PIMs and Invariant Parts for Shape Recognition

Zhibin Lei
Bell-Labs, Lucent Technologies
700 Mountain Avenue
Murray Hill, NJ 07974

Tolga Tasdizen
Department of Engineering
Brown University
Providence, RI 02912

David B. Cooper
Department of Engineering
Brown University
Providence, RI 02912

Abstract

We present completely new very powerful solutions to two fundamental problems central to computer vision. 1. Given data sets representing C objects to be stored in a database, and given a new data set for an object, determine the object in the database that is most like the object measured. We solve this problem through use of PIMs ("Polynomial Interpolated Measures"), which is a new representation integrating implicit polynomial curves and surfaces, explicit polynomials, and discrete data sets which may be sparse. The method provides high accuracy at low computational cost. 2. Given noisy 2D data along a curve (or 3D data along a surface), decompose the data into patches such that new data taken along affine transformations or Euclidean transformations of the curve (or surface) can be decomposed into correponding patches. Then recognition of complex or partially occluded objects can be done in terms of invariantly determined patches. We briefly outline a low computational cost image-database indexing-system based on this representation for objects having complex shape-geometry.

1 Introduction

Denote dth degree *explicit* polynomials in (x, y) and (x, y, z) by

$$f(x,y) = \sum_{i,j \geq 0; i+j \leq d} a_{ij} x^i y^j$$

and

$$f(x,y,z) = \sum_{i,j,k \geq 0; i+j+k \leq d} a_{ijk} x^i y^j z^k$$

respectively. By $2D$ *implicit polynomial* (IP) curves and $3D$ IP surfaces we mean the sets of (x, y) and (x, y, z) points, respectively, where the explicit polynomials are equal to 0. IP $2D$ curves (and $3D$ surfaces) have many desirable features for use in computer vision [1, 2, 3], but they have also had some weaknesses. Among these weaknesses have been: computation times on the order of a second to a few minutes for fitting a $2D$ curve (or $3D$ surface) to a few hundred data points; insufficient repeatability, under certain circumstances, in fitted IP coefficients for shapes that have undergone Euclidean or affine transformations and for which the data is somewhat variable; sometimes weird representations when an IP that cannot represent a data set accurately is fit to the data; an inability to represent and compare geometrically-complex shapes. Of course, the last problem is the Holy Grail of computer vision – the search for parts which can be computed invariantly and at low computational cost.

In this paper we present solutions to the preceding problems. Specifically, we present a completely new concept of IP representations based on the approximation of the *distance transform* for a data set by an explicit polynomial, and the resulting *linear* shape representation and recognition theory that flows from this. This new IP technology is ideally suited to representing and recognizing complicated $2D$ (and $3D$) shapes subject to partial occlusions and missing data. The material presented is the following. 1) How to fit polynomials of modest degree, e.g., $4th$ degree, quickly (milliseconds or orders of magnitude less) and fit polynomials such that zero sets and coefficients are roughly the same under perturbations in data, e.g., noise, slight differences in patch end points, etc. The method also fits IP curves of degrees up to $18th$ easily, though at greater computational cost. 2) How to compare shapes or data sets represented by implicit polynomials accurately and at low computational cost. 3) How to choose patches invariantly. That is, we want to store a set of patches with each object in a database, and each of several patches that are computed for an object to be recognized should be roughly the same as a patch stored with the object in the database. With these, we lay down a foundation that enables a technology based on implicit polynomial curves and surfaces for various object representation/recognition applications.

•**Application of Invariant Patches for Indexing into Pictorial Databases** To be able to access the vast amount of pictorial information now available, many content based query and indexing systems have been introduced [4, 5]. Most of the current pictorial database indexing schemes are appearance-rather than geometry-based. Geometry-based indexing allows rapid queries to be performed on large pictorial databases containing images with complex, rich shape-structure [6]. In Sec. 3.2 we briefly sketch an image database indexing system based on the new IP technology described in this paper. The approach should permit geometric-based indexing into large image databases of the order of 10^5 or 10^6 images in real time.

2 Polynomial Interpolated Measures

2.1 $3L$ Polynomial Fitting

We assume a set $\Gamma_0 = \{z_k : 1 \le k \le N\}$ of measurement points can be well represented by an implicit polynomial of degree d'. Note, z_k denotes (x_k, y_k). Consider representation by a polynomial of degree d, with $d \le d'$. Our representation is as follows. For each point z on a data curve Γ_0 generate two other points each at a distance c from z and in a direction perpendicular to the curve, each point on a different side of the curve. We denote these two sets of artificially generated points by Γ_c and Γ_{-c}. Note Γ_c is a set of points on one side of the curve, and Γ_{-c} is a set of points on the other side of the curve. Denote the union $\Gamma_{-c} \cup \Gamma_0 \cup \Gamma_c$ by Γ. Let $g(x, y)$ denote the *Distance Transform of* Γ_0. $g(x, y)$ takes value 0 on Γ_0 and $-c$ and $+c$ on the artificially generated points Γ_{-c} and Γ_c, respectively. We now determine a polynomial $f(x, y)$ of degree d such that its zero set approximates Γ_0. We do this by choosing $f(x, y)$ to be a least squares approximation to $g(x, y)$ on Γ. Specifically, $\hat{f}(x, y)$ is the d'th degree polynomial for which

$$\begin{gathered} \textstyle\sum_{(x,y)\in\Gamma}[f(x,y) - g(x,y)]^2 = \\ \textstyle\sum_{k=1}^{N}[g(x_k,y_k) - \sum_{0\le i,j;i+j\le d} a_{ij}{x_k}^i{y_k}^j]^2 \end{gathered} \tag{1}$$

is minimum. We call $\hat{f}(x, y)$, the polynomial approximation to the Distance Transform in the vicinity of Γ_0, the $3L$ (three level sets) polynomial fit [7]. Fig. 1 shows overlays of the original data and the IP curves of various degrees fit to three shapes using the $3L$ algorithm.

The limited repeatability in the coefficients of fitted polynomials using previous methods occurs because a polynomial is a function of all $x, y : -\infty < x, y < \infty$. Hence, estimating coefficients along a *finite* curve Γ_0 and expecting the resulting coefficient values to be repeatable in the presence of modest changes in the noise and the ends of the data set interval is unrealistic. In addition, there is nothing in previous fitting procedures to prevent singularities in $\hat{f}(x, y)$ from occuring in the vicinity of the data curve Γ_0. These singularities are a major cause of instability and repeatability in the fitted polynomial parameters and zero sets. Our $3L$ approach tends to prevent the occurrence of singularities in $\hat{f}(x, y)$ in the vicinity of Γ_0 since it discourages the occurrence of a local minimum, maximum, or saddle near Γ_0, and the $3L$ fitting uses data over a ribbon rather than a curve because of the artificially generated points. (Singularities occur at these extrema.) An effect of $3L$ fitting is that of applying a constraint in the fitting, namely, encouraging the gradient of $f(x, y)$ in the vicinity of the data points to be perpendicular to Γ_0 and of magnitude 1.

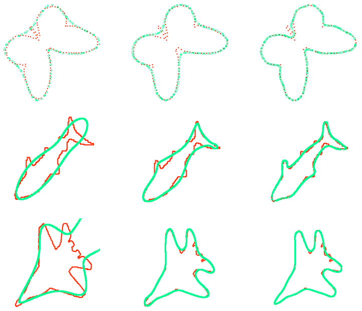


Figure 1: 3L fitting with degrees 4, 8 and 12.

2.2 PIMs (Polynomial Interpolated Measures)

The problem here is: given measured object boundary data Γ_0, check to see which stored data set $\Gamma_{1,0}, \Gamma_{2,0}, ..., \Gamma_{L,0}$ it is closest to. $\Gamma_{l,0}$, $1 \le l \le L$, contains N_l points, which may be noisy measurements of a curve, and Γ_0 may be a new noisy measurement. Hence, for some $\tilde{l}$, $\Gamma_{\tilde{l},0}$ may represent the same shape but not necessarily have any points in common. How can we do computationally fast robust comparison? One approach is to compute the average squared Euclidean distance from Γ_0 to each of the data sets $\Gamma_{l,0}$. The Euclidean distance can be computed by storing the Distance Transform $g_l(x, y)$ for each object in the database, or, more effeciently, its polynomial approximation $\hat{f}_l(x, y)$, and computing $N^{-1}\sum_{\{(x,y):(x,y)\in\Gamma_0\}} \hat{f}_l^2(x, y)$. In this section, we develop a powerful generalization – PIM (Polynomial Interpolated Measure) – which compares the distance between data sets in terms of the distance between their fitted polynomial coefficients weighted in a certain way. It provides a new concept for comparing data curves and leads to many useful results (e.g. orthogonal decomposition [8]).

Consider data sets Z_1 and Z_2 containing N_1 and N_2 points, respectively. The fitted IP (implicit polynomial) curve models for these are $f_1(x,y) = \sum_{0 \leq i,j;i+j \leq 4} a_{1ij} x^i y^j$ and $f_2(x,y) = \sum_{0 \leq i,j;i+j \leq 4} a_{2ij} x^i y^j$ (for simplicity we use 4th degree IPs here). Define the dissimilarity (distance) between Z_1 and Z_2 as

$$\begin{aligned} dist(Z_1, Z_2) &= \parallel f_1 - f_2 \parallel_{Z_1 \cup Z_2} \\ &\equiv \tfrac{1}{N_1} \textstyle\sum_{(x,y) \in Z_1} (f_1(x,y) - f_2(x,y))^2 + \\ &\quad \tfrac{1}{N_2} \textstyle\sum_{(x,y) \in Z_2} (f_1(x,y) - f_2(x,y))^2 \\ &= \tfrac{1}{N_{1,2}} \textstyle\sum_{(x,y) \in Z} [\sum_{0 \leq i,j;i+j \leq 4} (a_{1ij} - a_{2ij}) x^i y^j \quad]^2 \end{aligned} \tag{2}$$

Here $Z = Z_1 \cup Z_2$ and $N_{1,2} = N_1 + N_2$.

Denote $\mathbf{a} = (a_{00} \; a_{10} \; a_{01} \; a_{20} \; ... \; a_{04})^t_{15}$, $\mathbf{X} = (1 \; x \; y \; x^2 \; ... \; y^4)^t_{15}$ and the elements of Z as $\{(x_k, y_k) : 1 \leq k \leq N_{1,2}\}$.

$$\begin{aligned} dist(Z_1, Z_2) &= \tfrac{1}{N_{1,2}} \textstyle\sum_{k=1}^{N_{1,2}} (\Delta \mathbf{a}^t \mathbf{X}_k)^2 \\ &= \Delta \mathbf{a}^t (\tfrac{1}{N_{1,2}} \textstyle\sum_{k=1}^{N_{1,2}} \mathbf{X}_k \mathbf{X}_k^t) \Delta \mathbf{a} \equiv \Delta \mathbf{a}^t M \Delta \mathbf{a} \geq 0 \end{aligned} \tag{3}$$

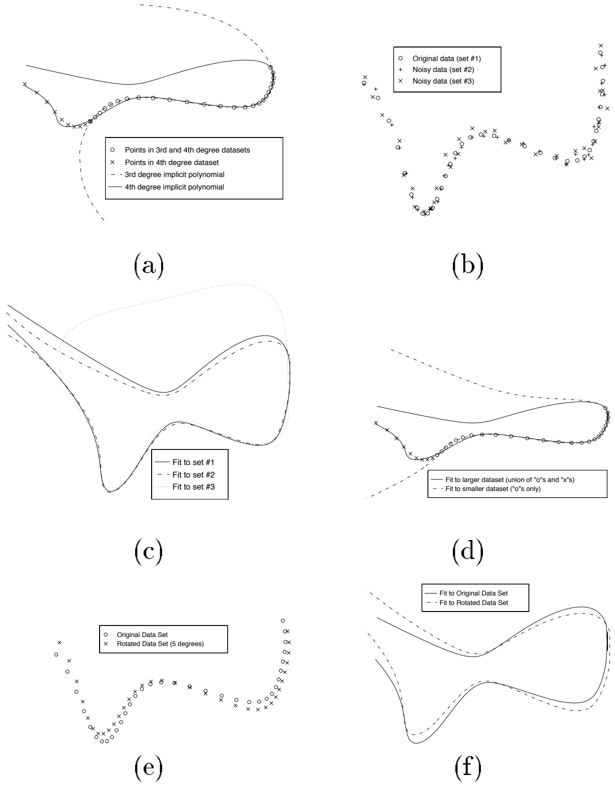
Here M is a nonnegative definite symmetric matrix. $dist(Z_1, Z_2)$ can be 0 for $Z_1 \neq Z_2$, i.e., different samplings of the same continuous curve. We can define such sets to be an equivalence class so that the requirements of $dist(Z_1, Z_2)$ being a metric are not violated. We call it *Polynomial Interpolated Measure* or *PIM* for short. Hence, eqn. 3 gives us *the approximate mean squared distance between two curves at a local data set in terms of a Mahalanobis distance between the IP curve coefficients.*

PIM can be generalized to other operations in the coefficient space. Assume $\mathbf{a}, \mathbf{b} \in \Re^{15}$ are two 4th degree IP coefficient vectors. We define an inner product $\mathbf{a} \cdot \mathbf{b}$ as $(\mathbf{a}, \mathbf{b})_{PIM} = \mathbf{a}^t M \mathbf{b}$. Hence the length $\parallel \mathbf{a} \parallel_{PIM} \equiv (\mathbf{a}, \mathbf{a})^{\frac{1}{2}}_{PIM} = (\mathbf{a}^t M \mathbf{a}_{PIM})^{\frac{1}{2}}$. Thus eqn. 3 can be written as

$$dist(Z_1, Z_2) = (\Delta \mathbf{a}, \Delta \mathbf{a})_{PIM} = \parallel \mathbf{a}_1 - \mathbf{a}_2 \parallel_{PIM} \tag{4}$$

We say two polynomials are orthogonal with respect to a given data set Z if $(\mathbf{a}, \mathbf{b})_{PIM} = \mathbf{a}^t M_Z \mathbf{b} = 0$.

For equations. 2 to 3 it is useful to think of these sums,e.g., $\frac{1}{N_{1,2}} \sum_{(x,y) \in Z} [f_1(x,y) - f_2(x,y)]^2$, as the integration of $[f_1(x,y) - f_2(x,y)]^2$ with respect to a discrete measure taking value $\frac{1}{N_{1,2}}$ at each point (x,y) in Z. These measures for Z_1 and Z_2 are then shape measures, and the sums are functionals on explicit polynomial space defined by these shape measures. Fig. 2 demonstrates the use of PIM distance measure for four cases.



(a) (b) (c) (d) (e) (f)

Figure 2: Demonstration of the use of PIM.

Case 1 Fig. 2(a) shows two data sets: one consisting of the "o" symbols only and one that is the union of "o" and "x" symbols. Two IPs of degrees 3 and 4 are fitted to these sets, respectively. Clearly, the two IPs agree very well on the data set composed only of "o"s, whereas over the larger data set the IPs are very different. Computing the PIM distance using the smaller data set gives a value of *3.8227* and using the larger data set gives a value of *2947.3*. Thus, PIM permits comparison of two polynomial zero sets in terms of their coefficients even when the degrees of the polynomials are different.

Case 2 Fig. 2(b) shows three sets of data: Set #1 is the original data set, Set #2 is a noisy version of the same data and Set #3 has more noise than Set #2. Fig. 2(c) are the three IPs fitted to the corresponding data sets; The PIM distance between Set #1 and Set #2 is *0.8353* and the distance between Set #1 and Set #3 is *4.7550.*

Case 3 Fig. 2(d) shows the same data sets as in

2(a), but now *4*'th degree IPs are fitted to both. Again the two IP curves agree quite well on the smaller data set, but not on the bigger data set. The coefficient vectors for the two IPs are remarkably different. The PIM distance is 3.5099 using the smaller data set where the IP curves agree well, and 3995.2 using the larger data set. PIM provides a way to measure local similarity around data sets when global curve behaviors are significantly different.

Case 4 The datasets shown in 2(e) are equivalent up to a rotation of 5 degrees which might be the case if the patch in the database and the patch to be recognized cannot be aligned exactly. The PIM distance for these two sets is 467.48 which suggests that the sets are not completely dissimilar.

We are presently extending *PIM* measure to recognition based on invariants and pose estimation.

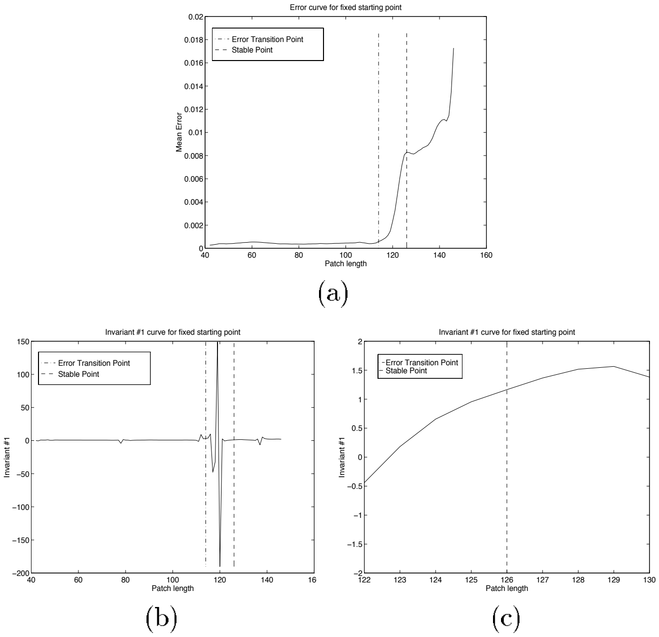
3 Invariant Patches and Parts

3L Fitting allows an IP representation to be arbitrary close to the shape given that the degree of IP used be high enough. In many computer vision and/or content-based indexing applications, data may not be available along the entire shape because of occlusion or missing data. Hence, it is important to be able to extract invariant patches with given degree-complexity from the object shape measurements. These are useful object features for shape recognition or indexing.

3.1 Maximum Invariant Patches and Parts

There is not general agreement on the concept of *patches* and *parts*, but there are a number of different approaches to the problem. Among the bases for doing this are: (a) general shape features [9]; (b) geometric primitives; (c) features pertinent to the restricted set of objects under consideration in a particular application; (d) area or volume scale ; (e) complexity. Our view of patches and parts is that they are simply chunks of $2D$ curves or $3D$ surfaces that can be found reliably in any data in which they are visible and irrespective of sensor viewing direction.

Our approach is the following. Choose a degree d for a polynomial. Choose any point s on an object shape curve. Now determine the curve patch starting at s and having maximum length $l(d, s)$ such that the patch can be "well fit" by a dth degree implicit polynomial. Now consider a Euclidean or affine transformation of the original curve. Take point s' on it corresponding to point s on the original curve. Choose a maximum length patch on this curve starting at point s', where by maximum length we again mean that the patch is of greatest length and still well fit by a dth degree IP. This patch will be the Euclidean or affine



(a)

(b)

(c)

Figure 3: (a) Mean error as a function of patch length; 118 is a first estimate of $l(s, d)$. (b) The value of invariant #1, displayed at coarse scale, as a function of patch length. Notice instability in vicinity of length 118. Locally stable regions are at slightly greater or smaller locations. (c) The value of invariant #1 displayed at finer scale in vicinity of length 118. $l(s, d)$ is chosen to be 126.

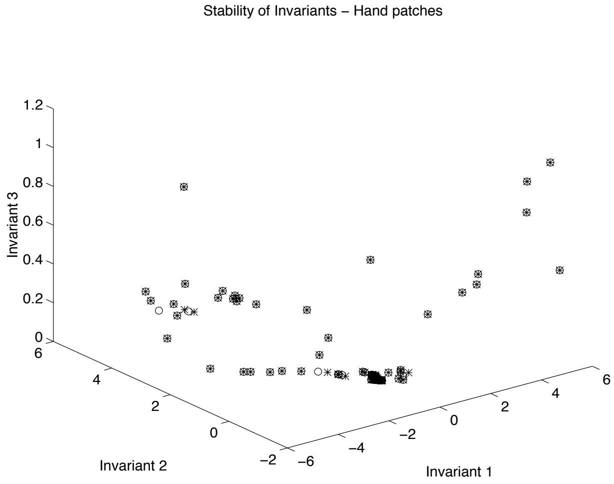


Figure 4: Algebraic invariants of maximum length invariant patches for a hand shape (circles) and its Euclidean transform (crosses). Algebraic invariants of 96% of the patches of the transformed shape are almost identical in value to those of the original patches.

transformation, respectively, of the patch on the original curve. *Hence, this is a way of choosing patches that are invariant to Euclidean or affine transformations.* Note, the *maximum length invariant patch* can be found with a reasonable amount of computation because our fitting is linear least squares, so that the fitting can actually be done recursively with a small amount of computation. If this is done starting at every point on a shape curve, the resulting set of maximum length invariant patches is our set of "parts" for use in recognition or indexing or other purposes. Of course, these "parts" do not have any physical meaning.

To find a maximum length patch, starting at point s, we use 3L fitting and look at the mean square error as a function of increasing patch arc length. This error measure will be roughly constant as long as a *dth* degree polynomial fits the data well. When a *dth* degree polynomial no longer is satisfactory, the error starts to increase rapidly, and that is where we stop. A typical curve of error measure as a function of patch arc length is shown in Fig. 3a. Finally, if the end goal is indexing based on algebraic invariants, we need to make sure that the invariants to be used take values that are independent of small changes in the invariant patch length. (An algebraic invariant is a function of the IP coefficients that is determined only by shape and is independent of position. See [2] for some examples.) To ensure this happens, in the vicinity of the aforementioned estimate of maximum invariant patch length, we perturb the patch length slightly until we find a length for which the values of the invariants are insensitive to small changes in patch length. This is a final maximum length invariant patch that we use. Fig. 4 displays the maximum length invariant patch determination procedure for a single invariant in the Euclidean invariant space. They are very consistent with respect to Euclidean transformations and can be used to distinguish different object shapes.

3.2 Indexing based on Maximum Length Invariant Patches/Parts

The distributions of vectors of algebraic invariants for maximum length invariant patches have the following important properties that lead to our choice of the indexing procedure:

1. The invariants of the patches extracted from an object do not form a compact cloud in the invariant space. This is a consequence of the fact that a complex object (one that cannot be represented well by a single 4th degree IP) contains patches that are significantly different from each other in shape and thus in the values of their invariants. Thus it is not possible to use classical recognition techniques that assume normal distributions.

2. The values of a vector of invariants from different objects can be similar since it is possible for these different objects to contain patches that are similar. Therefore, using Vector Quantization to get a set of representative invariant vectors for each object would require a large number of representative vectors to achieve acceptable performance.

Consequently one reasonable way to do indexing is to use a Nearest Neighbor Classifier where an invariant vector obtained from a single patch of the object to be classified is compared against all invariant vectors in the database. The object then can be assigned to that class which has an invariant vector that is closest in the sense of some distance to the observed vector. However, this requires that all patches from all objects be stored in the database and the comparison of the measured vector with all stored vectors which would result in an extremely slow classification procedure.

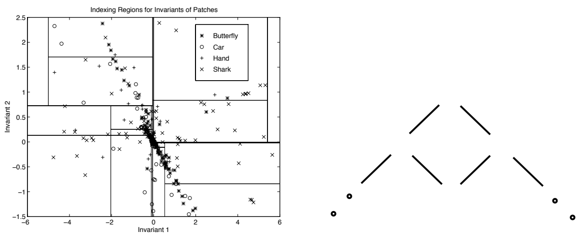
It is important to remember that at this stage we don't aim at classifying the given object, but only cutting down the possible candidates to a manageable number. These candidates would later be considered more carefully. The solution lies in using a relatively fast and coarse classification method, repeating the procedure for multiple patches from an object to achieve better accuracy.

Database preparation:

All the invariant vectors obtained from the standard pose of all the objects are pooled together. Given N, starting with the whole space of vectors of invariants as one region the space is iteratively split into smaller regions until no region contains more than N invariant vectors. Fig. 5 a shows the resulting regions for an invariant space containing invariant vectors from four objects. The decision boundaries used in breaking up the regions are stored in a binary tree structure, see Fig. 5b.

Indexing:

Now, given an invariant vector we use the tree structure stored in the database to determine to which region it belongs. The maximum depth of the decision tree structure will be $\lceil \log_2(\frac{M \times K}{N}) \rceil$ where M is the number of objects in the database, K is the maximum number of patches extracted from any object and N is the maximum number of vectors of invariants allowed in a single region. In our experiments we used $K = 200$ and $N = 8, 16$ and 32. Three invariants are used. For a database with one million objects the maximum depth of the binary decision tree is 17.



(a) (b)

Figure 5: (a) Some of the regions for a database containing 4 shapes. (b) The regions can be described using a tree structure to break up the space into parts. $R(n, m)$ is the m'th region of level n. $Q(n, m)$ and $!Q(n, m)$ represent the two sides of the boundary splitting region $R(n, m)$ into $R(n + 1, x)$ and $R(n + 1, y)$.

Thus the search can easily be done in real time even for very large databases.

Once we have determined that the observed invariant vector belongs to region r, we want to know the probability that it came from object c. This is represented by the following conditional probability

$$P(c \mid r) = \frac{P(r \mid c)P(c)}{\sum_{c=1}^{M} P(r \mid c)P(c)} = \frac{P(r \mid c)}{\sum_{c=1}^{M} P(r \mid c)} \quad (5)$$

where $P(c)$ and $P(r)$ stand for $P(object = c)$ and $P(region = r)$ respectively. The last step follows from the assumption that all objects have equal apriori probabilities of occuring.

$P(r \mid c)$ values are computed and stored off-line.

We could use eqn. 5 for a single patch to do indexing; however, because of the coarseness of the classification procedure and other reasons, it is necessary to use multiple patches. We can use P patches that are extracted from the object to be classified. $T(c) = \frac{1}{P}\sum_{i=1}^{P} P(c \mid r_i)$ represents an average conditional probability and becomes more reliable as P is increased. In our experiment, we used $P = 10$. (The optimum test statistic would have been $[\prod_{i=1}^{P} P(r_i \mid c)]/\sum_{c=1}^{M}\prod_{i=1}^{P} P(r_i \mid c)$.) The running time of the indexing procedure is determined by the extraction of the patches, the time for actual indexing is negligible. Thus the time required for indexing is linear with the number of patches used. Finally, we choose those objects which have the highest $T(c)$ values as the outcome of this stage of indexing. Results of a preliminary test on 12 objects are shown in table 1.

	$N = 32$	$N = 16$	$N = 8$
Top pick	% 83.1	% 93.6	% 97.1
In Top 2 Picks	% 87.7	% 95.3	% 97.3
In Top 3 Picks	% 90.8	% 97.0	% 98.8

Table 1: Success rates of indexing. Percentages based on 12 objects,1000 samples each. 10 patches were used for indexing.

Acknowledgements This work was partially supported by NSF Grant #IRI-9224963.

References

[1] G. Taubin, F. Cukierman, S. Sullivan, J. Ponce, and D. Kriegman, "Parameterized families of polynomials for bounded algebraic curve and surface fitting," *IEEE PAMI*, March 1994.

[2] J. Subrahmonia, D. Cooper, and D. Keren, "Practical Reliable Bayesian Recognition of 2D and 3D Objects Using Implicit Polynomials and Algebraic Invariants," *IEEE PAMI*, pp. 505–519, May 1996.

[3] J. L. Mundy, A. Zisserman, and D. Forsyth, *Applications of invariance in computer vision.* Springer Lecture Notes in Computer Science Series, 825, Springer-Verlag, 1994.

[4] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by Image and Video Content: The QBIC System," *Computer*, pp. 23–32, September 1995.

[5] J. Smith and S. Chang, "VisualSEEK: a fully automated content-based image query system," in *Proceedings of ACM Multimedia*, pp. 87–98, 1996.

[6] Y. Chan, Z. Lei, and D. Lopresti, "A Feature-based Approach for Image Retrieval by Sketch," in *SPIE Storage and Retrieval for Image and Video Databases II*, (Texas), November 1997.

[7] Z. Lei, M. Blane, and D. Cooper, "3L fitting of higher degree implicit polynomials," in *Third IEEE Workshop on Applications of Computer Vision*, (Sarasota, FL), pp. 148–153, December 1996.

[8] Z. Lei, T. Tasdizen, and D. Cooper, "PIMs and Invariant Parts for Shape Recognition," LEMS Tech. Report 163, Division of Engineering, Brown University, 1997.

[9] K. Siddiqi and B. Kimia, "Parts of visual form: computational aspects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, May 1995.