Affine Invariant Model-Based Object Recognition

YEHEZKEL LAMDAN, STUDENT MEMBER, IEEE, JACOB T. SCHWARTZ, MEMBER, IEEE, AND HAIM J. WOLFSON, MEMBER, IEEE

Abstract—**We describe new techniques for model-based recognition of flat objects in 3-D space. The recognition is performed from single gray scale images taken from unknown viewpoints. The objects in the scene may be overlapping and partially occluded. An efficient matching algorithm, which assumes affine approximation to the perspective viewing transformation, is proposed. The algorithm has an off-line model preprocessing (shape representation) phase, which is independent of the scene information, and a recognition phase, based on efficient indexing. It has a straightforward parallel implementation. The algorithm was successfully tested in recognition of industrial objects appearing in composite occluded scenes.**

I. Introduction

RECOGNITION of industrial parts and their location in a factory environment is a major task in robot vision. Most industrial part recognition systems are model-based systems (see a survey in [1]). The model-based approach is well suited for an industrial environment since the objects processed by the robot are usually known in advance and belong to a certain subset of the factory's tools and products.

We discuss the object-recognition problem, where the robot is faced with a composite scene of overlapping parts (thus partially occluding each other), taken from a database of known objects (e.g., the factory's warehouse). The task is to recognize the objects in the scene and their location.

No restriction on the viewing angle of the camera is assumed. In this paper, we discuss the recognition of flat objects arbitrarily positioned in space. However, our method can be extended to recognition of 3-D objects from single 2-D images, and we have already obtained some initial results in this direction (see [2]). The recognition is done from 2-D intensity images. The algorithms we describe were actually tested in a "real life situation" by recognition of objects comprising composite scenes of industrial tools, such as pliers, wrenches, etc. (see Figs. 3–7).

Since we are concerned with recognition of partially occluded objects, no use of global features can be made. We describe our objects by sets of local features. These features can be points, line segments [3], curve segments [4], etc. In this paper, we restrict ourselves to the use of points, which we denote as *interest points*. The point sets of the various model objects are matched simultaneously against the point set of the composite overlapping scene using a small number of corresponding points. Once a candidate correspondence was established, we find the best transformation in the least-squares sense to establish the correct position of the candidate model object in the scene image. This candidate match is verified by matching all the relevant model features with the corresponding scene features. A key factor in our scheme is its division into a preprocessing stage and a recognition stage. Our model point sets are preprocessed off line, independent of the scene information, thus enabling an efficient on-line recognition stage. A major advantage of the proposed matching algorithm is its straightforward parallelism both in the preprocessing and recognition stages.

Much work was done on recognition of overlapping objects in 2-D scenes. Ayache and Faugeras [5] developed the HYPER system for recognition of 2-D objects from 2-D images. This system models objects as polygons and performs *privileged* segment matching, exploiting angle constraints. Since angles and length ratios are not affine invariant properties, the system cannot be naturally extended to deal with affine invariant object recognition. The local feature focus (LFF) system [6] performs 2-D object recognition by using local geometric constraints. A later system (3DPO) [7] works in a similar fashion and is used for recognition of 3-D objects from range data. Both systems compute local features and perform matching of distinguished features, which are labeled by means of their neighboring features. Matching is done by a graph search algorithm, which is inherently NP complete. Since the geometric constraints used are distance and angle, the systems performance is not affine invariant. The recognition and attitude finder (RAF) system, which is proposed in [8] and [9], also applies local geometric constraints. RAF utilizes simple distance and angle constraints to find consistent model and image features. The search is organized as a tree, which makes its complexity inherently exponential. In addition, the geometric constraints used by RAF are not affine invariant.

Turney *et al.* [10] propose a recognition technique of 2-D objects that have undergone translation and rotation by matching their boundary curves. They match subtemplates of model object boundary curves to subtemplates of the boundary curve of an observed scene. The object transformation is

Manuscript received July 27, 1988; revised May 30, 1990. This work was supported by the Office of Naval Research under Grant N00014-82-K-0381 and N00014-85-K-0077, Work Unit NR 4007006, and National Science Foundation Grant NSF-DCR-83-20085.

Y. Lamdan and J. Schwartz are with the Robotics Research Laboratory, Courant Institute of Mathematical Sciences, New York University, New York, NY 10003.

H. J. Wolfson is with the Computer Science Department, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv, Israel.

IEEE Log Number 9038813.

recovered by Hough-type clustering of the transformation parameters for the matching subtemplates. A somewhat similar but computationally more efficient curve matching algorithm, which is suitable for recognition of 2-D from 2-D and 3-D from 3-D is proposed by Kalvin *et al.* [11] and Hong and Wolfson [12]. Rather then match a given image curve to all model curves, matching is attempted only for candidate model curves, which are 'similar' enough to the image curve. This is facilitated by a hash table, which is created in a preprocessing stage. The above-mentioned methods are limited to objects that can be well modeled by curve segments and can deal only with translation and rotation.

Cyganski, Orr, and their colleagues [13], [14] developed affine invariant curve matching methods. However, they use global region information; hence, these methods are not applicable for recognition of partially occluded objects. Another global affine invariant recognition technique was proposed by Hong and Tan [15]. Bamieh and De Figueiredo [16] suggest a 3-D object recognition method based on general moment invariants of 2-D affine transformations. The model database consists of polyhedral objects. The objects are represented by an attributed graph describing the connectivity of the faces along with the moment invariants and the angles between the normals to the surfaces. Recognition is facilitated by matching model attributed graphs against attributed graphs extracted in a similar fashion from the image. In order to compute moment invariants of a given face, that face has to be fully visible, which limits the ability of the algorithm to cope with occlusion. The matching algorithm, which performs subgraph isomorphism, is inherently exponential and thus is not suitable for recognition in complex images. Even though an extension of the algorithm is suggested to deal with objects that have arbitrarily shaped flat faces, it is mainly suitable for recognition of polyhedral objects.

Additional examples of 2-D recognition techniques can be found in [1]. We refer to [1] and [17] for a comprehensive survey of existing 3-D object recognition systems. More recent results that are relevant to this paper are [18]–[20]. The SCERPO system of Lowe [18] recognizes 3-D objects from 2-D intensity scenes. It is viewpoint invariant and is based on perceptual grouping of object features. At its present stage, however, the SCERPO system is mainly suited for polyhedral scenes. Thompson and Mundy [19] use a clustering (Hough-transform) approach to discover the transformation between the model and the scene images. Huttenlocher and Ullman [20] proposed the so-called *alignment* technique, which is based on matching of minimal sets of features in a model and the scene. They try an exhaustive match of all such minimal sets describing the transformation. Our task is similar in its basic assumptions to that described in [20]. Both methods, which have been developed independently, are trying to solve the same problem under the same assumptions. However, our approach and the approach in [20] are somewhat complementary. In [20], there is an emphasis on the classification of the model and image features to reduce the complexity of matching, whereas the matching algorithm itself is straightforward. We, on the other hand, consider the case where no such effective classification can be done (this is also the assumption in [19]), and hence, our emphasis is on the development of an efficient feature matching algorithm, which processes the models and the scene images independently, allowing fast recognition. This is achieved using indexing by affine invariants. In case feature classification is possible, it can be incorporated in our algorithm in a natural way to improve its efficiency.

This paper reports results in a series of experiments to develop a 3-D object-recognition system from 2-D images, based on efficient point, line, and curve matching procedures (see also [2] and [4]).

II. Definition of the Problem

Consider the problem of object recognition in a cluttered 3-D scene. The models of the objects to be recognized are assumed to be known in advance. The objects in the scene may overlap and may also be partially occluded by other (unknown) objects (see Fig. 6(a)). We allow the image to be obtained from an arbitrary viewpoint. At this stage, we will assume that we are dealing with flat objects. These initial assumptions are similar to those in [20]. We also assume that the depth of the centroids of the objects in the scene is large compared to the focal length of the camera and that the depth variation of the objects is small compared with the depth of their centroids. Under these assumptions, it is well known that the perspective projection is well approximated by a parallel projection with a scale factor (see, for example, p. 79 of [21] or [19], [22]). Hence, two different images of the same flat object are in an affine 2-D correspondence, namely, there is a nonsingular 2×2 matrix $\boldsymbol{A}$ and a 2-D (translation) vector $\boldsymbol{b}$, such that each point $\boldsymbol{x}$ in the first image is translated to the corresponding point $\boldsymbol{Ax} + \boldsymbol{b}$ in the second image.

Our problem is to recognize the objects in the scene and for each recognized object to find the affine transformation that gives the best least-squares fit (see Section VI for details) between the model of the object and its transformed image in the scene.

III. Choice of 'Interest Points'

Our matching algorithm, which is described in the next section, extracts so-called interest points, both in the object model images and in the scene image to find the best match between those point sets. We do not try, at this stage, to optimize the point extraction methods. These should be database dependent so that different databases of models will suggest different natural interest points. For example, a database of polyhedral objects naturally suggests the use of polyhedra vertices as interest points, whereas 'curved' objects suggest the use of sharp convexities, deep concavities, and zero curvature points. Interest points do not have to appear physically in the image. For example, a point may be taken as the intersection of two nonparallel line segments, which are not necessarily touching. An interest point does not necessarily have to correspond to a geometrical feature. An interest operator based on high variance in intensity is described in [23] and was used in [24].

Our basic assumption is that enough interest points can be extracted in the relevant images. We assume no special classification of these points. It is important to observe that since our matching algorithm is designed to deal with missing and spurious points, we do not require foolproof performance of the interest point operator.

In our experiments, we used points of sharp convexities and deep concavities (see Figs. 3(c), 3(d), 4(b), 5(b), 6(b), and 7(b)). The extraction of these points is described in Section XI.

IV. Recognition of a Single Model in a Scene

For the sake of clarity, we describe our algorithm in the simpler situation, where the database consists only of one model. However, the presentation given here applies to the general case, where a number of models may appear in the scene.

As was mentioned in the previous section, the models and the scene are described by sets of interest points. Hence, we may rephrase the model-based recognition problem to the point-set matching task, namely, is there a (large enough) affinely transformed subset of a model point set that matches a subset of the scene point set?

Given m points on a model and n points in the scene, there are $O(n^m)$ ways to match the model points to the scene points. Since such an exponential complexity is unacceptable for object recognition algorithms, various attempts were made to prune the space of possible matches. Some of them try to employ efficient tree search techniques, where the pruning is based on geometric constraints (see, for example, [8]). However, the search still remains exponential in the number of scene features for recognition of partially occluded objects (see [25]).

To overcome this exponential complexity, one may observe that a transformation of a rigid object is usually defined by the transformation of a small number of the object's points. This geometric observation is at the core of the so-called *pose clustering* [26] and *alignment* [20] techniques. For example, it is well known that an affine transformation of the plane is uniquely defined by the transformation of three ordered noncollinear points (see [27]). Moreover, there is a unique affine transformation that maps any ordered noncollinear triplet in the plane to another ordered noncollinear triplet. Hence, we may extract interest points on the model and the scene and try to match noncollinear triplets of such points to obtain candidate affine transformations. Each such transformation can be checked by matching the transformed model against the scene. This is the basic approach in [20].

However, the complexity of such a scheme is quite unfavorable. Given m points in the model and n points in the scene, the worst-case complexity is $(m \times n)^3 \times t$, where t is the complexity of matching the model against the scene. If we assume that m and n are of the same magnitude and t is at least of magnitude m, the worst case complexity is of order n^7. One way to reduce this complexity [20] is to classify the points in a distinctive way so that each triplet can match only a small number of other triplets. We consider, however, the situation where such a distinction does not exist or cannot be made in a reliable way (see [19]). Hence, we present a more efficient triplet matching algorithm. In addition, in Section VIII, we show how, in special cases, the complexity can be reduced by using certain affine metric invariants.

The algorithm consists of two major steps. The first one is a preprocessing step, which is applied to the model points. This step does not use any information about the scene and is executed off line before actual matching is attempted. The second step (matching proper) uses the data prepared by the first step to match the models against the scene. The execution time of this second step is the actual recognition time.

In order to separate the computation into such two independent steps, we have to represent the model and scene point information in a way that is independent and still allows comparison of corresponding triplets. The fact that an affine transformation is uniquely defined by the transformation of three noncollinear points can also be expressed as follows. Consider a set of m points, and pick any ordered subset of three noncollinear points. The two linearly independent vectors based on these points are a 2-D linear basis. One can express the coordinates of all the model points in this basis. (The basis points will have the coordinates $(0,0)$, $(1,0)$, $(0,1)$, respectively.) Any affine transformatin applied to the set points will not change the set of coordinates based on the same ordered basis triplet. Specifically, let $\boldsymbol{e}_{00}$, $\boldsymbol{e}_{10}$, $\boldsymbol{e}_{01}$ be an ordered affine basis triplet in the plane. Then, the affine coordinates (α, β) of a point $\boldsymbol{v}$ are

$$\boldsymbol{v} = \alpha(\boldsymbol{e}_{10} - \boldsymbol{e}_{00}) + \beta(\boldsymbol{e}_{01} - \boldsymbol{e}_{00}) + \boldsymbol{e}_{00}.$$

Application of an affine transformation $\boldsymbol{T}$ will transform the point $\boldsymbol{v}$ to

$$\boldsymbol{Tv} = \alpha(\boldsymbol{Te}_{10} - \boldsymbol{Te}_{00}) + \beta(\boldsymbol{Te}_{01} - \boldsymbol{Te}_{00}) + \boldsymbol{Te}_{00}.$$

Hence, $\boldsymbol{Tv}$ has the same coordinates (α, β) in the basis triplet $\boldsymbol{Te}_{00}$, $\boldsymbol{Te}_{10}$, $\boldsymbol{Te}_{01}$.

Our algorithm will efficiently compare these sets of coordinates belonging to different bases.

A. Preprocessing

Assume that we are given an image of a model where m interest points have been extracted. For each ordered noncollinear triplet of model points, the coordinates of all other $m - 3$ model points are computed taking this triplet as an affine basis of the 2-D plane. Each such coordinate (after a proper quantization) is used as an entry to a hash table, where we record the *basis-triplet* at which the coordinate was obtained as was the model (in case of more than one model). This encoding of each point in all possible affine basis coordinates gives us an affine invariant representation of the m-point set, which will enable efficient indexing in the recognition stage. The complexity of the preprocessing step is of order m^4 per model. New models added to the database can be processed independently without recomputing the hash table.

B. Recognition

In the recognition stage, we are given an image of a scene where n interest points have been extracted. We choose an arbitrary ordered triplet in the scene and compute the coordinates of the scene points taking this triplet as an affine basis. For each such coordinate, we check the appropriate entry in the hash table, and for every pair (model, basis triplet) that appears there, we tally a vote for the model and the basis triplet as corresponding to the triplet that was chosen in the scene. (If there is only one model, we have to vote for the basis triplet alone).

If a certain pair (model, basis triplet) scores a large number of votes, we decide that this triplet corresponds to the one chosen in the scene. The uniquely defined affine transformation between these triplets is assumed to be the transformation between the model and the scene. This candidate transformation is then verified in two successive verification steps, where the first is based on the best least-squares match of all the candidate model points (see Section VI), and the second is based on direct verification of all the model edges under the appropriate transformation versus the scene edges (this time, all the edges and not only interest points, are considered). If the current triplet does not score high enough, we pass to another basis triplet in the scene.

For the algorithm to be successful, it is enough, theoretically, to pick three noncollinear points in the scene belonging to one model. The voting process per triplet is linear in the number of points in the scene. Hence, the overall recognition time is dependent on both the number of model points in the scene and the number of additional interest points that belong to the scene that did not appear on any of the models. Although, in the worst case, we might have an order of n^4 operations, in most cases, especially when the number of models is small, the algorithm will be much faster. For example, if there are k model points in a scene of n points, the probability of not choosing a model triplet in t trials is approximately

$$p = \left(1 - \left(\frac{k}{n}\right)^3\right)^t.$$

Hence, for a given $\epsilon \leq 1$, if we assume a lower bound on the 'density' $d = k/n$ of model points in a scene, the number of trials t giving $p < \epsilon$ is of order $\log \epsilon / \log(1 - d^3)$, which is a constant independent of n. Since the verification process is linear in n, we have, in this case, an algorithm of complexity $O(n)$, which will succeed with probability of at least $1 - \epsilon$.

The method, the way it is presented here, assumes no *a priori* classification of the model and scene points to achieve matching candidates. In practical applications, some additional information might be available about the interest points. For example, points of sharp concavities and points of sharp convexities belong to different classes. The hash table can be easily adjusted to accommodate such qualitative information, thus improving the recognition speed.

A major potential advantage of the suggested algorithm is its high inherent parallelism. Parallel implementation of this algorithm is straightforward; moreover, it should be quite easy to build a special device for this implementing it at very high speed.

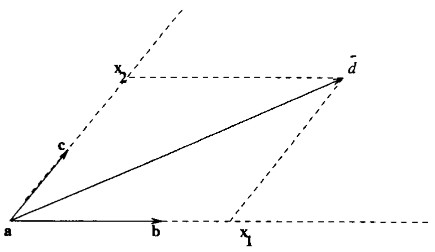


Fig. 1. Coordinates of the point $\tilde{d}$ in the affine basis triplet $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$.

V. Error Analysis

So far, we have discussed our algorithm in its 'noiseless' version. However, since real scene measurements are noisy, this noise will affect both the implementation of the algorithm and its performance. We give a short description of the issues raised by noise analysis and their handling by our method. A more extensive noise analysis is given in [28].

We assume that the models can be acquired under 'ideal' circumstances (from a CAD model, for example); hence, the preprocessing step is noiseless. In the recognition step, image coordinates of interest points are measured. These coordinates are represented by 2-D vectors. One may define a norm on this 2-D vector space. We will usually use either the Euclidean L_2 or the maximum coordinate L_∞ norm. Assume that image point measurements introduce an error of at most ϵ in the given norm.

The computation of the coordinates of an interest point $\tilde{\boldsymbol{d}} = (\tilde{d}_1, \tilde{d}_2)$ in the affine basis $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ can be formulated as a solution of the linear system of two equations in two unknowns $A\boldsymbol{x} = \boldsymbol{d}$. If $\boldsymbol{a}$ is the origin of the affine basis triplet (see Section IV), the two columns of the matrix A are the difference vectors of the basis interest points $\boldsymbol{b} - \boldsymbol{a}$, and $\boldsymbol{c} - \boldsymbol{a}$, respectively, and the free vector is $\boldsymbol{d} = \tilde{\boldsymbol{d}} - \boldsymbol{a}$. These vectors are represented in image coordinates, whereas the solution vector $\boldsymbol{x}$ gives the representation of the point $\tilde{\boldsymbol{d}}$ in the affine basis $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ coordinates (see Fig. 1).

Taking the errors into account, our task can be formulated as the solution of the following linear system:

$$(A + \delta A)(x + \delta x) = d + \delta d \qquad (1)$$

where δA, δx, and δd are the errors of the matrix A and the vectors x and d, respectively. By the nature of our point measurements, we may assume that the absolute values of entries of the matrix δA and the vector δd are less than some given measurement error ϵ. An extensive treatment of such 'approximate' linear systems is given in [29]. Let us just mention the following well-known result from standard numerical analysis (see, for example, p. 25 of [30]) stating that

$$\frac{\|\delta \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \kappa(A)\left[\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta \boldsymbol{d}\|}{\|\boldsymbol{d}\|}\right] + O(\epsilon^2) \qquad (2)$$

where $\kappa(A) = \|A\| \, \|A\|^{-1}$ is the condition number of the matrix A. The above inequality holds for any vector norm and its appropriate matrix norm.

Inequality (2) gives an estimate of the maximal relative error that can be introduced by the image measurement noise into the coordinates of the hash table address $\boldsymbol{x}$. Hence, the voting procedure reflects this noise. For an address $\boldsymbol{x}$, all the bins with addresses in the $\delta \boldsymbol{x}$ neighborhood of $\boldsymbol{x}$ participate in the voting. This ensures that votes for a correct model basis are not missed due to noise. In practice, tighter bounds usually apply [29]. Note that the amount of error ϵ is defined by the known imaging process; hence, the worst-case δA and $\delta \boldsymbol{d}$ can be computed in advance. Since for a given basis triplet $\kappa(A)$ can also be computed, we can evaluate in advance the relative merit of voting for a given basis triplet and eliminate those triplets that are going to introduce excessive noise. It should be noted that if a certain basis triplet belonging to some model was not chosen in the image (or did not get enough votes), we still have a chance to recover this model from another basis triplet.

Since, as we have shown, the appropriate voting bins for each address can be evaluated in advance, we do not expect a correct basis triplet to achieve less votes than the corresponding number of unoccluded model points. There still remains the possibility of a 'random' basis triplet achieving a large number of votes. Such a 'wrong' candidate will be discovered by two verification procedures that are incorporated in the algorithm and described in the following sections (see (4) and (5) of the Summary of the Algorithm in Section VII). Although 'wrong' candidates will be discovered in the verification steps and discarded, we would still like to show that the probability of a 'random' configuration to get a high vote is small. This probabilistic analysis is out of the scope of this paper and will be presented in [28].

VI. Finding the Best Least-Squares Match

As we have mentioned before, an affine transformation in the plane is uniquely defined by the transformation of three noncollinear points. However, in practical applications, this transformation may be somewhat distorted because of noisy computation of these three points. Knowledge of additional points, which were transformed to each other, may help us to improve the accuracy of the computed transformation. In this section, we show how to compute an affine transformation, giving the best least-squares match between sets of points that are transformed to each other. In [31], this problem was solved for Euclidean transformations. Our presentation generalizes the solution to affine transformations.

Specifically, assume that we are looking for an affine match between the sequences of planar points $(\boldsymbol{u}_j)_{j=1}^n$ and $(\boldsymbol{v}_j)_{j=1}^n$. We would like to find the affine transformation $\boldsymbol{Tu} = \boldsymbol{Au} + \boldsymbol{b}$ of the plane that will minimize the l^2 distance between the sequences $(\boldsymbol{Tu}_j)_{j=1}^n$ and $(\boldsymbol{v}_j)_{j=1}^n$:

$$\delta = \min_{T} \sum_{j=1}^{n} |\boldsymbol{Tu}_j - \boldsymbol{v}_j|^2.$$

Without loss of generality, we may translate the set $\boldsymbol{u}_j$ so that

$$\sum_{j=1}^{n} \boldsymbol{u}_j = 0.$$

Then

$$\begin{aligned}\delta &= \min_{A,b} \sum_{j=1}^{n} |\boldsymbol{Au}_j + \boldsymbol{b} - \boldsymbol{v}_j|^2 \\ &= \min_{A,b} \left[\sum_{j=1}^{n} |\boldsymbol{b} - \boldsymbol{v}_j|^2 + \sum_{j=1}^{n} |\boldsymbol{Au}_j|^2 \right. \\ &\qquad \left. + 2\sum_{j=1}^{n} \boldsymbol{b} \cdot \boldsymbol{Au}_j - 2\sum_{j=1}^{n} \boldsymbol{Au}_j \cdot \boldsymbol{v}_j \right].\end{aligned}$$

However

$$\sum_{j=1}^{n} \boldsymbol{b} \cdot \boldsymbol{Au}_j = \boldsymbol{b} \cdot \boldsymbol{A}\left(\sum_{j=1}^{n} \boldsymbol{u}_j\right) = 0.$$

Hence, $\boldsymbol{b}$ and $\boldsymbol{A}$ appear independently in δ, and we can minimize their contributions separately.

To minimize over $\boldsymbol{b}$, we simply put

$$\boldsymbol{b} = \frac{1}{n} \sum_{j=1}^{n} \boldsymbol{v}_j.$$

As to $A = (a_{ij})$, $(i = 1,2;\ j = 1,2)$, denote

$$\begin{aligned}g(A) &= g(a_{11}, a_{12}, a_{21}, a_{22}) \\ &= \sum_{j=1}^{n} |\boldsymbol{Au}_j|^2 - 2\sum_{j=1}^{n} \boldsymbol{Au}_j \cdot \boldsymbol{v}_j.\end{aligned}$$

We have to find

$$\min_{A} g(A) = \min g(a_{11}, a_{12}, a_{21}, a_{22}).$$

To find this minima, one has to solve the following system of four equations $(i = 1,2;\ j = 1,2)$

$$\frac{\partial g}{\partial a_{ij}} = 0. \qquad (*)$$

Since g is a quadratic function in each of its unknowns, $(*)$ is a system of four linear equations with four unknowns. (Actually, these happen to be two independent sets of two linear equations with two unknowns.) Omitting the tedious details, we present the final solution of this system.

Since $\boldsymbol{u}_j = (u_j^1, u_j^2)$ and $\boldsymbol{v}_j = (v_j^1, v_j^2)$ are two-dimensional vectors for each $j = 1, \cdots, n$, we may define the following four n-dimensional vectors for $i = 1,2$:

$$U^i = (u_j^i)_{j=1}^n$$
$$V^i = (v_j^i)_{j=1}^n.$$

Then, the solution of $(*)$ is given by

$$a_{11} = \frac{(U^1 \cdot V^1)(U^2 \cdot U^2) - (U^2 \cdot V^1)(U^1 \cdot U^2)}{\Delta}$$

$$a_{12} = \frac{(U^1 \cdot U^1)(U^2 \cdot V^1) - (U^1 \cdot V^1)(U^1 \cdot U^2)}{\Delta}$$

$$a_{21} = \frac{(U^1 \cdot V^2)(U^2 \cdot U^2) - (U^1 \cdot U^2)(U^2 \cdot V^2)}{\Delta}$$

$$a_{22} = \frac{(U^1 \cdot U^1)(U^2 \cdot V^2) - (U^1 \cdot V^2)(U^1 \cdot U^2)}{\Delta}$$

where

$$\Delta = (U^1 \cdot U^1)(U^2 \cdot U^2) - (U^1 \cdot U^2)(U^1 \cdot U^2)$$

($\cdot$ is the vector dot product.)

As we can see, Δ is dependent only on one set of points (in this case, the model points); therefore, we can know in advance which sets of model points will give a stable solution for the minima.

In Fig. 5(c) we see an example of a fit obtained by calculating the affine transformation from three basis points, and in Fig. 5(d), the same model is fitted using the best least-squares affine match, which is based on ten points, all of which were recovered as corresponding points by the transformation in Fig. 5(c).

VII. Summary of the Algorithm

Our algorithm can be summarized as follows:

1) Represent the model objects by sets of interest points.
2) For each noncollinear triplet of model points, compute the coordinates of all the other model points according to this basis triplet, and hash these coordinates into a table that stores all the pairs (model, basis triplet) for every coordinate.
3) Given an image of a scene, extract its interest points, choose a triplet of noncollinear points as a basis triplet, and compute the coordinates of the other points in this basis. For each such coordinate, vote for the pairs (model, basis triplet), and find the pairs that obtained a large number of votes. For each such pair, its model and the affine transformation between its basis triplet and the image basis triplet become candidates for the verification steps 4) and 5). If there is no high scoring (model, basis triplet), pair, continue by checking another basis triplet in the image.
4) For each candidate model and affine transformation from the previous step, establish a correspondence between the model points and the appropriate scene points, and find the affine transformation giving the best least-squares match for these corresponding sets. If the least-squares difference is too big, go back to step 3) for another candidate triplet.
5) Finally, the model edges are transformed according to the transformation of step 4) and compared with the scene edges (this time, we are not considering previously extracted interest points only). If this comparison gives a bad result, go back again to step 3). (In our experiments, we compared the boundaries of our objects at equally spaced sample points.)

This is a short summary of the basic algorithm. It is schematically represented in Fig. 2. Of course, various improvements can be incorporated in its different steps, e.g., the complexity reduction described in Section VIII.

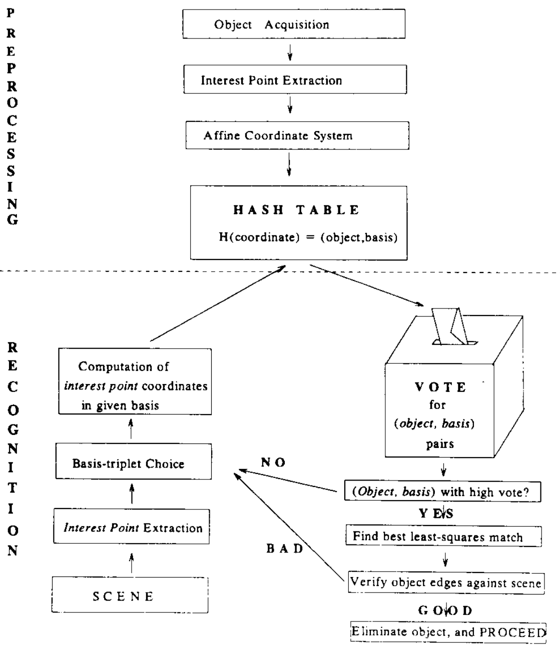

PREPROCESSING
Object Acquisition
Interest Point Extraction
Affine Coordinate System
HASH TABLE
H(coordinate) = (object,basis)
RECOGNITION
Computation of *interest point* coordinates in given basis
Basis-triplet Choice
Interest Point Extraction
SCENE
VOTE for (*object, basis*) pairs
(*Object, basis*) with high vote?
NO
YES
Find best least-squares match
BAD
Verify object edges against scene
GOOD
Eliminate object, and PROCEED

Fig. 2. Scheme of the algorithm.

VIII. Reduction of Complexity Using Affine Invariants

When the number of interest points on the models is large, various affine invariants can be exploited to reduce the complexity of the method presented in Section IV. We give one such example. We will use the following:

Lemma (see, for example, p. 73 of [21]): Two straight lines that correspond in an affine transformation are 'similar', i.e., corresponding segments on the two lines have the same length ratio.

Moreover, the same statement holds for sets of parallel lines. Hence, if we have a set of points located on parallel lines in a model and another set of points on parallel lines in the scene, we can efficiently check the conjecture that some of these points correspond.

Let us see how the method described in Section IV can be reduced for the case of a one-dimensional line.

We again have two major steps.

A. Model Preprocessing

Extract the interest points on the model and find those that are positioned on the same line. (A point may belong to different lines.) Take a pair of points on a line and compute the coordinates of all other model points on this line, taking this pair to be the standard one-dimensional basis of the line. Each such coordinate is used as an entry to a hash table, where we record the basis pair at which the coordinate was obtained, the line, and the model.

B. Recognition

Extract sets of points positioned on the same line in the image. Choose a pair of points on such a line as a basis and compute the coordinates of the other points on the same line according to this basis. For each such coordinate, check the appropriate entry in the hash table, and vote for every triple of (model, line, basis pair) that appears there. A triple that scores a large number of votes gives the correspondence between the points on the appropriate lines. Correspondence of three noncollinear points (obtained from different lines, of course) already gives a full affine basis, and we proceed as before.

Since we have to choose only two points as a basis and not three, the worst-case complexity is reduced by a factor of n.

IX. Recognition of Objects Under Similarity Transformation

In numerous vision problems, we are confronted with the problem of recognition of objects that have undergone a similarity transformation, namely, rotation, translation, and scale. This is the situation in which the viewing angle of the camera is the same for both the model and the image of a scene. Such conditions can be achieved, for example, in a factory environment where the viewing angle of a camera on a conveyor belt can be kept constant.

Our algorithm is obviously applicable to the case of a similarity transformation since it is a restricted case of an affine transformation. Moreover, since the similarity case is simpler than the general affine case, the complexity of both the preprocessing and recognition stage can be reduced. The key observation here is that since the similarity transformation is orthogonal, two points are enough to form a basis that spans the 2-D plane. (The first point is assigned coordinates $(0,0)$ and the second $(1,0)$. The third basis point $(0,1)$ is uniquely defined by these two points.) Hence, we may repeat the procedure described in Section IV using basic pairs instead of basic triplets. This reduces the complexity of the preprocessing step by a factor of m and the worst-case complexity of the recognition step by a factor of n.

X. Line Matching

In the previous sections, we dealt with point matching algorithms. However, extraction of points might be quite noisy. A line is a more stable feature than a point. Thus, in scenes, in which lines can be extracted in a reliable way, e.g., scenes of polyhedral objects, we might be interested to apply similar procedures to lines.

All the point matching techniques of Section IV apply directly to lines since lines can be viewed as points in the dual space. Thus, three lines, which have no parallel pair, are a basis of the affine space, each line has unique coordinates in this basis, and we repeat exactly the matching procedure of Section IV.

We can also make use of line segments to reduce the complexity of the matching algorithm of Section IV. If the endpoints of line segments can be reliably extracted, then instead of a triplet of points or lines as a basis, we can take a line segment plus an additional point. The reduction of complexity is significant.

Since an affine transformation maps collinear points into collinear points and points of line intersection into points of the same line intersection, we may develop algorithms that combine point and line information. For example, even if the algorithm utilizes point triplets as an affine basis, the verification can be done not only on other interest point coordinates but also on line equations, etc.

XI. Experimental Results

We have done several experiments of recognizing models in composite scenes. Models of pliers and wrenches have been used (Fig. 3(a) and (b)). These objects are actually three dimensional, but they have a good 2-D approximation. The model objects have one degree of freedom, namely, the angle between their handles. However, this free parameter was kept constant for each model object throughout the experiments. (A natural extension of our methods to deal with parametrized bodies is under current investigation.) We attempted to recognize these models in several composite scenes (see, for example Figs. 4, 6, and 7).

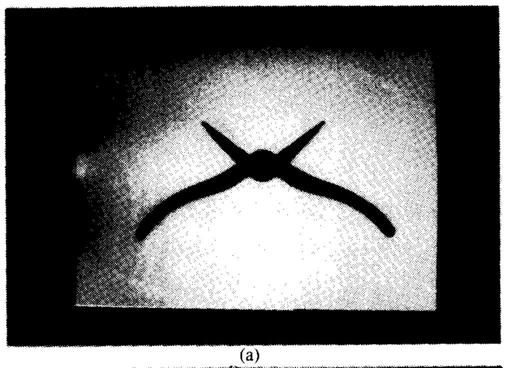
Gray-level images of models and scenes were taken using a Vidicon camera. The objects were put on a flat surface. While the camera was kept at a fixed angle, the surface slant and tilt have been changed (this is equivalent to changing the camera viewpoint). In addition, the distance between the camera and the surface was variable to induce scale changes. Hence, we were able to effect a variety of 2-D affine transformations of the model objects.

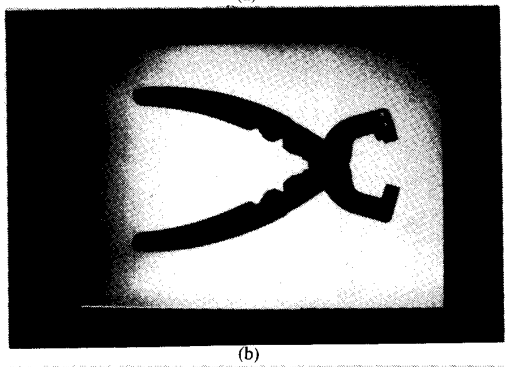
The model database was created by taking images of the models in which the camera viewing axis was perpendicular to the plane (Fig. 3(a) and (b)). However, since the recognition method is 2-D affine invariant, any other feasible camera viewpoint could be chosen.

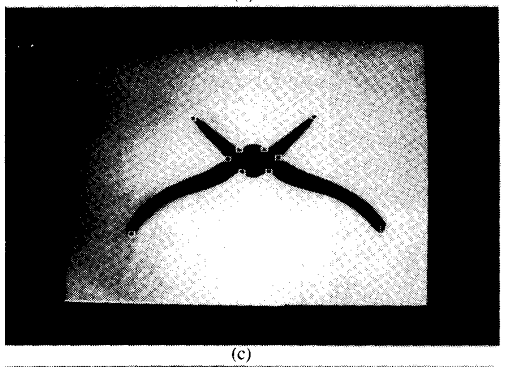
The images were digitized and stored as 512×512 pixel images using a Vicom image processor that was interfaced to SUN workstations. The images were segmented by intensity thresholding, and approximating polygonal boundary curves were extracted. Since the digitizing camera system introduces some noise, the extracted curves were smoothed using the smoothing algorithm described in [31]. The smoothing algorithm basically places a narrow band around the curve and finds the shortest path between the curve's endpoints that lie completely within the band. The band is just the ϵ neighborhood of the curve. The parameter ϵ is chosen in a manner reflecting our *a priori* knowledge of the noise introduced by the imaging process. In our experiments, ϵ was chosen to be two pixels.

After smoothing, points of sharp convexities and deep concavities were extracted to serve as interest points. We followed the method described in [11] for detecting deep concavities and used it to find sharp convexities as well. The technique can be described as follows:

a) Discretize the (smoothed) curve into a sequence of evenly spaced points $(c_j)_{j=1}^{n}$.
b) For each j, compute an estimate of the boundary tangent at (c_j) by calculating the line of best (least-squares) fit to a prescribed number of k successive boundary points starting at j.


(a)


(b)


(c)

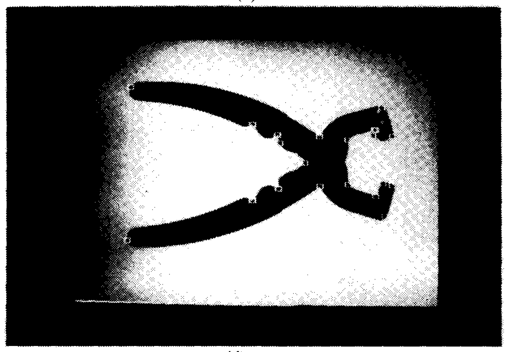
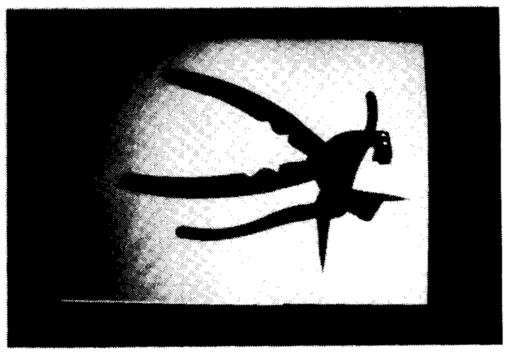
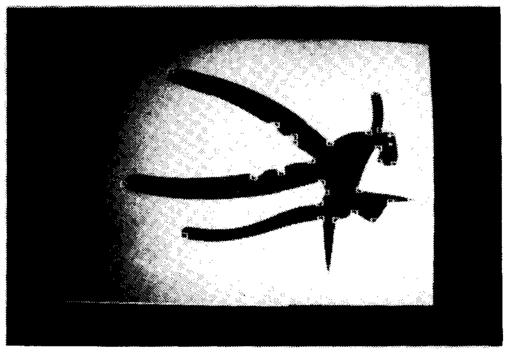

(d)

Fig. 3. Models of the two pliers and their extracted interest points: (a), (b) Models of pliers; (c), (d) extracted interest points.

c) Calculate the second derivative by differencing successive tangents. Look for the maxima and minima of the second derivatives. The maxima points correspond to deep concavities, whereas the minima points correspond to sharp convexities.


(a)


(b)

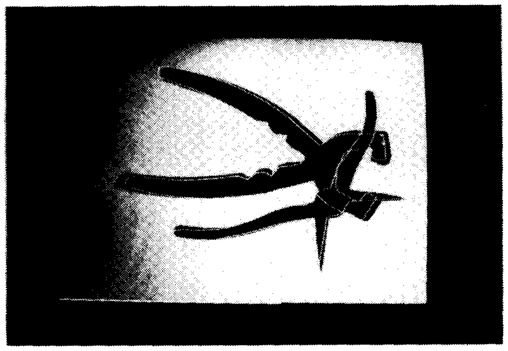
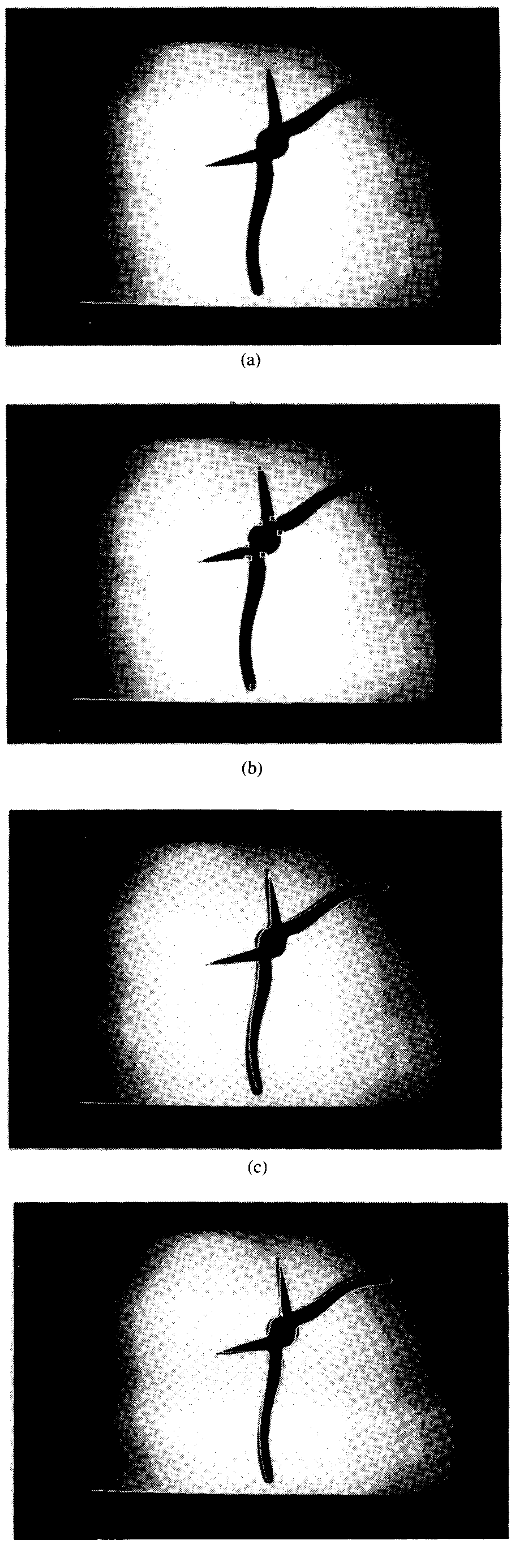

(c)

Fig. 4. (a) Composite scene of the pliers of Fig. 3 (observe different lengths of handles due to the tilt); (b) extracted interest points; (c) recognition of the models in the scene.

In our experiments, we used a 2-pixel distance between points c_j and c_{j+1}, $(j = 1, \cdots, n-1)$ in step (a), and $k = 7$ points for the tangent fit in step (b).

Quantized coordinates (in the appropriate bases) of the extracted interest points were stored in the model hash table. The quantization of the hash table was done following a simple noise model. The basic assumption was that if the coordinate value was bigger, the more (absolute) variation was introduced into this value as a consequence of image noise. This model is rather simplistic, as opposed to the worst-case noise model, which we discuss in Section V and expand on in [28]. However, at that stage of the experiments, we were not concerned with obtaining optimal results.

We used a hash table with variable size bins. The hash-table bin size becomes bigger as a function of the coordinate

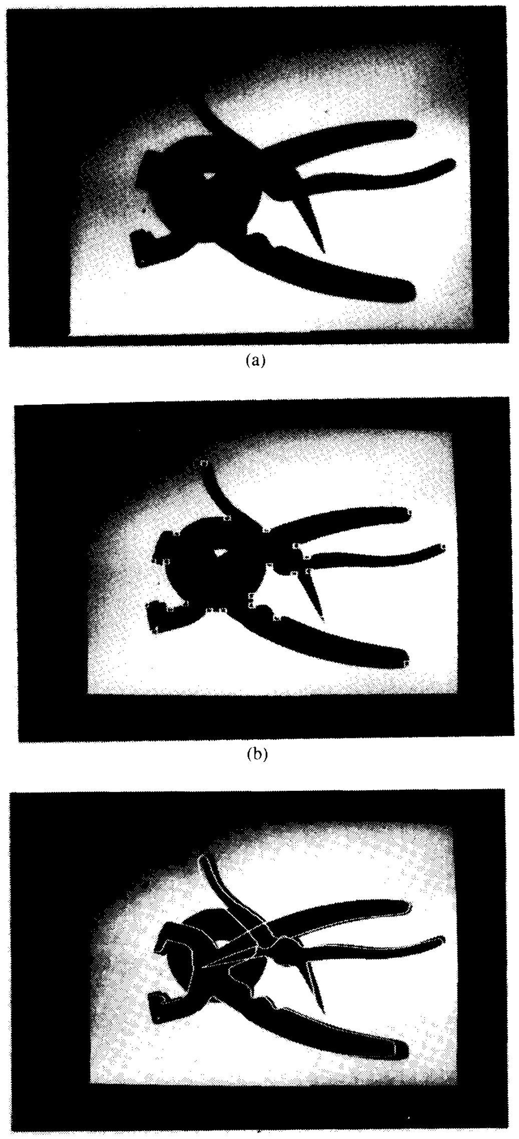

(a)

(b)

(c)

(d)

Fig. 5. (a) Pliers rotated and tilted in space (see different length of handles); (b) extracted interest points; (c) matching based on one basis triplet; (d) best least squares affine correspondence.


(a)

(b)

(c)

Fig. 6. (a) Composite scene of the pliers of Fig. 3 with an additional occluding object; (b) extracted interest points; (c) recognition of the models.

value. The amount of growth is linear in each coordinate value. Each bin has two parameters: the x and y coordinates of a point. A point (x, y) belongs to bin (i, j), if $X(i-1) \leq x < X(i)$ and $Y(j-1) \leq y < Y(j)$. The range of each parameter is determined independently. We initiate by setting

$$X(0) = Y(0) = 0;\; X(1) = Y(1) = L_0;\; X(-1) = Y(-1) = -L_0.$$

The bin size is then defined recursively as follows. For positive $i = 1, 2, \cdots$

$$X(i+1) - X(i) = Y(i+1) - Y(i) = i\delta$$

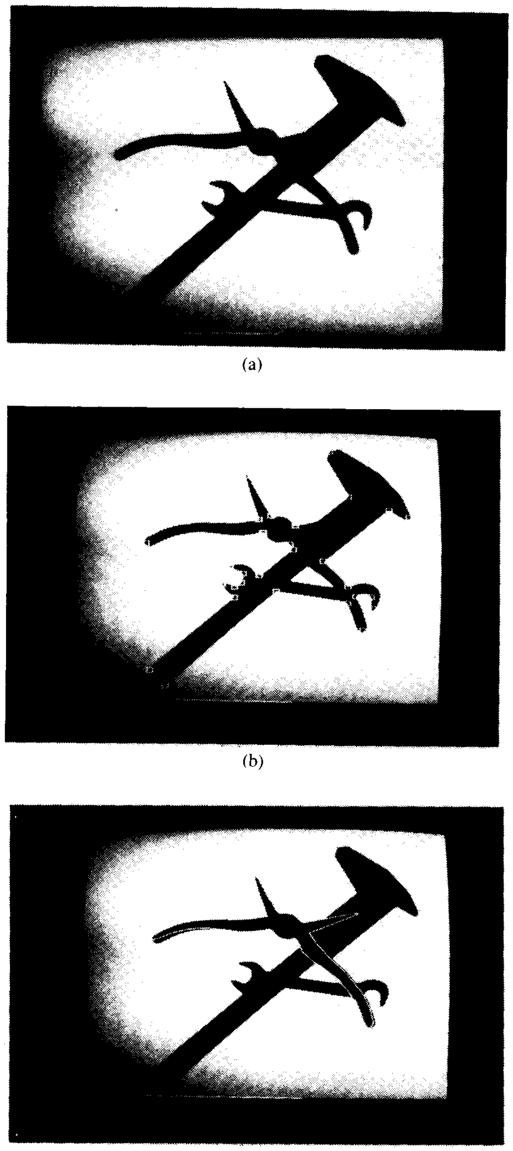

(a) (b) (c)

Fig. 7. (a) Composite scene of one of the pliers of Fig. 3 with an additional occluding object (observe the change in scale); (b) extracted interest points; (c) recognition of the model.

and for negative $i = -1, -2, \cdots$

$$X(i-1) - X(i) = Y(i-1) - Y(i) = i\delta$$

where L_0 and δ are fixed parameters, regulating the size of the bins. In our experiments, we chose $L_0 = 0.1$, and $\delta = 0.01$. The coordinates were limited to the range of $(-10, +10)$ along each axis.

In the recognition, phase voting was done by computing the appropriate bin containing the given image coordinates and advancing the accumulators of the bases stored in that bin. There was no attempt to accumulate votes from neighboring bins. Because we did not take the worst-case approach, part of the correct votes were missed. Nevertheless, the performance of the system was still acceptable.

Fig. 3(a) and (b) are the original images of two models (pliers), and Fig. 3(c) and (d) show the extracted interest points of the models. In Fig. 5(a), we see an image of the pliers of Fig. 3(a) rotated, translated, and tilted at about 40° (observe the different lengths of both handles in the image). The recognition algorithm was performed to obtain a number of matching basis triplets. The corresponding affine transformations were calculated, and for each such transformation, the transformed model was superimposed on the scene of Fig. 5(a). Fig. 5(c) shows such a transformation computed according to a basis triplet that gives a somewhat noisy match. This solution is significantly improved by the best least-squares match, which is given in Fig. 5(d), and was calculated using all the points recognized as model points by the basis triplet of Fig. 5(c) (see Section VI).

In Fig. 4, we see an image of a composite overlapping scene of both pliers, (which was also significantly tilted), its extracted interest points, and the recognition results. Note that in Fig. 4(b), we have additional interest points that are created by the superposition of the two objects. These points do not correspond to the interest points of the original models. In addition, one can see that a number of the original interest points are occluded in the scene.

To give an intuitive feeling of our algorithm's performance, we include some statistics on the example of Fig. 4. The total number of interest points in the scene of Fig. 4(b) is 28. Sixteen of them are unoccluded model points of the second plier out of 21 original model points (see Fig. 3(d)). To get the statistics, we run our recognition algorithm on all the possible basis triplets of Fig. 4(b). For each triplet, we found the set of best (maximum vote) matching model triplets. The number of points identified by such a triplet as model points are the so-called 'no. of votes' in the first column of Table I. The second column gives the number of triplets that obtained these votes, and the third column gives the number of triplets that were verified as belonging to the model (correct triplets).

The results are summarized in Table I.

Remarks:

a) Since we have 16 model points in the scene, we expect a maximum of 13 votes for a correct triplet.
b) Since all six ordered occurrences of the same unordered triplet will give the same voting result, we counted unordered triplets in our statistics. In the algorithm, we are dealing with ordered triplets; thus, for example, we have $4 \times 6 = 24$ ordered basis triplets with the maximal number of votes.
c) Since we did not use a worst-case noise model, many correct votes were missed. When using a worst-case noise model, most correct votes should be around the table row representing the maximum votes for a correct triplet (in this case 13).

Figs. 6 and 7 give additional recognition examples. These examples include additional occluding objects that do not belong to the model data base. Except for the tilt, there was

TABLE I
Performance of the Algorithm on the Example of Fig. 4

no. of votes	no. of basis triplets	correct basis triplets
14+	0	0
13	4	4
12	11	10
11	12	8
10	29	12
9	56	22
8	145	38
7	287	62
6	805	151

also a significant change in the distance of the camera from the scene, which is also in the example of Fig. 7.

At this stage of our experiments, no effort was made to optimize various parameters of the algorithm.

XII. Conclusions and Future Research

This paper represents our preliminary ideas and experiments in 3-D object recognition from 2-D images. Here, we based our methods on the representation of objects by point sets and matching the corresponding sets of points. By applying geometric constraints, these sets of points can be further invariantly represented by their coordinates in a small subset of points (basis points). The size of the basis subset depends on the transformation applied to the models of the image scene. We have demonstrated how a basis of two points is sufficient for 2-D scenes, where we allow rotation, translation, and scale transformations, and a basis of three points suffices under the assumption of the affine approximation for the perspective view. An important characteristic of our method is the division of the matching process into preprocessing and recognition stages. This significantly reduces the complexitgy of our algorithm compared with the straightforward approach and enables execution of the preprocessing stage off line. The algorithms presented are inherently parallel.

Our algorithm has been successfully implemented and tested on real life data of industrial parts.

The methods described in this paper naturally extend to additional cases. In particular, 3-D object recognition from range data can be accomplished by similar methods using three point bases. The above-mentioned methods also extend to the following topics, where active experimentation is currently performed:

1) Recognition of nonflat 3-D objects from 2-D images (see [2])
2) implementation of similar matching procedures based on synthesis of point and line information
3) affine invariant curve matching (see [4])
4) recognition of objects using parametrized models.

References

[1] R. T. Chin and C. R. Dyer, "Model-based recognition in robot vision," *ACM Comput. Surveys*, vol. 18, pp. 67-108, 1986.

[2] Y. Lamdan and H. J. Wolfson, "Geometric hashing: A general and efficient model-based recognition scheme," in *Proc. Second Int. Conf. Comput. Vision* (Tampa, FL), Dec. 1988, pp. 238-249.

[3] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, "On recognition of 3-D objects from 2-D images," in *Proc. IEEE Int. Conf. Robotics Automat.* (Philadelphia, PA), April 1988, pp. 1407-1413; Robotics Rep. No. 122, New York University, Sept. 1987.

[4] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, "Object recognition by affine invariant matching," in *Proc. IEEE Conf. Comput. Vision Patt. Recog.* (Ann Arbor, MI), June 1988, pp. 335-344; Robotics Rep. No. 136, New York University, Nov. 1987.

[5] N. Ayache and O. D. Faugeras, "HYPER: A new approach for the recognition and positioning of two-dimensional objects," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. PAMI-8, pp. 44-54, Jan. 1986.

[6] R. C. Bolles and R. A. Cain, "Recognizing and locating partially visible objects: The local feature focus method," *Int. J. Robotics Res.*, vol. 1, pp. 57-82, 1982.

[7] R. C. Bolles and P. Horaud, "3DPO: A three-dimensional part orientation system," *Int. J. Robotics Res.*, vol. 5, pp. 3-26, 1986.

[8] W. E. L. Grimson and T. Lozano-Perez, "Model based recognition and localization from sparse range or tactile data," *Int. J. Robotics Res.*, vol. 3, pp. 3-35, 1984.

[9] W. E. L. Grimson and T. Lozano-Perez, "Localizing overlapping parts by searching the interpretion tree," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. PAMI-9, pp. 469-482, 1987.

[10] J. L. Turney, T. N. Mudge, and R. A. Volz, "Recognizing partially occluded parts," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. PAMI-7, pp. 410-421, July 1985.

[11] A. Kalvin, E. Schonberg, J. T. Schwartz, and M. Sharir, "Two dimensional model based boundary matching using footprints," *Int. J. Robotics Res.*, vol. 5, pp. 38-55, 1986.

[12] J. Hong and H. J. Wolfson, "An improved model-based matching method using footprints," in *Proc. 9th Int. Conf. Patt. Recog.* (Rome, Italy), Nov. 1988; Robotics Rep. 137, New York University, Nov. 1987.

[13] R. Cyganski and J. A. Orr, "Applications of tensor theory to object recognition and orientation determination," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. PAMI-7, pp. 662-673, Nov. 1985.

[14] R. Cyganski, J. Orr, T. Cott, and R. Dodson, "Development, implementation, testing, and application of an affine transform invariant curvature function," in *Proc. 1st Int. Conf. Comput. Vision* (London), 1987, pp. 496-500.

[15] J. Hong and X. Tan, "The similarity between shapes under affine transformation," Robotics Res. Rep. 133, New York University, Dec., 1987.

[16] B. Bamieh and R. J. P. De Figueiredo, "A general moment-invariants/attributed-graph method for 3-D object recognition from a single image," *IEEE J. Robotics Automat.*, vol. RA-2, pp. 31-41, Mar. 1986.

[17] P. J. Besl and R. C. Jain, "Three-dimensional object recognition," *ACM Comput. Surveys*, vol. 17, pp. 75-154, 1985.

[18] D. G. Lowe, *Perceptual Organization and Visual Recognition*. Boston: Kluwer Academic, 1985.

[19] D. W. Thompson and J. L. Mundy, "Three-dimensional model matching from an unconstrained viewpoint," in *Proc. IEEE Int. Conf. Robotics Automat.* (Raleigh, NC), 1987, pp. 208-220.

[20] D. P. Huttenlocher and S. Ullman, "Object recognition using alignment," in *Proc. 1'st Int. Conf. Comput. Vision* (London), 1987, pp. 102-111.

[21] F. Klein, *Elementary Mathematics from an Advanced Standpoint: Geometry* (3rd. ed.). New York: Macmillan, 1925 (English translation).

[22] B. K. P. Horn, *Robot Vision*. Cambridge, MA: MIT Press, 1986.

[23] H. Moravec, "Towards automatic visual obstacle avoidance," in *Proc. 5th IJCAI*, 1977, p. 584.

[24] S. Barnard and W. Thompson, "Disparity analysis of images," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. PAMI-1, pp. 333-340, 1980.

[25] W. E. L. Grimson, "The combinatorics of local constraints in model-based recognition and localization from sparse data," *J. ACM*, vol. 33, pp. 658-686, 1986.

[26] G. Stockman, "Object recognition and localization via pose clustering," *Comput. Vision Graphics Image Processing*, vol. 40, pp.

361-387, 1987.

[27] I. M. Yaglom and V. G. Ashkinuze, *Ideas and Methods of Affine and Projective Geometry, Part 1.* Moscow, 1962 (in Russian).

[28] Y. Lamdan and H. J. Wolfson, "*On the Error Analysis of 'Geometric Hashing*,'" Robotics Rep. 213, New York University, New York, NY, 1989.

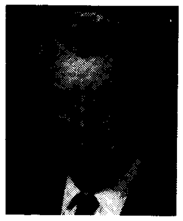
[29] I. B. Kuperman, *Approximate Linear Algebraic Equations.* New York: Van Nostrand, 1971.

[30] G. H. Golub and C. F. van Loan, *Matrix Computations.* Baltimore, MD: J. Hopkins Press, 1983.

[31] J. T. Schwartz and M. Sharir, "Identification of partially obscured objects in two dimensions by matching of noisy 'characteristic curves,'" *Int. J. Robotics Res.*, vol. 6, pp. 29-44, 1987.

Yehezkel Lamdan (S'86) was born in Israel in 1960. He received the B.Sc. in mathematics and computer science from Tel-Aviv University in 1983 and the M.A. degree in computer science from City University of New York in 1984. He is currently completing the Ph.D. degree in computer science at the Courant Institute of Mathematical Sciences, New York University.

He has performed research and development in the areas of biomedical imaging and expert systems for automatic fault diagnosis in large communications networks. His current research interests include computer vision, artificial intelligence, and data communications.



Jacob T. Schwartz (M'87) received the B.Sc degree in 1948 from the City College of New York and the Ph.D degree from Yale University, New Haven, CT.

From 1953-1956, he was an Assistant Professor of Mathematics at Yale. In 1956, he moved to the Courant Institute, New York University, where he has subsequently remained except for several short periods of academic and government-service leave. He is the author of various works in mathematics, mathematical economics, computer science, and robotics.

His current research interests include computer vision, robotics, and computational issues in neuroscience.

Haim J. Wolfson (M'87) received the B.Sc, M.Sc., and Ph.D degrees in mathematics from Tel Aviv University, Israel.

From 1985 to 1989, he was an Associate Research Scientist and a Visiting Assistant Professor at the Robotics Research Laboratory of the Courant Institute of Mathematics, New York University. Since 1989, he has been with the Department of Computer Science, School of Mathematical Sciences, Tel Aviv University, Israel. His current research interests include computer vision, pattern recognition, artificial intelligence, and robotics.