figure 2 shows the trial with highest vote for using high curvature points as features using method (1). Although there is a probe which can recognize the object correctly (see Figure 3), the total weight is a little lower than the weight for the probe in Figure 2.

Features	Size of hash table
(1) High curvature points	139676 bytes
(2) Endpoints of line segments	1522980 bytes
(3) Midpoints of line segments	195788 bytes

Table 1: The size of hash tables for different features extraction methods for simulated polygonal objects.

The result of using method (2) and method (3) for the same test scene is shown in Figures 4 and Figure 5 respectively. The results demonstrate the ability of reducing the false alarm rate by using attributed information. That is, only a few candidates pass the "filter" and need to be verified further. Consistent results are obtained with other test scenes.

The second set of the experiments uses real images which are taken from the street. Fourteen car models (containing seven kinds of cars) are used in the model database, which is shown as Figure 6. The models are built by scanning the pictures (using a Silver scanner) and removing the background. The the test image is a different, unretouched image, and was taken at different position as the one used for the corresponding model. Further, the test image (shown as Figure 7) is derived from a Sony CCD camera. The result of feature extraction is shown at the top right corner of Figure 8, using method (3). The experiment shows that the correct model is identified even under such complicated background conditions. The model was transformed with the parameters derived from the matching basis sets, and is shown overlaid on the original image in the top left corner of Figure 8.

Finally, we prepared a set of experiments using industrial parts imaged by a CCD camera. The collection of parts is shown in Figure 9. Using method (3) for feature extraction, a hash table database was created from the models without cleaning the images. Figure 10 shows an example of a successful recognition of an example of a model in a test scene, and Figure 11 shows an unsuccessful test. In the unsuccessful case, the difficulty is that the resolution of the embedded model is coarse due to its small size. Accordingly, the errors in position and orientation of the normalized features are relatively large, no matter which basis pair is chosen. This experiment underscores the need for stable and accurate feature extraction.

Figure 2: The probe with highest vote for using high curvature points as features. The organization of this figure is: (Top left) the recognition result of the probe with the highest vote overlaid on the original image; (Top right) feature extraction (dots) and the successful trial basis (dots bounded by a ranctangle); and (Bottom) the top nine model-basis combinations that received the most vote corresponding to this trial. The white bar shows the length-encoded weight for each model-basis with respect to the maximum of the weight received among all hypotheses. In this case, where point features are used without orientation attributes, the highest vote is for an incorrect match.

7 Conclusions

We have shown that the use of orientation information as an attribute to point features based on edge extraction can be used to significantly enhance the performance of geometric hashing as an object recognition system. In particular, we have seen that the orientation information filters out false recognitions. We have made use of a weighted voting scheme with a Bayesian foundation, incorporating a noise model for the feature components, including the orientation feature. Our experiments include large model databases, with varying numbers of features in each model.

Acknowledgements

The authors would like to thank Dr. Isidore Rigoutsos and Dr. Frank Chee-Da Tsai for helpful discussions.

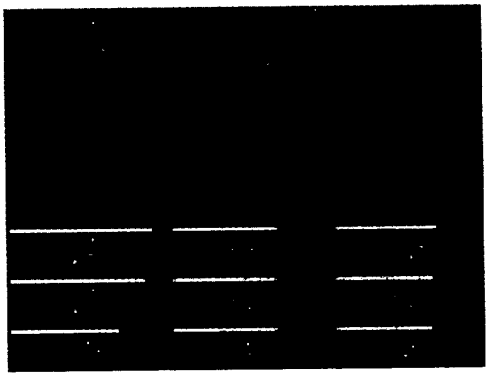

Figure 3: Again, using unattributed features (method (1) for the feature extraction). The correct match is shown, which in this case received a lower vote than the previous trial.

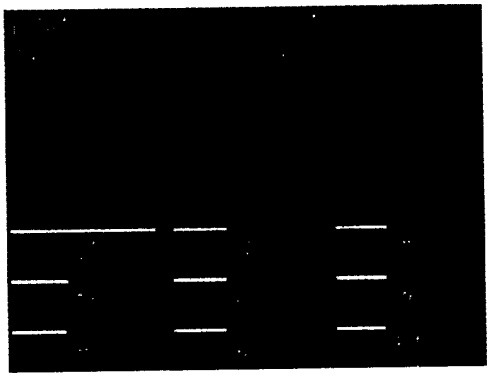

Figure 5: The correct recognition by using midpoints of line segments as features (Method (3)).

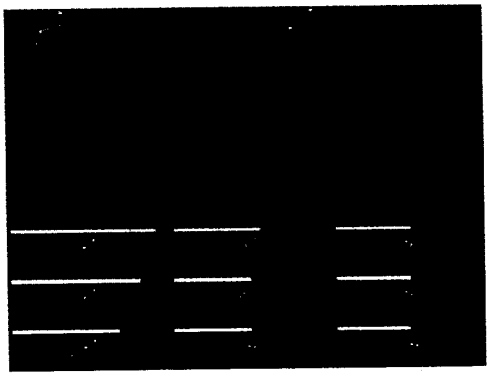

Figure 4: The correct recognition by using endpoints of line segments as features (Method (2)).

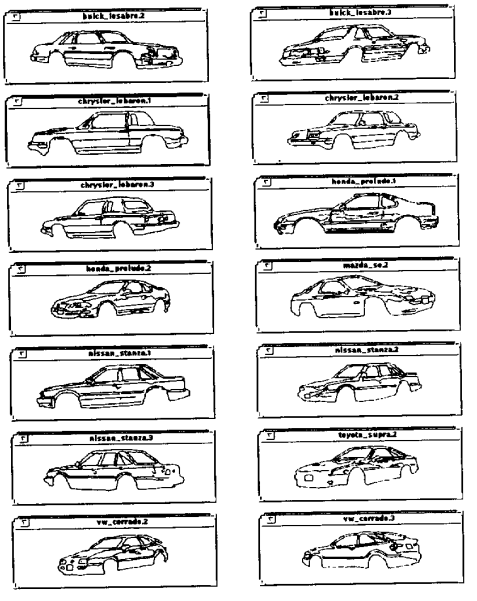

Figure 6: The fourteen car models used in the experiment.

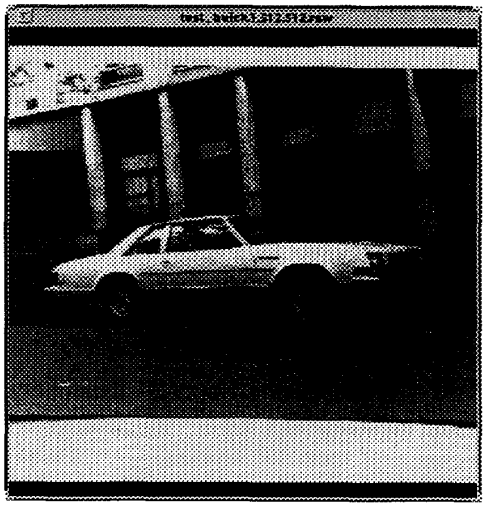

Figure 7: The test image—a Buick Lesabre.

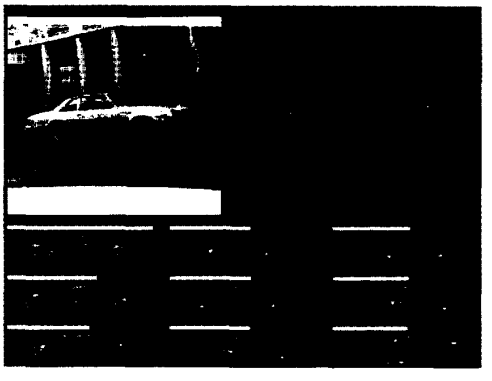

Figure 8: The correct recognition by using midpoints of line segments as features for test image Buick Lesabre.

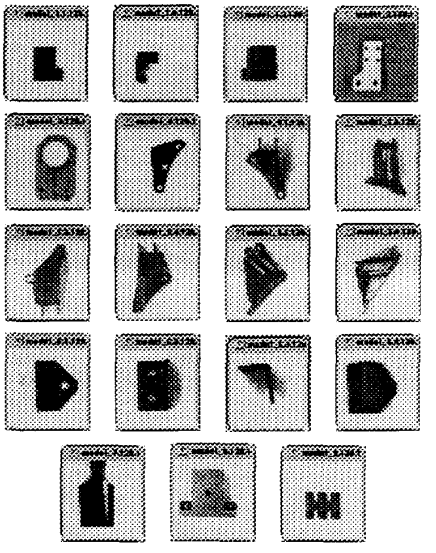

Figure 9: The collection of industrial parts. Some models have multiple stable states. For example, model 5 has six stable states which are shown as model_5.1 through model_5.6.

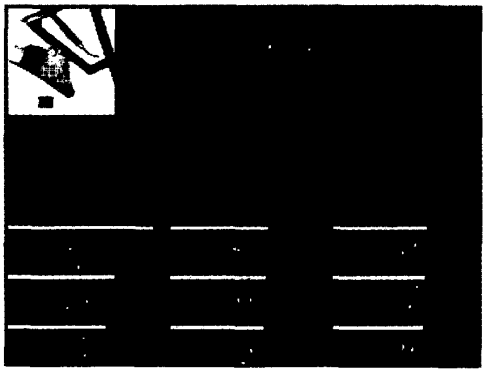

Figure 10: The correct recognition of an industrial part by using midpoints of line segments as features.

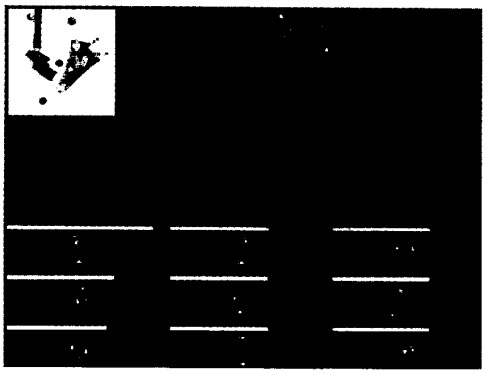

Figure 11: An incorrect recognition of the industrial part using midpoints of line segments as features.

References

[1] Andrea Califano and Ruud M. Bolle. The multiple window parameter transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(12):1157–1170, Dec. 1992.

[2] M. Costa, R. Haralick, and L. Shapiro. Optimal affine matching. In *Sixth Israeli Conference on Artificial Intelligence and Computer Vision*, Tel Aviv, Israel, Dec. 1989.

[3] D. M. Gavrila and F. C. A. Groen. 3D object recognition from 2D images using geometric hashing. *Pattern Recognition Letters*, 13(4):263–278, Apr. 1992.

[4] W. Grimson and D. Huttenlocher. On the sensitivity of geometric hashing. In *International Conference on Computer Vision*, pages 334–338, Osaka, Japan, Dec. 1990.

[5] André Guéziec and N. Ayache. Smoothing and matching of 3-D-space curves. In *Second European Conference on Computer Vision*, Santa Margherita Ligure, Italy, May 1992.

[6] Robert Hummel and Isidore Rigoutsos. Geometric hashing as a Bayesian maximum likelihood object recognition method. In preparation, 1993.

[7] Robert Hummel and Haim Wolfson. Affine invariant matching. In *Proc. DARPA IU Workshop*, pages 351–364, 1988.

[8] E. Kishon and H.J. Wolfson. 3-D curve matching. In *AAAI Workshop on Spatial Reasoning and Multisensor Fusion*, pages 250–261, 1987.

[9] Yehezkel Lamdan. *Geometric Hashing.* PhD thesis, New York University, June 1989.

[10] Yehezkel Lamdan, Jacob T. Schwartz, and Haim J. Wolfson. Object recognition by affine invariant matching. In *Proc. Computer Vision and Pattern Recognition*, pages 335–344, 1988.

[11] Yehezkel Lamdan and Haim J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proc. Second International Conference on Computer Vision*, pages 238–249, 1988.

[12] Isidore Rigoutsos. *Massively Parallel Bayesian Object Recognition.* PhD thesis, New York University, July 1992.

[13] Isidore Rigoutsos and Robert Hummel. Implementation of geometric hashing on the Connection machine. In *IEEE Workshop on Directions in Automated CAD-Based Vision*, Maui, Hawaii, June 1991.

[14] Isidore Rigoutsos and Robert Hummel. Massively parallel model matching: Geometric hashing on the connection machine. *IEEE Computer: Special Issue on Parallel Processing for Computer Vision and Image Understanding*, Feb. 1992.

[15] Frank Chee-Da Tsai. Robust affine invariant matching with application to to line features. In *Computer Vision and Pattern Recognition*, pages 393–399. IEEE Computer Society, June 1993.