Shapeme Histogram Projection and Matching for Partial Object Recognition

Ying Shan, *Senior Member*, *IEEE*, Harpreet S. Sawhney, *Member*, *IEEE*, Bogdan Matei, *Member*, *IEEE*, and Rakesh Kumar, *Member*, *IEEE*

Abstract—Histograms of shape signature or prototypical shapes, called shapemes, have been used effectively in previous work for 2D/3D shape matching and recognition. We extend the idea of shapeme histogram to recognize partially observed query objects from a database of complete model objects. We propose representing each model object as a collection of shapeme histograms and match the query histogram to this representation in two steps: 1) compute a constrained projection of the query histogram onto the subspace spanned by all the shapeme histograms of the model and 2) compute a match measure between the query histogram and the projection. The first step is formulated as a constrained optimization problem that is solved by a sampling algorithm. The second step is formulated under a Bayesian framework, where an implicit feature selection process is conducted to improve the discrimination capability of shapeme histograms. Results of matching partially viewed range objects with a 243 model database demonstrate better performance than the original shapeme histogram matching algorithm and other approaches.

Index Terms—Shapeme histogram, spin image, Gibbs sampling, feature saliency, object recognition, Bayesian analysis.

1 Introduction

New developments in range sensors make it possible to quickly acquire 3D range information with relatively low cost. This has shed new light on the problem of 3D recognition with range images. As compared with video sensors, range sensors can provide 3D range information without solving the difficult 3D reconstruction problem. three-dimensional data greatly simplifies the grouping and segmentation problems, which are much more difficult for 2D video data. However, the point clouds for a model object cover the complete object, whereas the scene range images provide only partial views of a query scene object, as shown in Fig. 1. This situation mimics range sensors that capture an object using view-based 3D data. Scenes are created using one or a small number of range images (3D views) of an object.

During the last decade, invariant semilocal features, such as splash feature [1] and spin image [2], have been used for robust 3D object recognition. They are more flexible than global structural features such as global spherical harmonic feature [3] because they are still valid as long as the information inside the effective neighborhood is not changed. They are also more discriminating than local features such as shape index [4] because they tend to encode more information around larger neighborhoods. Since the dimensionality of semilocal feature is usually very high, brute-force matching is prohibited for database sizes of practical interest. A promising way of solving this problem is to use the powerful concept of "shapeme histogram," proposed in [5] to solve 2D recognition problems. Shapemes are defined as clusters of shape signatures that correspond to different parts on the objects. Histograms of shapemes characterizing shape distributions is a compact, albeit rich descriptor ideal for rapid and accurate shape matching. Readers can refer to the next section for a detailed description of the shapeme histogram. Normally, matching shapeme histograms of two objects requires both objects to be complete. It would not work properly if the query object is only a part of the model object, e.g., the query is a range image that covers only a limited portion of an object. To address this problem, we propose dividing a model into smaller parts and computing shapeme histograms for each of them. Matching a query histogram with a model involves two steps: 1) Find the parts on the model object that correspond to the query object and 2) match the shapeme histogram computed from those parts to the histogram of the query object. The first step is formulated as a constrained optimization problem where the goal is to find the optimal projection of the query histogram onto the subspace spanned by all the shapeme histograms of the model. We then propose a sampling-based algorithm to solve this optimization problem efficiently. The second step is expressed into a Bayesian framework, where an implicit feature selection process is conducted to improve the discrimination capability of shapeme histograms.

The proposed method provides a general framework that can be used for matching partial objects of any kind with shapeme histograms. In this paper, we are interested in the application of finding 3D models in a database that are similar to an object presented in a range image. The 3D model objects and the query range images are both represented as 3D point clouds. The point clouds for a model object cover the complete object, whereas the query range images provide only partial views of a query object. The goal is to find a short list of model objects that are the closest matches to a query object. The actual model object will be picked from the short list with more precise but time consuming verification algorithms such as the RANSAC and saliency-based approaches proposed in [6] and [7]. This paper is an extended version of [8].

• *The authors are with the Vision Technologies Laboratory, Sarnoff Corporation, 201 Washington Road, Princeton, NJ 08540. E-mail: {yshan, hsawhney, bmatei, rkumar}@sarnoff.com.*

Manuscript received 8 June 2004; revised 4 Aug. 2004; accepted 15 Aug. 2005; published online 14 Feb. 2006.
Recommended for acceptance by R. Chellappa.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0291-0604.

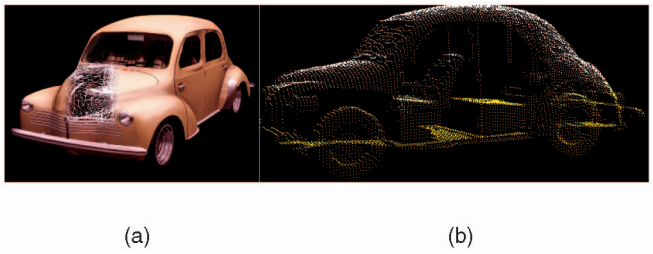


Fig. 1. Model points cloud can be generated from complete 3D facet models as in (a) while query point clouds obtained from range sensor (b) can cover only a part of the object.

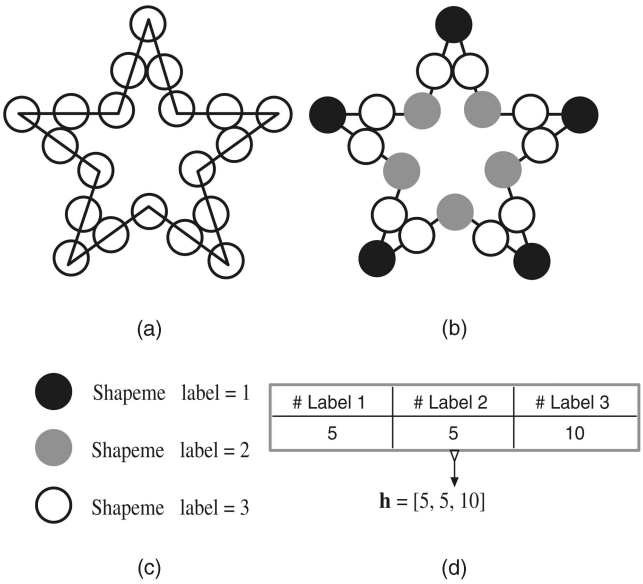


Fig. 2. Shapeme histogram. (a) A star-shaped object and the 20 basis points (the circles) where 2D invariant features are computed. (b) The same object with each feature point labeled with a set of three shapemes as in (c). (d) The shapeme histogram of the object in both table and vector formats. Since there are 5 #1, 5 #2, and 10 #3 labels on the object, respectively, the shapeme histogram of this object is $\mathbf{h} = [5, 5, 10]$.

2 Related Work

Our basic representation is based on the idea of shapeme and shapeme histogram, proposed by Mori et al. [5] for 2D object recognition. A shapeme is a prototype shape feature that represents a cluster of similar invariant features. These features are computed at basis points that are densely sampled on the 2D object and, hence, a single object may contain many features. Each feature on an object is assigned to its closest shapeme. Counting the frequency of shapemes over the whole object gives the shapeme histogram of the object. Fig. 2 shows an example. Shapeme histogram is used by [9] and [10] for texture recognition. In this paper, we extend shapeme histogram-based approach to handle the matching of partially observed objects. Moreover, we also propose to select shapemes that are critical to a given task and embed this process into a Bayesian matching framework.

The shapeme histogram for 3D objects can be constructed likewise in terms of 3D shape features and the associated 3D shapemes. Invariant, semilocal shape features such as splash feature [1], spherical harmonic [3], and spin image [2] have been developed in the past few years for matching and recognizing 3D objects. Among these features, we selected spin image according to previous work [11], [12] as well as our

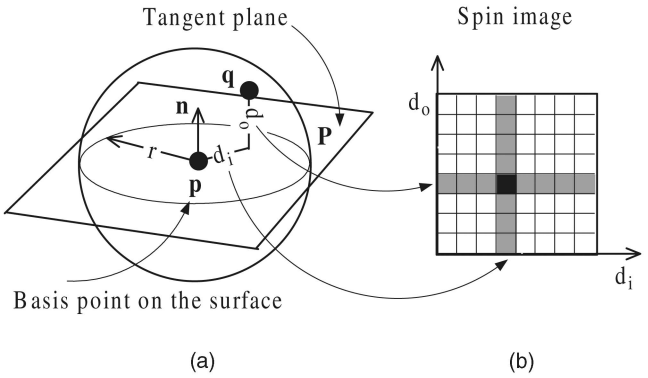


Fig. 3. Construction of invariant semilocal spin image. (a) Spin image is computed at each basis point $\mathbf{p}$ inside a spherical neighborhood with radius r. It is a two-dimensional histogram of the in-plane distances d_i and the out-of-plane distances d_o of other 3D surface points (inside the neighborhood) defined with respect to the tangent plane $\mathbf{P}$ at the basis point. (b) Shows d_i and d_o of a point $\mathbf{q}$, and the corresponding pixel (marked as black) on the spin image the point votes for.

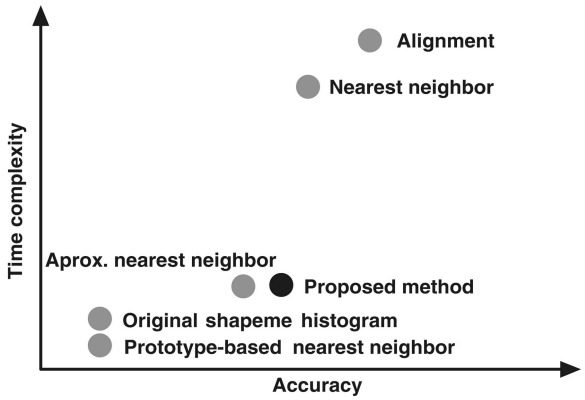


Fig. 4. The proposed method with respect to other approaches for partial object recognition. The alignment-based method is the most accurate but expensive approach. Our method is fast and accurate and can ideally be used as a model pruning front-end for the alignment-based approach. See text for more details.

own experiments, which indicated that spin image is a reasonable choice for pose invariant semilocal shape matching in the presence of noise and scene clutter. It should be noted that, though selecting an optimal invariant feature is always important, it is not critical to the points that we want to make in this paper. Fig. 3 briefly illustrates the construction of the spin image. More details about the spin image representation can be found in [2].

Fig. 4 is the picture showing the relationships of the original shapeme histogram method and our proposed method with respect to other recognition approaches. Obviously, the most accurate method is to align the objects being matched [1], [2], [13] because it takes into account the global geometric constraints. The alignment-based approach is expensive, and has difficulty when aligning nonrigid objects. The nearest neighbor approach without global geometric constraints can be quite accurate when using semilocal features such as spin images. The problem is that it is too slow, especially when the feature dimension is high and the database is large. The approximate nearest neighbor approach [14] and the prototype-base nearest neighbor approach [15], [16] are employed to address this problem. However, the performance of the former depends

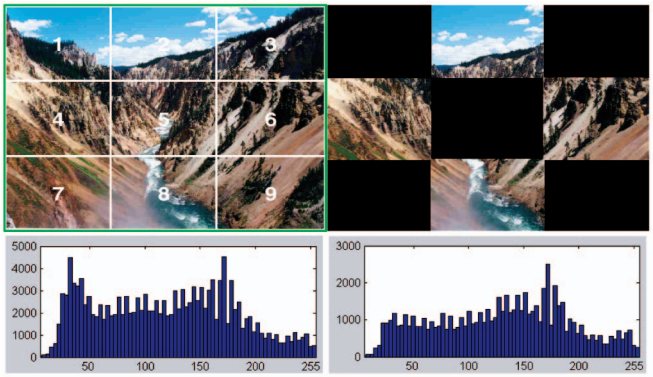


Fig. 5. An example of using histogram projection for 2D image matching. The left column shows the original image and its intensity histogram. The image is segmented into nine pieces and each of them is labeled with a number. The right column shows an image consisting of only four pieces of the original image. The corresponding intensity histogram is computed without including the black areas.

on the type of feature and the distribution of the features in the database, while the latter is, in general, not accurate. Shapeme histogram is a fast method because both the model and the query are represented with single vectors. However, the original shapeme histogram approach does not handle a partial query. This justifies the use of our proposed approach, which is fast and accurate even when the query object is partially observed. An ideal application of our method is to use it as a model pruning front-end where the goal is to find a short list of model objects that can be verified by more sophisticated and accurate approaches.

3 Histogram Projection

We now introduce the histogram projection idea to address the problem of matching the histogram of a partial query against a complete model. As an illustration example, consider an image matching problem. Fig. 5 shows an original image (left column) and another image (right column) consisting of only four segments from the original image. Obviously, the intensity histogram computed from the image in the left column looks different from the image in the right column. Given the intensity histogram $\mathbf{h}_i^s$ for each segment i of the image in the left column and the histogram $\mathbf{h}^q$ of the image in the right column, the problem is to find the set $\mathcal{S}$ of all the segments such that $\mathbf{h}^q = \sum_{k \in \mathcal{S}} \mathbf{h}_k^s$.

To solve the problem, let $\mathbf{A} = [\mathbf{h}_1^s, \mathbf{h}_2^s, \ldots, \mathbf{h}_9^s]$ be the $l \times 9$ matrix, where l is the number of bins of each histogram and $\mathbf{x} \in \mathcal{R}^9$ is an unknown vector. Suppose that $rank(\mathbf{A}) = 9$, solving the linear system $\mathbf{A}^T\mathbf{A}\mathbf{x} - \mathbf{A}^T\mathbf{h}^q = 0$ gives us the solution $\mathbf{x} = [0, 1, 0, 1, 0, 1, 0, 1, 0]^T$. We now conclude that $\mathcal{S} = \{2, 4, 6, 8\}$, i.e., the image in the right column consists of these numbered segments from the image in the left column.

In summary, we have projected $\mathbf{h}^q$ onto the subspace spanned by $\mathbf{h}_i^s$, and the projection is $\mathbf{A}\mathbf{x}$, which, in this case, is exactly the same as $\mathbf{h}^q$. Note that this is just an ideal example where the query tiles are the same as the tiles used to construct the histogram subspace.

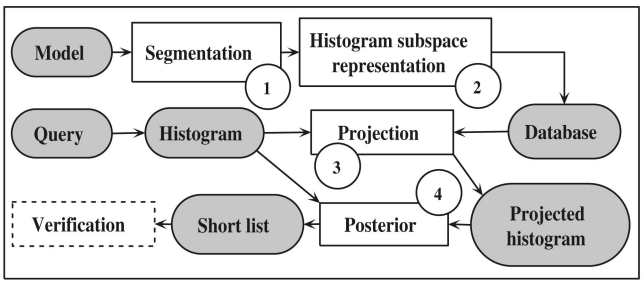


Fig. 6. Shapeme histogram projection for partial 3D object recognition. Model objects are segmented in Block 1 and component-based histogram are computed to form a subspace representation in Block 2. The component histograms are added into a histogram database. Query histogram is computed and projected in Block 3 onto the subspace spanned by the component model histograms in the database. Posteriors are computed in Block 4 based on the query histogram and its projected histogram, as described later in Section 5 (14). A short list of matching model objects is generated for optional verification.

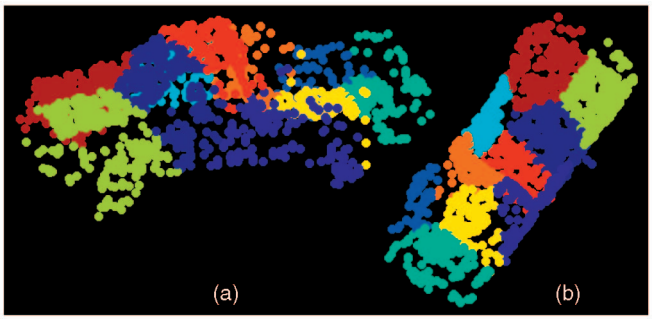


Fig. 7. Spatial object segmentation of a pickup truck (best viewed with color). The circular points represent the 3D basis points where the spin images are computed. Basis points from different spatial segments are labeled with different color. (a) A side view of the segmented object. (b) The top view.

4 Shapeme Histogram Projection for Partial 3D Object Recognition

This section elaborates on the idea in the previous section and proposes a shapeme histogram projection approach for matching partially viewed 3D objects. A schematic description of the approach is given in Fig. 6. Blocks 1, 2, and 3 are covered in this section and Block 4 is covered in the next section. In the following discussion, model objects are assumed to be complete.

4.1 Spatial Object Segmentation

In Block 1, a set of spin image $\mathcal{B}_i$ is computed for object M_i at basis points $\mathcal{P}_i$. The basis points are densely sampled from the surface of the object (for details, see Appendix C). Each spin image $\mathbf{b}_{i,j} \in \mathcal{B}_i$ is associated with a 3D basis point $\mathbf{p}_{i,j} \in \mathcal{P}_i$. The spatial segmentation algorithm performs k-means clustering in the basis point set $\mathcal{P}_i$ and returns n clusters of 3D points, where n is a predetermined number. The set of spin images $\mathcal{B}_i$ is split into n groups accordingly. Note that this process should not be confused with clustering in the spin image space, where the goal is to find the prototype features, the shapemes. Fig. 7 shows two views of a segmented pickup truck.

4.2 Histogram Subspace Representation

In Block 2, once the spin images of object M_i have been separated into n groups, a shapeme histogram $\mathbf{h}_i^s$ can then

be computed for each group. The histogram subspace representation is then the $m \times n$ matrix $\mathbf{A}$, as mentioned in the previous section, where $\mathbf{A} = [\mathbf{h}_1^s, \mathbf{h}_2^s, \cdots, \mathbf{h}_n^s]$ and m is the number of shapemes. This matrix is added into the model database. Obviously, as compared with the approaches that put all the raw features inside the database, the saving in storage space with the shapeme histogram representation is huge.

4.3 Histogram Projection

In Block 3, given a query shapeme histogram $\mathbf{h}^q$, its projection (the closest approximation) in the subspace spanned by the ith model histograms can be computed as $\hat{\mathbf{h}}^q = \mathbf{A}_i\mathbf{x}$, where $\mathbf{A}_i$ is the $\mathbf{A}$ matrix of the ith model, and $\mathbf{x}$ can be computed as

$$\text{minimize} \|\mathbf{A}_i\mathbf{x} - \mathbf{h}^q\|_2 \quad \text{subject to } \mathbf{x} \in \{0,1\}^n. \tag{1}$$

The solution of the unconstrained version of (1) is the solution of the linear system $\mathbf{A}_i^T\mathbf{A}_i\mathbf{x} - \mathbf{A}_i^T\mathbf{h}^q = 0$, if $\mathbf{A}_i$ has full column rank. This is what we used to solve the ideal image matching problem in Fig. 5. In the case when the query histogram is noisy or the query range image cuts through significant portions of some segments, the constraint $\mathbf{x} \in \{0,1\}^n$ can be used to "regularize" the solution. Solving (1) with exhaustive search requires 2^n operations, which is tractable only when n is small. When n is large, a Gibbs sampler [17] is employed to explore the binary solution space more efficiently. The Gibbs distribution that corresponds to the objective function in (1) is

$$G(x) = \frac{1}{Z}\exp[-(\|\mathbf{A}_i\mathbf{x} - \mathbf{h}^q\|_2)/T], \tag{2}$$

where Z is an unknown normalization factor and T is the temperature constant. Because x is binary, the local conditional probability (or the local characteristic function) can be derived easily from (2) as

$$\begin{aligned} &G(x_j = 0 \,|\, \{x_k | k \neq j\}) \\ &= \frac{G(x_j = 0, \{x_k\})}{G(\{x_k\})} \\ &= \frac{G(x_j = 0, \{x_k\})}{\sum_{x_j \in \{0,1\}} G(x_j, \{x_k\})} \\ &= \frac{1}{1 + G(x_j = 0, \{x_k\})/G(x_j = 1, \{x_k\})}, \end{aligned} \tag{3}$$

where x_k is the kth coordinate of $\mathbf{x}$. Note that the unknown factor Z is canceled out in (3). Given a random initial guess, the sampler sweeps through each coordinate x_k sequentially, and flips its value according to the local conditional probability in (3). The computational cost in each step is negligible since only one coordinate is touched and the actual cost $\mathbf{A}_i\mathbf{x} - \mathbf{h}^q$ can be computed incrementally. The same process is repeated for several iterations and the $\mathbf{x}$ that has the smallest $G(\mathbf{x})$ is selected as the solution. Because our objective function is simple and the dimension is relatively small ($n < 100$), Gibbs sampler can quickly converge to a solution very close to the global optimum. In fact, we find that, when $n = 10$, Gibbs sampler gives the identical result as the exhaustive search.

The histogram projection method proposed in this section is in some degree related to the NMF [18] method. One major difference is that we use the binary constraint instead of the nonnegative constraint used in NMF. For the class of problems discussed in this paper, the binary constraint is more accurate. The Gibbs Sampling algorithm used to impose these constraints is different from NMF. Also, the saliency-based weighting scheme described under a Bayesian framework is not reflected in the NMF approach.

5 A Bayesian Framework for Shapeme Histogram Matching

This section provides a Bayesian framework for the shapeme histogram matching. It explains why, under some mild assumptions, matching shapeme histogram is equivalent to computing the posterior of a model given the query. More importantly, it reveals that an implicit feature selection process is naturally embedded when matching under the proposed framework. To simplify the discussion, we will lay out the framework based on the assumption that both the query object and the model object are complete and then apply it to a partial query using the histogram projection approach proposed in the previous section.

5.1 Notations

Let M_i be the ith model object and Q the query object. The problem is to find a set $\mathcal{C}$ of candidate model objects with the largest posteriors, $P(M_i \,|\, Q)$, such that $\sum_{i \in \mathcal{C}} P(M_i \,|\, Q) \geq \epsilon$, where $0 \leq \epsilon \leq 1$ is a predefined value close to 1. The framework is designed to pick the top few candidate models out of all the models in the database and, hence, we need to find a set of posterior probabilities whose sum is greater than a small threshold. In other words, the proposed method serves as a front end process for pruning a large set of models and lets other slower but more accurate methods resolve the remaining ambiguity. By doing this, we gain sublinear speed since the second matching algorithm (usually much slower) only needs to work on the pruned set of models, instead of the whole database. As an illustrative example to justify this, suppose that we have a model database with 100 models and we have a simple pruning method runs one second per query and a slower but more accurate method that runs at 10 seconds per query. The pruning method picks say the top 10 candidate models and the slower methods uses 100 seconds to resolve the ambiguity among the 10 models. Altogether we spend 110 seconds to find the correct model, while we have to spend 1,000 seconds without the pruning step.

For each model object M_i, a set of spin images, $\mathcal{B}_i = \{\mathbf{b}_{i,1}, \mathbf{b}_{i,2}, \cdots, \mathbf{b}_{i,t_i}\}$, is computed at a number of basis points, $\mathcal{P}_i = \{\mathbf{p}_{i,1}, \mathbf{p}_{i,2}, \mathbf{p}_{i,t_i}\}$, densely sampled along the surface of the object, where $\mathbf{b}_{i,\cdot} \in \mathcal{R}^d$, $\mathbf{p}_{i,\cdot} \in \mathcal{R}^3$, d is the spin image dimension, and t_i is the number of spin images in the object. For all the model objects under consideration, a set of shapemes, $\mathcal{A} = \{\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_m\}$, is computed from the full set of spin images $\Omega = \bigcup_{j=1}^{o} \mathcal{B}_j$ by clustering in the spin image space, where m and o are the number of shapemes and model objects, respectively. For each shapeme $\mathbf{a}_k$, a median radius r_k is computed during the clustering process. Median radius is the median distance from the center of the shapeme (cluster) to the spin images that belong to this shapeme (cluster).

Each model object M_i is then represented by a shapeme histogram $\mathbf{h}_i = [u_{i,1}, u_{i,2}, \cdots, u_{i,m}]^T$, where the number of bins m is the same as the number of shapemes in $\mathcal{A}$ and $u_{i,k}$ is the number of spin images that have been labeled with the kth shapeme. The probability of each shapeme $\mathbf{a}_k$ given a model object M_i is then given by

$$P(\mathbf{a}_k \mid M_i) = u_{i,k}/u_i, \tag{4}$$

where $u_i = \sum_k u_{i,k}$ is the total number of labeled spin images in the object. The query is represented by the set of all spin images computed from the object.

5.2 Model Posterior Given a Query

The posterior probability of a model given a query can be computed as

$$\begin{aligned} &P(M_i \mid Q) \\ &= \sum_{\mathbf{a}_k} P(\mathbf{a}_k \mid Q)\ P(M_i \mid \mathbf{a}_k, Q) &\text{(5a)}\\ &\approx \sum_{\mathbf{a}_k} P(\mathbf{a}_k \mid Q)\ P(M_i \mid \mathbf{a}_k), &\text{(5b)}\end{aligned}$$

where (5b) is an approximation of (5a) assuming that $P(M_i \mid \mathbf{a}_k, Q) \approx P(M_i \mid \mathbf{a}_k)$, that is, when both a shapeme and the query are given, the probability of M_i is determined only by the shapeme. The same assumption is used in [15] and is coined as the "homogeneity assumption." Applying Bayes' rule to (5b) leads to

$$P(M_i \mid Q) \approx \sum_{\mathbf{a}_k} P(\mathbf{a}_k \mid Q)\ \frac{P(\mathbf{a}_k \mid M_i)\ P(M_i)}{\sum_{M_l} P(\mathbf{a}_k \mid M_l)\ P(M_l)}. \tag{6}$$

Assuming that all models are equally likely, we have

$$P(M_i \mid Q) \approx \sum_{\mathbf{a}_k} P(\mathbf{a}_k \mid Q)\ \frac{P(\mathbf{a}_k \mid M_i)}{\sum_{M_l} P(\mathbf{a}_k \mid M_l)}. \tag{7}$$

The first term in (7) is the likelihood of a shapeme conditioned on the query and can be computed as

$$\begin{aligned} &P(\mathbf{a}_k \mid Q) \\ &= \sum_{\mathbf{b}_l} P(\mathbf{a}_k \mid \mathbf{b}_l, Q)\, P(\mathbf{b}_l \mid Q) &\text{(8a)}\\ &\approx \sum_{\mathbf{b}_l} P(\mathbf{a}_k \mid \mathbf{b}_l)\, P(\mathbf{b}_l \mid Q) &\text{(8b)}\\ &\propto \sum_{\mathbf{b}_l} P(\mathbf{a}_k \mid \mathbf{b}_l) &\text{(8c)}\\ &= \sum_{\mathbf{b}_l} \frac{P(\mathbf{b}_l \mid \mathbf{a}_k)\, P(\mathbf{a}_k)}{\sum_{\mathbf{a}_g} P(\mathbf{b}_l \mid \mathbf{a}_g)\, P(\mathbf{a}_g)} &\text{(8d)}\\ &= \sum_{\mathbf{b}_l} \frac{P(\mathbf{b}_l \mid \mathbf{a}_k)}{\sum_{\mathbf{a}_g} P(\mathbf{b}_l \mid \mathbf{a}_g)}, &\text{(8e)}\end{aligned}$$

where (8b) is obtained as in (5b) with the homogeneity assumption, $\mathbf{b}_l$ is the lth spin image of Q. Note that the raw feature likelihood conditioned on the query $P(\mathbf{b}_l \mid Q)$ is assumed to be uniformly distributed in (8b)-(8c). This assumption holds because Q is a point cloud with unknown identity and all features b are equally possible. The probability $P(\mathbf{a}_g)$ is also assumed to be uniformly distributed in (8d)-(8e) because all features are equally possible without knowing anything about the model. Substituting (8e) into (7), $P(M_i \mid Q)$ is proportional to

$$\sum_{\mathbf{a}_k} \sum_{\mathbf{b}_l} \frac{P(\mathbf{b}_l \mid \mathbf{a}_k)}{\sum_{\mathbf{a}_g} P(\mathbf{b}_l \mid \mathbf{a}_g)}\ \frac{P(\mathbf{a}_k \mid M_i)}{\sum_{M_l} P(\mathbf{a}_k \mid M_l)}, \tag{9}$$

where $P(\mathbf{b}_l \mid \mathbf{a}_k)$ is the likelihood of a spin image in the query conditioned on a shapeme. This likelihood is defined as

$$P(\mathbf{b}_l \mid \mathbf{a}_k) = \begin{cases} 1 & \mathbf{a}_k = \arg\min_{\mathbf{a}_g} D(\mathbf{a}_g, \mathbf{b}_l) \\ 0 & \text{otherwise,} \end{cases} \tag{10}$$

where $D(\cdot,\cdot)$ is the distance between two features. Equation (10) says that, if $\mathbf{a}_k$ is the shapeme that is the closest to the spin image $\mathbf{b}_l$, the likelihood of $\mathbf{b}_l$ is one. This is spin image labeling based on a nearest neighbor vector quantization approach. Since the query object may be noisy and obscured by scene clutter, a restricted labeling process defined as follows is preferred.

$$P(\mathbf{b}_l \mid \mathbf{a}_k) = \begin{cases} 1 & D(\mathbf{a}^*, \mathbf{b}_l) \le \alpha\, r^* \\ 0 & \text{otherwise} \end{cases}, \tag{11}$$

where $\mathbf{a}^*$ is the shapeme closest to $\mathbf{b}_l$ and r^* is the median radius of the shapeme as defined in Section 5.1. Theoretically, the const α is determined by χ^2 distribution according to the degrees of freedom of spin images and the confidence. In practice, we found that $\alpha = 2.5$ worked well for our application. Given the value of α, a feature whose distance to a shapeme is larger than $\alpha\, r^*$ will not be assigned to this shapeme, even if it is the nearest to the feature. As a result, spin images severely corrupted by noise or scene clutter will not be assigned to any shapeme. By simply deleting them from the query spin image set, probabilities computed based on the remaining spin images become more reliable. It is then obvious from both (10) and (11) that $\sum_{\mathbf{a}_g} P(\mathbf{b}_l \mid \mathbf{a}_g) = 1$ always. More generally,

$$\sum_{\mathbf{b}_l} \frac{P(\mathbf{b}_l \mid \mathbf{a}_k)}{\sum_{\mathbf{a}_g} P(\mathbf{b}_l \mid \mathbf{a}_g)} = v_k, \tag{12}$$

where v_k is the number of spin images in the query that are labeled by $\mathbf{a}_k$. Finally, substituting (4) and (12) into (9) leads to

$$P(M_i \mid Q) \propto \sum_{k=1}^{m} \left(\frac{v_k\ u_{i,k}}{u_i}\ \gamma_k^{-1} \right), \tag{13}$$

where $\gamma_k = (\sum_{M_l} P(\mathbf{a}_k \mid M_l))/o$ is a normalization factor that counts for the discrimination capability of the kth shapeme and o is the number of model objects. It is obvious that (13) represents the correlation between the model and the query, with each bin scaled by γ_k^{-1}. The normalization factor γ_k can be regarded as the average probability of the shapeme $\mathbf{a}_k$'s occurrence within the whole model database. Equation (13) gives high weights to shapemes that are rare in the database since they have small γ_k. This represents a feature selection process where shapemes that belong to many model objects are down weighted. γ is similar to the $tf \times idf$ weight [19] widely used in the document analysis community.

5.3 Posterior for Projected Histograms of a Partial Query

When the query is partially observed, we can use the method from Section 4 to find its projection $\hat{\mathbf{h}}_i = [\hat{\boldsymbol{u}}_{i,1}, \hat{\boldsymbol{u}}_{i,2}, \cdots, \hat{\boldsymbol{u}}_{i,a}]^T$ in the subspace spanned by all the component histograms of model M_i. This histogram corresponds to a virtual model $\hat{M}_i$ that matches the observed part of the query object. The model posterior can then be computed as

Fig. 8. Examples of the vehicle models used for the test.

$$P(\hat{M}_i \mid Q) \propto \sum_{k=1}^{m} \left(\frac{v_k \ \hat{\boldsymbol{u}}_{i,k}}{\hat{\boldsymbol{u}}_i} \ \gamma_k^{-1} \right). \tag{14}$$

Strictly speaking, γ_k should also be computed according to the virtual models corresponding to the query object. In practice, we find it sufficient to compute γ_k from all the complete model objects in the database.

6 Experimental Results

We have tested our method with a model set of 243 vehicles on a 2GHz PC with 2GB main memory. We will present comparative results of our proposed method against other methods. The following is a list of the analyzed approaches:

1. The proposed histogram projection method, denoted as HP, is described in Section. 1. For the Gibbs sampler, we use 30 iterations (n steps per iteration, where n is the number of segments of each model) and set the temperature const $T = 2$.
2. The nearest neighbor method, denoted as NN, matches each spin image in the query object with all the spin images in the database. A vote for that model is incremented whose spin image is the closest to the query spin image. The total matching score for each model object is computed by w_i/w, where w_i is the number of votes the ith model collects and w is the total number of scene points. See Appendices A and B for details.
3. The locality-sensitive hashing method, denoted as LSH, uses the same voting mechanism as in NN, but uses an approximated nearest neighbor search strategy while speeding up the search. We use eight hash tables. See [14] for details of the LSH method.
4. The prototype-based method, denoted as PR, is detailed in [15].
5. The prototype-based subspace method, denoted as SPR, is detailed in [16]. We use a three-dimensional subspace.

The clustering and the codebook generating process is the same as in reference [9]. The spin image feature is 100D as compared with the 48D texton feature used in [9]. We used 500 prototype features (the center of clusters or the shapemes) for all the prototype-based methods and the shapeme histogram methods. Clustering of almost 1 million spin images into 500 clusters took less than an hour when using the k-means clustering algorithm. Also, as in [9], we observed that the final result is not sensitive to the number of clusters as long as it is larger than a certain number, which is 500 in our case.

6.1 Noise Free Data Results

Fig. 8 shows the facet models of the original objects used for the test. A laser range sensor simulator is used to convert facet models into range images. View points are sampled from a hemisphere centered around the object and the viewing direction is always targeting the center. The spherical coordinate for each view point is denoted as (r, ϕ, θ), where r is the radius, ϕ is the azimuthal angle, and θ is the polar angle. The radius is set to be a constant such that the object occupies the full view of the simulated laser sensor for most of the times. It is therefore ignored in the view point notation hereafter. Each model object is generated by combining eight pieces of point clouds sampled from the view point set of $\{(0, 45°), (45°, 45°), \cdots, (360°, 45°)\}$ that covers the whole object. By constructing an octtree-like spatial structure from the combined point cloud of an object M_i, a set of basis points $\mathcal{P}_i$ is uniformly selected from the (implicit) object surface, and the corresponding set of spin images $\mathcal{B}_i$ is computed (see Section 5.1 for notations). A set of 500 shapemes $\mathcal{A}$ is computed from the set Ω of all spin images from the 243 models. Each object M_i is then segmented into n pieces and a histogram h_i^s is computed for each piece. The model histogram database is then constructed as described in Section 4. The query set contains also 243 views, one for each object. Each view is randomly sampled from $\phi \in \{10°, 20°, 30°\}$ and $\theta \in \{40°, 45°, 50°\}$, respectively, and covers 20-60 percent of the complete object. The average number of model basis points is around 4,000, while the numbers on the query is around 1,500. The radius of spin images in the model and query need to be the same. We have used one query per object, but, since we had 243 queries from different objects, the results presented are statistically meaningful.

Note that camera placement may affect the performance of the algorithm. For example, if the camera is placed such that all the queries contain only the side views of the vehicles, the performance will be lower. The camera configuration in our experiments tried to match the real situations where cameras are placed to collect as much information as possible from a single viewpoint.

Table 1 compares the recognition accuracy of our method against the others, where the accuracy is given with respect to the number of models in the short candidate

TABLE 1
Recognition Accuracy of Our Approach as Compared with Other Methods

Methods	NN	LSH	HP-1	HP-10	HP-30	PR	SPR
Accuracy with top 1 model (%)	97.0	85.2	78.5	87.2	91.1	78.0	84.0
Accuracy with top 3 models (%)	1.0	93.0	88.9	94.8	97.2	87.0	91.0

HP-1, HP-10, and HP-30 are the histogram projection methods with $n = 1, 10$, and 30, respectively. HP-1 is the original shapeme histogram method.

TABLE 2
Time and Storage Space Used by Each Method

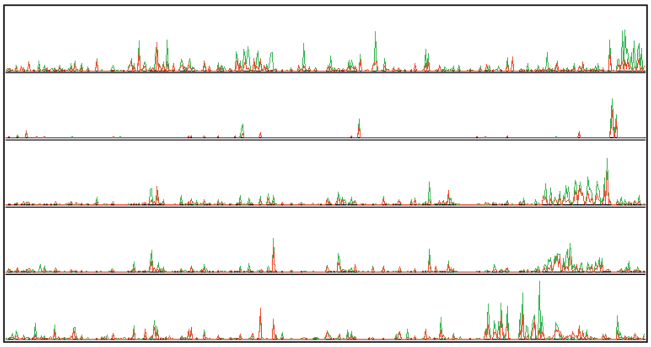
Methods	NN	LSH	HP-1	HP-10	HP-30	PR	SPR
Time (sec)	240	6	1.7	2.7	3.8	1.5	4.1
Space (MB)	480	500	0.6	6	18	1	720

list. As compared with other HP methods, the one with $n = 30$ components produces the best result. If we represent each model as a single histogram, i.e., $n = 1$, the HP method degenerates into the original histogram matching method. The bad result is expected since histogram matching in its original form does not handle partially observed queries. It can be also observed that increasing n from 10 to 30 does not improve the accuracy as dramatic as from 1 to 10. In fact, the HP method with $n = 50$ produces almost the identical result as $n = 30$.

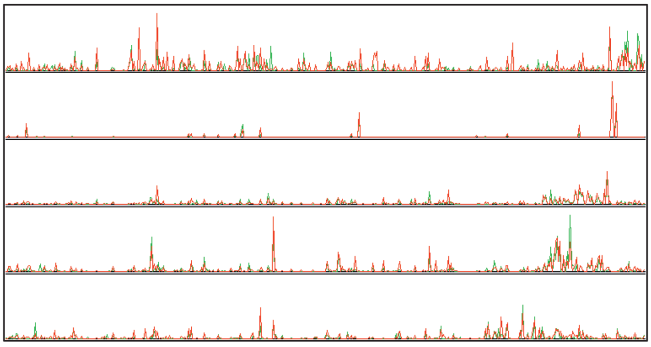
From Table 1, it can also be seen that the LSH method does not work well for this set of databases. On the other hand, the accuracy of the HP method with $n = 30$ and three candidates is close to the nearest neighbor method and is much better than other approaches.

Table 2 compares the time and storage space used by each method. The time reported is the average time for a single query, while the space is for the whole model database. It can be seen that the speed of the HP method is, in general, slower than the PR method, but is still more than 60 times faster than the nearest neighbor approach. It is also faster than the LSH method. It can also be observed from the table that the HP method requires only a small fraction of the storage space needed by both the NN method and the LSH method.

One interesting aspect from the table is that the sampling-based histogram projection process does not bring in much overhead in terms of the computational time. This is because our Gibbs sampler usually converges within 30 iterations and each iteration involves a trivial update of the objective function. Fig. 9 shows some snapshots of this iterative convergence process. Fig. 10 compares query histograms with the corresponding complete model histograms and the projected histograms. Obviously, the projected version is more similar to the query.



(a)



(b)

Fig. 10. The accuracy of the projected histograms (best viewed with color). (a) Model histograms (green) overlaid with query histograms (red). (b) Projected histograms (green) overlaid with query histograms (red). The projected histograms are much more similar to the query histograms.

6.2 Data with Synthesized Clutter and Noise

To test the performance of our method under noise, we added Gaussian noise to the query objects, and computed the accuracy with noise level from 0.0 cm to 0.25 cm. The results are shown in Table 3. It can been seen that the accuracy with the top one model decreases gracefully as the level of noise increases. On the other hand, the accuracy with the top three models does not decrease too much as the noise level increases.

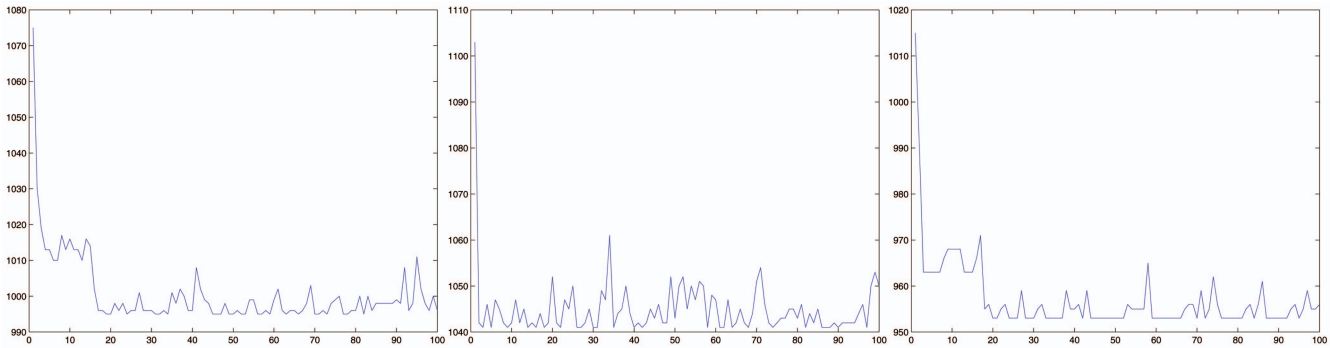


Fig. 9. Snapshots of the iterative sampling process for histogram projection of three queries to their corresponding models. The x-axis is the number of iterations ($0 - 100$) and the y-axis is the value of the objective function in (3). Thanks to the rapid convergence of the process, the best solution can usually be found within 30 iterations.

TABLE 3
Accuracy of the HP-30 Method with Different Level of Gaussian Noise Added to the Query Point Cloud

Noise in cm	0.0	0.05	0.10	0.15	0.20	0.25
Accuracy with top 1 (%)	91.1	88.5	86.8	86.4	82.7	80.3
Accuracy with top 3 (%)	97.2	97.2	94.2	93.4	93.0	93.8

TABLE 4
Accuracy of the HP-30 Method with Different Level of Synthesized Structural Noise

λ	0.0	0.1	0.2	0.3	0.4	0.5
Accuracy with top 1 (%)	91.1	89.0	87.3	84.0	76.5	70.1

The larger the λ, the more noise is added into the query.

To test the performance of our method under scene clutter, we added to each query histogram h_i^q a percentage of another query histogram h_j^q, $i \neq j$, and form a new query $\tilde{h}_i^q = h_i^q + \lambda h_j^q$, which now contains a certain degree of "structural" noise from another query object. This to a certain extent simulates the situation where two objects are present in the scene (without occlusion). We varied λ from 0.1 to 0.5 in the test and the result for the HP method ($n = 30$) is shown in Table 4. It can be seen that the accuracy of the method decreases gracefully as the level of noise increases. Note, here, the decrease in accuracy is mainly caused by the confusion between the projected histogram and the query histogram. In most cases, we observed that the projected histograms were still very accurate. To see this, Fig. 11 shows an example where Fig. 11b is the sum of Figs. 11c and 11d, i.e., $\lambda = 1$. If we project Fig. 11b onto the histogram subspace of the model corresponding to Fig. 11c, we get the projection in Fig. 11a, which indeed looks similar to Fig. 11c.

7 Conclusion and Discussion

We have proposed a two-step approach to address the problem of shapeme histogram matching with partially observed objects. We have applied the proposed method to the problem of 3D object recognition with range image. We then compared our method with other approaches and demonstrated its advantages on a commercially available database of 243 models.

One concern of the proposed approach is that it causes problems when query object cuts through model segments. This is indeed the weakness of an approach that relies on global histogram matching. In fact, it was this weakness that motivated the development of this approach. The original method using the histogram of the whole model can be regarded as the worst case where the number of model segments is equal to 1 and the query has to cut through this single piece. When the number of segments increases, more and more model segments escape the "cutting through" because "cutting through" only happens on the boundary of the query. This has been verified by the experiments in the paper. As we can see from Table 2, when the number of segments increases from 1 (HP-1) to 10 (HP-10), the absolute error drops down about 6 percent. The error drops down further, 2.4 percent, when the number of segments increases to 30 (HP-30). This clearly shows the benefit of the proposed approach. Moreover, when we increase the number from 30 to 50, the accuracy is almost identical to HP-30. This demonstrates that, after the number of segments reaches a certain number, the "cutting through" issue is no longer the dominant source of the errors.

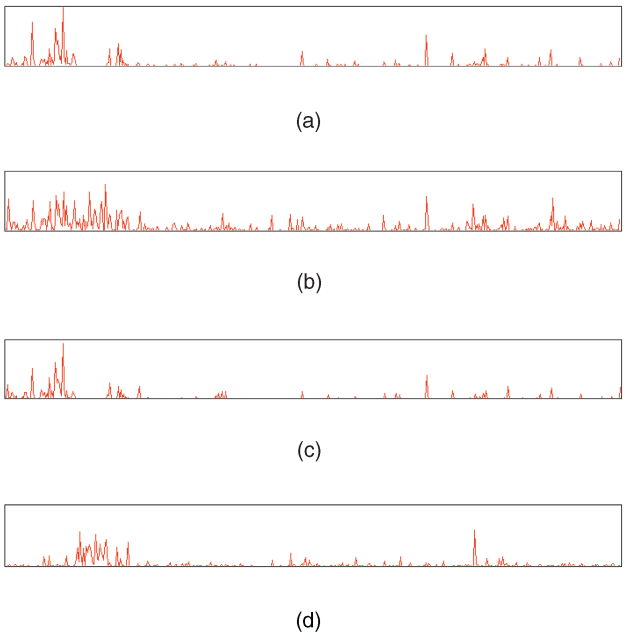


Fig. 11. Projection of a query histogram with large structural noise. See text for details.

Appendix A

A General Bayesian Framework for Matching Objects Represented as Collections of Features

This appendix describes a general framework for matching a query object represented as a collection of features. We will show how this problem is linked to the traditional classification problem. In the following discussion, the query object will be represented as a set of spin images $\mathcal{Q} = \{\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_w\}$, where w is the number of spin images, in the query object. Other notations such as M_i, Q, and $\mathcal{B}_i$ have been defined in Section 5.1. The posterior probability of a model given a query can be computed as:

$$P(M_i \,|\, Q)$$
$$= \sum_k P(h = k \,|\, Q)\; P(M_i \,|\, h = k,\, Q) \qquad (15a)$$
$$= \sum_k P(h = k \,|\, Q)\; P(M_i \,|\, \mathbf{q}_k), \qquad (15b)$$

where $h \in \{1, 2, \cdots, N\}$ is a random variable of which feature in $\mathcal{Q}$ is selected. Note that (15b) sustains because the event of the kth feature is selected in $\mathcal{Q}$, i.e., $\{h = k, Q\}$, is equivalent to the event of q_k is given. It is obvious that $P(h = k \,|\, Q)$ is uniform; from (15b), we have:

$$P(M_i \mid Q) = \lambda \sum_k P(M_i \mid \mathbf{q}_k), \tag{16}$$

where λ is a const for all models. We then have:

$$P(M_i \mid Q) = \frac{\sum_k P(M_i \mid \mathbf{q}_k)}{\sum_i \sum_k P(M_i \mid \mathbf{q}_k)}. \tag{17}$$

This equation provides a framework of computing the posterior of a model given a set of query features from the posterior of a model given a single query feature. The latter fits into the traditional classification problem and, therefore, can be computed with the existing methods such as nearest neighbor, SVM, etc.

APPENDIX B

NEAREST NEIGHBOR CLASSIFIER FOR MATCHING OBJECTS REPRESENTED AS COLLECTIONS OF FEATURES

Nearest neighbor classifier uses the following rule to compute $P(M_i \mid \mathbf{q}_k)$:

$$P(M_i \mid \mathbf{q}_k) = \begin{cases} 1 & \tilde{\mathbf{q}}_k \in \mathcal{B}_i \\ 0 & \text{otherwise,} \end{cases} \tag{18}$$

where $\tilde{\mathbf{q}}_k$ is the nearest neighbor of $\mathbf{q}_k$ over the whole model database $\bigcup_i \mathcal{B}_i$. From (17) and (18), it is obvious that:

$$P(M_i \mid Q) = w_i / w, \tag{19}$$

where w_i is the number of scene features that are the closest to the features of the ith model object and w is the total number of the scene features. This is the nearest neighbor method used in our experiment.

APPENDIX C

BASIS POINT SELECTION

Ideally, basis points are sampled uniformly from the object surface to compute spin images. Since our input is a set of unorganized 3D points, the data needs to be voxelized to facilitate the computation of spin images. During the voxelization process, the space occupied by the point cloud is discretized into a volume of voxels with equal size. We then uniformly select a set of voxels from all the occupied voxels. At each of these voxels, we compute the centroid and normal by fitting a plane to the points inside the voxel and it neighbors within a certain distance. The centroids define the basis points and are associated with the center voxel used for plane fitting.

REFERENCES

[1] F. Stein and G. Medioni, "Structural Indexing: Efficient 3D Object Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 14, pp. 125-145, 1992.

[2] A.E. Johnson and M. Hebert, "Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 5, pp. 433-449, May 1999.

[3] M. Kazhdan and T. Funkhouser, "Harmonic 3D Shape Matching," *Technical Sketch, Proc. SIGGRAPH,* 2002.

[4] C. Dorai and A.K. Jain, "COSMOS-A Representation Scheme for 3D Free-Form Objects," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 10, pp. 1115-1130, Oct. 1997.

[5] G. Mori, S. Belongie, and J. Malik, "Shape Contexts Enable Efficient Retrieval of Similar Shapes," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 723-730, 2001.

[6] B. Matei, Y. Shan, H.S. Sawhney, R. Kumar, D. Huber, M. Hebert, and Y. Tan, "Rapid Object Indexing Using Locality Sensitive Hashing and Joint 3D-Signature Space Estimation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* submitted for publication.

[7] Y. Shan, B. Matei, H.S. Sawhney, R. Kumar, D. Huber, and M. Hebert, "Linear Model Hashing and Batch RANSAC for Rapid and Accurate Object Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2004.

[8] Y. Shan, H.S. Sawhney, B. Matei, and R. Kumar, "Partial Object Matching with Shapeme Histograms," *Proc. European Conf. Computer Vision,* 2004.

[9] T. Leung and J. Malik, "Representing and Recognizing the Visual Appearance of Materials Using Three-Dimensional Textons," *Int'l J. Computer Vision,* vol. 43, no. 1, pp. 29-44, 2001.

[10] S. Lazebnik, C. Schmid, and J. Ponce, "Affine-Invariant Local Descriptors and Neighborhood Statistics for Texture Recognition," *Proc. Int'l Conf. Computer Vision,* 2003.

[11] S. Ruiz-Correa, L.G. Shapiro, and M. Meila, "A New Signature-Based Method for Efficient 3-D Object Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 769-776, 2001.

[12] S. Ruiz-Correa, L.G. Shapiro, and M. Meila, "A New Paradigm for Recognizing 3-D Object Shapes from Range Data," *Proc. Int'l Conf. Computer Vision,* 2003.

[13] C.-S. Chen, Y.-P. Hung, and J.-B. Cheng, "RANSAC-Based DARCES: A New Approach to Fast Automatic Registration of Partially Overlapping Range Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 11, pp. 1229-1234, Nov. 1999.

[14] A. Gionis, P. Indyk, and R. Motwani, "Similarity Search in High Dimensions via Hashing," *VLDB J.,* pp. 518-529, 1999.

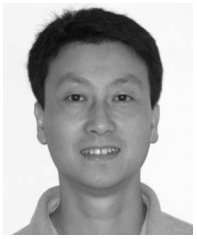
[15] L.I. Kuncheva and J. Bezdek, "Presupervised and Postsupervised Prototype Classifier Design," *IEEE Trans. Neural Networks,* vol. 10, no. 5, pp. 1142-1152, 1999.

[16] J.-T. Chien, "Discriminant Waveletfaces and Nearest Feature Classifiers for Face Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 12, pp. 1644-1649, Dec. 2002.

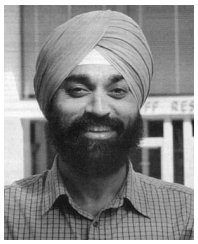
[17] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 6, no. 6, pp. 721-741, 1984.

[18] D.D. Lee and H.S. Seung, "Learning the Parts of Objects by Non-Negative Matrix Factorization," *Nature,* vol. 40, pp. 788-791, 1999.

[19] G. Salton, "Developments in Automatic Text Retrieval," *Science,* vol. 253, pp. 974-980, Aug. 1991.



Ying Shan received the BE degree in chemical engineering, focusing on automatic process control, from Zhejiang University, People's Republic of China in 1990, and the MS and PhD degrees in computer science from Shanghai Jiaotong University, People's Republic of China in 1997. He is currently a senior member of technical staff at Sarnoff Corporation's Vision and Learning Laboratory, where has contributed to, initiated, led, and successfully delivered a number of government and commercial projects. He does research and development in the fields of computer vision, machine learning, object recognition, and computer graphics, with applications in 3D object and face modeling, 2D/3D image registration, video object identification, video data mining, video surveillance, and 3D object recognition. He was a research assistance at Hong Kong Polytechnic University, Hong Kong, from 1996 to 1997. He was a postdoctoral fellow at Nanyang Technological University, Singapore, from 1997 to 1999, and a postdoctoral researcher at Microsoft Research, from 1999 to 2001. He has published more than 25 papers, holds four US patents, and has many others pending. He has served as a reviewer for top journals such as the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Multimedia*, *IEEE Transactions on Image Processing*, *IEEE Computer Graphics and Applications*, and the *International Journal of Computer Vision*. He was on the program committee at numerous international conferences such as ACCV, CVPR, ICCV, and ECCV. He was the recipient of Sarnoff's Recognition Award in 2003, Innovation Award in 2003, and the Innovation Award in 2004. Dr. Shan is a senior member of the IEEE and the IEEE Computer Society.



Harpreet S. Sawhney received the PhD in computer science in 1992 from the University of Massachusetts, Amherst, focusing on computer vision. He is the technical director of and leads the Vision and Learning Technologies Lab. at the Sarnoff Corporation. His areas of interest are object recognition, motion video analysis, 3D modeling, vision and graphics synthesis, video enhancement, video indexing, data mining, and compact video representations. Since 1995, he has led government and commercial programs in immersive telepresence, image-based 3D modeling, video object fingerprinting, video mosaicing, georegistration, 2D and 3D video manipulation, and object recognition. Dr. Sawhney was one of the key technical contributors toward the founding of two Sarnoff spinoffs, VideoBrush Inc. and Lifeclips Inc. Between 1997 and 2004, he was awarded the Sarnoff Technical Achievement Awards seven times for his contributions in video mosaicing, video enhancement, 3D vision and immersive telepresence. Between 1992 and 1995, he led video annotation and indexing research at the IBM Almaden Research Center in San Jose, California. He is an associate editor for the *IEEE Transactions on Pattern Analysis and Machine Intelligence*. He has also served on the program committees of numerous computer vision and pattern recognition conferences. He has published more than 60 papers and holds 15 patents. He is a member of the IEEE and the IEEE Computer Society.

Bogdan Matei received the Dipl. Engn. and MSc degrees in electrical and computer engineering from the Polytechnic University of Bucharest, Romaniac in 1994 and 1995c respectively, and the PhD degree in electrical engineering from Rutgers University in 2001. Between 1994 and 1997, he was with the Department of Electronics and Telecommunications at the Polytechnic University of Bucharest as a teaching fellow and research associate. In 1994, he was awarded a TEMPUS scholarship funded by the European Union at Polytechnics of Turin, Italym to research optical character recognition using neural networks. He had received research fellowships at the Technical University of Darmstadt, Germanym in 1995 and 1996. From 1997 until 2001, he was with the Center for Advanced Information Processing affiliated with Rutgers University, researching the application of modern statistical methods for optimal parameter estimation under heteroscedastic noise and performance evaluation of computer vision algorithms. Since 2001, he has been with the Vision Technologies Group at Sarnoff Corporation, first as a member of the technical staff and lately as a senior member of the technical staff. His research interests include object recognition, statistical learning, video-based aerial and ground autonomous navigation, real-time aerial video surveillance, and georegistration. He has coauthored more than 10 papers and book chapters in the areas of computer vision and holds several patents awarded or pending. He was on the program committees at numerous international conferences and workshops such as CVPR, ICCV, and ECCV. He received the Best Student Paper award at the IEEE Computer Vision and Pattern Recognition Conference in 1999. He is a member of the IEEE and the IEEE Computer Society.

Rakesh (Teddy) Kumar received the PhD degree in computer science from the University of Massachusetts at Amherst in 1992. He received the MS degree in Electrical and Computer Engineering from the State University of New York, Buffalo and the BTech degree in electrical engineering from the Indian Institue of Technology-Kanpur. He is currently the technical director of the Vision and Robotics Laboratory at Sarnoff Corporation, Princeton, New Jersey. Prior to joining Sarnoff, he was employed at IBM. His technical interests are in the areas of computer vision, computer graphics, image processing, and multimedia. At Sarnoff, he has been directing and performing commercial and government research and development projects in the areas of video surveillance and monitoring, video and 3D exploitation and analysis, object recognition, immersive tele-presence, 3D modeling, medical image analysis, and multisensor registration. He has been one of the principal founders from Sarnoff for multiple spin-off and spin-in companies: VideoBrush, LifeClips, and Pyramid Vision Technologies. He was an associate editor for the *IEEE Transactions on Pattern Analysis and Machine Intelligence* from 1999 to 2003. He has served in different capacities on a number of computer vision conferences and US National Science Foundation review panels. He has coauthored more than 40 research publications and has received more than 18 patents, with numerous others pending. He is a member of the IEEE and the IEEE Computer Society.