

Invariant-Based Shape Retrieval in Pictorial Databases

Michael Kliot and Ehud Rivlin*

Computer Science Department, Technion—Israel Institute of Technology, Haifa 32000, Israel

Received July 30, 1997; accepted April 30, 1998

One of the strongest cues for retrieval of content information from images is shape. However, due to the wide range of transformations that an object might undergo, this is also the most difficult one to handle. It seems that shape retrieval is one of the major barriers nowadays to image databases being commonly used. We present an approach for shape retrieval from pictorial databases which is based on invariant features of the image. In particular we use a combination of semi-local multivalued invariant signatures and global features. Spatial relations and global properties are used to eliminate nonrelevant images before similarity is computed. The advantages of the proposed approach are its ability to handle images distorted by different viewpoint transformations, its ability to retrieve images even in situations in which part of the shape is missing (i.e., in case of occlusion or sketch-based queries), and its ability to support efficient indexing. We have implemented our approach in a heterogeneous database having a SQL-like user interface augmented with sketch-based queries. The system is built on top of a commercial database system and can be activated from the Web. We present experimental results demonstrating the effectiveness of the proposed approach. © 1998 Academic Press

1. INTRODUCTION

The area of pictorial databases has recently become a subject of extensive research. Large databases of images are used in different applications including multimedia systems, medical imaging systems, and documents systems. The most common form of retrieval is based on image content. For most of the applications, content is defined in terms of three basic components: color, shape, and texture. This paper concentrates on shape-based retrieval.

Traditional approaches to shape retrieval, while providing good accuracy and efficiency, still suffer from instability as a result of viewpoint transformations. This follows from the fact that the invariance of the exploited features is limited. Global characteristics that are commonly used, like the aspect ratio or circularity [1], are invariant, for example, only to similarity transformation. Approaches that are based on edge directions [2] or

elastic sketch matching [3] are usually stable to scaling and rotation in a restricted range, but they do not hold for full similarity and not for the more general affine transformation. The queries are usually limited to be example-based, where the input image is supposed to be a version of one of the database images, which may undergo scale, rotation, and translation (i.e., similarity transformation). However, sometimes the input image can be occluded, which makes global features like moments inapplicable for retrieval. In addition, the user might want to present an input image having only part of the shape information, i.e., having only the coarse image structure, omitting fine details. In some cases the user does not want to present an input image at all and wants instead to use a logical description of shape features, e.g., “Give me all the images having three circles and a rectangle with ellipse inside.”

We developed an approach for shape-based retrieval that aims to cope with the problems described above, mainly the restriction of the viewing transformation to similarity, and a requirement for a complete, unoccluded input image. We use features, invariant for a wide range of viewpoint transformations, including affine and perspective. Shape similarity is measured using semi-local multivalued invariant signatures that allow one to handle situations in which part of the shape information is missing (i.e., occlusion or sketch-based retrieval). To efficiently handle such cases we introduce a data structure, the *containment tree*, that exploits the topological structure of the image. This structure supports indexing mostly in sketch-based retrieval. In this paper we present a series of experiments done on a heterogeneous database system which we built to test our approach. The system allows us to query images by logical or shape descriptions (query by example) or by a combination of both. It has an interface to the Web, providing a convenient user interface. We give a brief description of its data model in Appendix A.

The paper is organized as follows. In Section 2 we give some background on invariants and cover some related works. In Sections 3 and 4 we present our approach for invariant-based retrieval. Section 3 presents the machinery needed for indexing and retrieval with full and partial shape information, while Section 4 describes the query processing itself. Section 5 presents a variety of results using our approach for retrieving images from four different databases. Section 6 contains the concluding remarks.

* To whom correspondence should be addressed.

2. BACKGROUND AND RELATED WORK

Geometric Invariants

The subject of viewpoint invariants in vision has developed rapidly in recent years. A simple projective, or viewpoint, invariant, namely the cross ratio of four points on a line, was introduced in vision by Duda and Hart [4]. However, its domain of applicability was very limited. More general invariants were studied in the nineteenth century and were introduced in the field of computer vision. There are two main types of invariants: algebraic invariants and differential invariants. Algebraic invariants are based on a global description of the shapes by algebraic entities such as lines, conics, and polynomials. Differential invariants are based on describing the shape by arbitrary differentiable functions. These methods have been applied to various vision problems. The algebraic approach was used for example by [5] and [6], while the use of differential invariants can be found in [7] and [8]. Both methods proved to have advantages and disadvantages. The algebraic method, while simple and easy to implement, is quite limited in the kinds of shapes that it can handle because most shapes are not representable by simple low-order polynomials. The differential method is more general because it can handle arbitrary curves, but it relies on the use of local information such as derivatives (of quite high orders).

This situation has led to the introduction of various kinds of intermediate, or hybrid, methods that try to combine the advantages of the algebraic and differential methods. References [9], [10], and others introduced invariants that contain both derivatives and reference points. Each reference point reduces the number of derivatives that one needs in order to obtain invariants. In [11] a “canonical” coordinate system without curve parameterization is used to obtain the same goal. This results in fewer derivatives and in the capability of using feature lines in addition to points. However, in all these methods, the correspondence must be established between the reference points of the two images that are being matched.

In this paper we reduce the number of derivatives by using a scale space approach [12]. The scale space has to be invariant so we cannot use simple Gaussian-like smoothing. Instead, we rely on some reference points as a function of the given curve and a variable scale parameter. These reference points are not assumed to be readily available in the image, as in previous methods [13, 14], but are determined from the curve in an invariant way. Thus, no correspondence is needed. Using low-order derivatives and our variable reference points, we build invariant scale space representations of the given curves.

Content-Based Retrieval of Images

Jain and Vailaya [2] broadly classified work in the field into two categories on the basis of the approach used for extracting image attributes. Spatial information preserving methods derive features that preserve the spatial information in the image and it

is possible to reconstruct the images on the basis of their feature sets. Representative techniques include polygonal approximation of the object of interest [15], physics-based modeling, and principal component analysis [16]. The nonspatial information preserving methods extract statistical features that are used to discriminate among objects of interest. These include various feature-vector-based approaches, such as histograms and invariant moments [2, 17]. Both of these categories extract features based on cues, such as color and shape, that may be present in images.

Several works use *global numerical attributes* for indexing and retrieval of shapes. Those features are usually invariant only under Euclidean or similarity transformations, limiting the system. In QBIC (query by image content) [17] the shape features are based on a combination of global features: area, circularity, eccentricity, major axis orientation, and a set of algebraic moments. The assumption is that the shapes are unoccluded. The allowed shape queries are based on an example image. Chang and Smith [1] focus on automatic extraction of low-level visual features such as texture, color, and shape, especially in compressed form. In determining the similarity between different shapes, they use higher-level attributes, such as area, orientation, and aspect ratio, as well as intermediate-level representations, such as Fourier descriptors and chain code. In the work of this group, the input image is generally allowed to undergo transformations up to similarity, and occlusion is not handled.

Another form of shape description is based on *feature points*, detected from the image. Pentland *et al.* [16] present a shape model that is based on “interconnectedness” of shape features, e.g., edges, corners, or high-curvature points. Mehrotra and Gary [15] use for shape representation a collection of few adjacent interest points, like maximum local curvature boundary points or vertices of the shape boundary’s polygonal approximation. Eakins *et al.* [18, 19] use a boundary description that is based on features such as straight line segments, circular arc segments, and discontinuities. Their work can be distinguished from others by the fact that in addition to global features, local boundary features are exploited. However, the retrieval is impossible when only part of the shape is available (i.e., under occlusion), and the handled transformations are limited to similarity.

Jain and Vailaya [2] use a histogram of the edge directions as a shape feature. The method is limited to similarity transformation and unoccluded input image. Del Bimbo and Pala [3] use physical models of deformations for elastic sketch-based retrieval.

Although content-based retrieval of images has been a subject of extensive research in the past few years, only a few commercial systems support such a retrieval. Illustra [20] provides a library for storing and managing image data. QBIC [17] offers content-based image query in conjunction with standard search. In both cases image similarity retrieval is based on similarity of color and texture between the query image and the database images. However, the results are highly subjective and there is no intuitive metrics that can be used to decide whether the result images are in fact those that are most similar to the query image.

Several prototype research image database management systems have been reported in recent years that address the issue of how to index images in which the objects have already been recognized and tagged with their semantic meaning in order to support retrieval by image similarity [21, 22]. These systems are mainly concerned with spatial–relational information and do not deal with spatial–locational information. The most common data structure that is used is the 2-D string. Another data structure called the spatial orientation graph is used for spatial-similarity-based retrieval of symbolic images. In [23] a method that handles queries that deal with both spatial–relational and spatial–locational data is introduced. The method can deal with the distance between objects. In addition, as part of the pictorial specification, the user indicates the degree of desired similarity, and thus the results are not subjective. The method checks for contextual similarity by checking if the symbols in one image appear in the other image, where a symbol is a group of connected pixels that together have some common semantic meaning. In contrast, in our work the basic element is a curve.

Our work attempts to overcome three main limitations of the existing approaches: a restriction of the allowed viewing transformation to similarity, a requirement for a whole unoccluded input image, and a restriction of the queries to be example-based only.

3. INDEXING AND INVARIANTS

The purpose of indexing is to support fast retrieval from the database. Usually, two requirements are imposed on features used for indexing: the features should allow for substantial reduction in the number of candidates from the database, and a feature-based distance function should exist which supports ranking of images according to their distance from the input image.

As primitives for shape description we use various geometric entities. We exploit these entities for filtering the database while searching for a candidate set of images which answer a query. Features on which the description is based include various properties varying from the number of the different geometric entities, their dimensions and positions, to some topological relationships.

Our approach emphasizes the use of *geometric invariants* which provide a good mean for indexing while supporting ranking. Different features are used for extraction of descriptors that are invariant under various transformations.

Note that different applications may allow for different kinds of invariants. In applications that use images of trademarks, for example, similarity invariants are usually enough. For some other applications, however, a wider set of transformations, affine or projective, may be required. The applicability of a certain feature for indexing can be adjusted as well to the class of transformations derived from the application at hand.

Our invariant representation of objects is based mainly on *semi-local multivalued invariant signatures*. These signatures

provide a measure of distance between two curves, which allows us to rank images according to their distance from the input image. Below we describe the extraction and matching of such invariant signatures.

3.1. Semi-local Multivalued Invariant Signatures

Invariant signature is a function of a curve, calculated pointwise, and invariant for a given set of transformations. Invariant signatures can be used for recognition of planar curves. The important property of invariant signatures is their applicability in situations where the input curve is partially occluded. Having an invariant signature, curve matching reduces to matching signatures.

In order to calculate an invariant signature, one should find a local invariant for the given set of transformations and apply it at each point of the curve. Usually, local invariants are based on derivatives of the curve with respect to some parameterization. Several approaches try to combine the numerical stability of global invariants with the applicability of local invariants for cases of occlusion. Our approach follows that presented in [12, 24, 25]. First, the curve (object boundary), given in arbitrary parameterization, is reparameterized invariantly, using the lowest possible order of derivatives. Since we want to build the signature reflecting the global invariant geometric features in the small neighborhood of each point, we have to have the invariant measure of closeness of the contour points. It is important to note that the reparameterization process should be *local*, in order to allow us to deal with occluded images.

Formally, for a given planar shape transformation set $\mathbf{T}_\psi : \mathbf{R}^2 \rightarrow \mathbf{R}^2$, and for a given curve $\mathbf{P}(t)$, having an arbitrary parameterization, the target is to find an *invariant reparameterization* $\tilde{\mathbf{P}}(\tilde{t})$ so that if $\mathbf{P}(t)$ and $\tilde{\mathbf{P}}(\tilde{t})$ are related via $\tilde{\mathbf{P}}(\tilde{t}) = \mathbf{T}_\psi[\mathbf{P}(t(\tilde{t}))]$ then $\tilde{\mathbf{P}}(\tilde{t}) = \mathbf{T}_\psi[\mathbf{P}(t + \tau_0)]$. In other words, the reparameterization is invariant if the corresponding points have the corresponding parameter value, up to some constant cyclic shift.

The equations for invariant reparameterization are obtained using the differential properties of the viewpoint transformations. Let's define $K^{n,m}[x, y | t] \triangleq x^{(n)}y^{(m)} - x^{(m)}y^{(n)}$. The similarity transformation is parameterized using four parameters: an angle of rotation ω , translation vector \mathbf{v} (two components), and a scale factor $\alpha > 0$. For a given point (vector) \mathbf{u} , the transformation $\mathbf{T}_\psi : \mathbf{R}^2 \rightarrow \mathbf{R}^2$ is $\mathbf{T}_\psi(\mathbf{u}) = \tilde{\mathbf{u}} = \alpha \mathbf{U}_\omega \mathbf{u} + \mathbf{v}$, where \mathbf{U}_ω is a rotation matrix

$$\mathbf{U}_\omega = \begin{bmatrix} \cos \omega & -\sin \omega \\ \sin \omega & \cos \omega \end{bmatrix}.$$

For the similarity transformation we obtain that

$$d\tau = \frac{K^{1,2}[x, y | t] dt}{|\mathbf{P}'|^2}$$

is an invariant, generalized arc length, reparameterization.

Affine transformation is parameterized in the following form: $\mathbf{T}_\psi(\mathbf{u}) = \tilde{\mathbf{u}} = \mathbf{A}\mathbf{u} + \mathbf{v}$, where \mathbf{A} is a general nonsingular

($\det \mathbf{A} \neq 0$) 2×2 matrix. For the affine transformation we obtain that for $d\tau^* = |K^{1,2}[x, y | t]|^{1/3} dt$, $d\tilde{\tau}^* = |K^{1,2}[\tilde{x}, \tilde{y} | \tilde{t}]|^{1/3} d\tilde{t}$ we have $\tilde{\tau}^* = |\det \mathbf{A}|^{1/3} \tau^* + \tau_0^*$. This gives us linear scaled invariant reparameterization. In the particular case $|\det \mathbf{A}| = 1$, the reparameterization is absolutely invariant. Exploring further the properties of affine transformation, we obtain that

$$d\tau = \left| \frac{d}{d\tau^*} \right| \frac{K^{2,4}[x, y | \tau^*]}{K^{2,3}[x, y | \tau^*]^{3/2}} \left\| d\tau^* \right.$$

is a nonscaled affine invariant reparameterization.

The implementation of the reparameterization process intends to reduce the influence of the noise. First, the contour is transferred to a parametric form by B-spline approximation. Since we want the spline value at a certain point to depend on the coordinates of the contour points in a sufficiently large region, so that the influence of a small error in the edge detection is diminished, we adopted the technique used in [26]. The solution subsamples the contour, computes the spline values, and averages the results for different positions of samples on the contour. So, the first sampling set consists of each l th point on the contour, the points next to the chosen ones form the second set, etc. Thus, no information is lost, because each point participates in some sample. After the B-spline representation of the curve is obtained, the reparameterization is straightforward.

The computation of the semi-local signature is based on geometric features that stay invariant under the viewpoint transformation. Under similarity transformation, ratios of lengths, ratios of areas, and angles stay invariant. Thus, we can use those invariants locally to generate signature functions of various types. After the invariant reparameterization is completed, the two curves $\mathbf{P}(\tau)$ and $\tilde{\mathbf{P}}(\tilde{\tau})$ are related by $\tilde{\mathbf{P}}(\tilde{\tau}) = \mathbf{T}_\psi[\mathbf{P}(\tilde{\tau} + \tilde{\tau}_0)]$. This equation shows that one can compute the ratio of lengths, or the angle between, the segments defined by $(\mathbf{P}(\tau - s_B), \mathbf{P}(\tau))$ and $(\mathbf{P}(\tau), \mathbf{P}(\tau + s_F))$ for *a priori* chosen values s_B and s_F (locality parameters). Thus

$$\frac{\delta[\mathbf{P}(\tau), \mathbf{P}(\tau + s_F)]}{\delta[\mathbf{P}(\tau), \mathbf{P}(\tau - s_B)]}$$

and the angle formed by the points as functions of τ are invariant signature functions (see Fig. 1(left)).

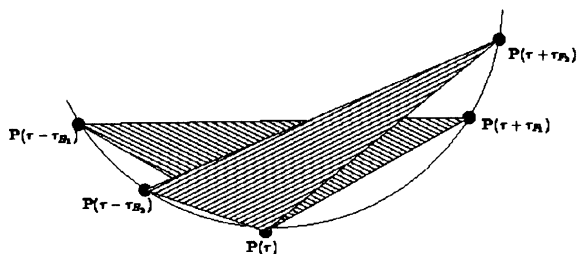
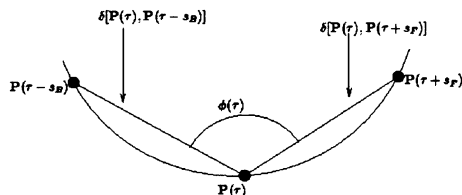


FIG. 1. Semi-local invariants. On the left, invariants for similarity transformation. On the right, invariants for the affine transformation.

TABLE 1
Local Invariants—Summary

Desired invariance	Local measure	Locality per sample
Euclidean	length	2
Similarity	angle, lengths ratio	3
Affine	areas ratio	5
Projective	cross-ratio	8

Under affine transformation, areas are uniformly scaled by $\det \mathbf{A}$. This fact implies that the ratio of areas is affine invariant. Assuming that an invariant parameterization was already obtained for the curve, we can choose four values for τ , τ_{B_1} , τ_{B_2} , τ_{F_1} , and τ_{F_2} , and calculate at each point $\mathbf{P}(\tau)$ the ratio of areas,

$$\frac{Area_{\Delta}(\mathbf{P}_{B_1}, \mathbf{P}(\tau), \mathbf{P}_{F_1})}{Area_{\Delta}(\mathbf{P}_{B_2}, \mathbf{P}(\tau), \mathbf{P}_{F_2})},$$

where \mathbf{P}_{B_1} , \mathbf{P}_{F_1} , \mathbf{P}_{B_2} , and \mathbf{P}_{F_2} are defined using locality parameters as for similarity case. This quantity is an invariant signature as a function of the invariant “arc length” τ (see Fig. 1(right)).

Table 1 summarizes the various invariant signatures we use. The table describes for each transformation the local geometric invariant that is used for building the signature and the number of localities needed per sample. One can see that as the desired invariance becomes more general, the number of points involved in the invariant calculation grows.

If the locality parameter set is allowed to be free parameters rather than setting them in advance, we obtain a whole range of invariants at each point rather than a single value (Bruckstein *et al.* [12]). The signature functions for curves become signature vectors or even continuum of values, i.e., surfaces or hypersurfaces. Matching them is less sensitive to peculiarities that may exist at some fixed pre-set value of the locality parameters.

It is important to note that since, in the general case, there is no correspondence between the initial points of the curves, the corresponding points on the curves have, after the invariant reparameterization, the same parameter value up to unknown constant cyclic shift value. This fact adds to the complexity of the matching process.

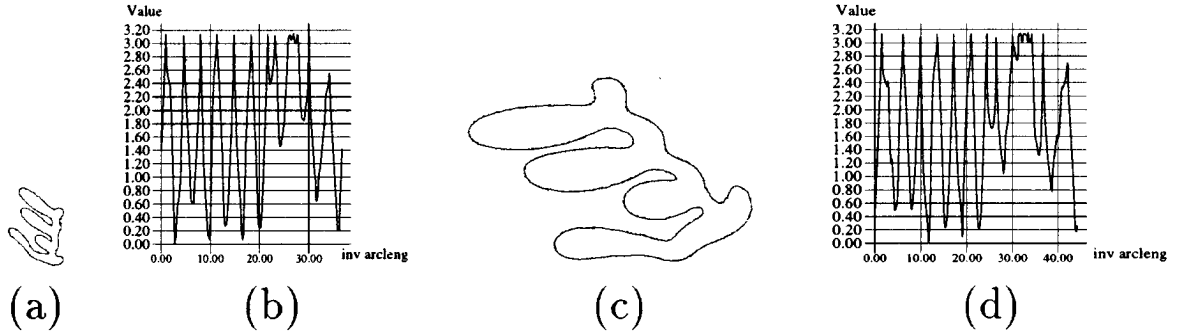


FIG. 2. Reparameterization inaccuracy. (b) and (d) Present graphs of invariant signature value versus invariant arclength. (a) and (b) Library image and its invariant signature. (c) and (d) The transformed library image and its invariant signature. Note the difference of the domains of (b) and (d).

We designed an effective automatic matching procedure, which can handle images distorted by affine and similarity transformations, and situations in which image is partially occluded. To match two signatures, we map them to the matrices with the number of rows equal to the number of signatures in the multivalued signature, and the number of columns equal to the number of the samples. In order to estimate the unknown value of relative cyclic shift, we exploit reference points. As a reference point we use the signature extrema. Minimizing the difference between the matrices over all checked shift values provides a measure for the distance between the signatures.

Values of the invariant perimeters of the same curves, up to transformation, may differ due to possible error in the edge detection process (see Fig. 2). We map the signatures having different invariant perimeters to matrices with the same number of columns and calculate the difference, while taking the difference between the invariant perimeters into account. The differences between two multivalued signatures is given by

$$\begin{aligned} & \text{Diff}(\mathbf{S}_{imp_0..Num-1}, \mathbf{S}_{lib_0..Num-1}) \\ &= \min_{sh \in \mathbf{Sh}} \frac{\sum_{i=0}^{Lev-1} \sum_{j=0}^{Num-1} (M_{inp}[i][\tilde{j}] - M_{lib}[i][j])^2}{Lev \times Num} \\ & \times \left(\frac{\max(\mathbf{p1}, \mathbf{p2})}{\min(\mathbf{p1}, \mathbf{p2})} \right)^p, \end{aligned}$$

where \mathbf{Num} is the number of signatures in the multivalued signature, \mathbf{Lev} is the number of samples, $\mathbf{s1}$ and $\mathbf{s2}$ are the compared signatures, $\tilde{j} = (j + sh) \bmod Lev$, \mathbf{Sh} is the set of all the checked

shift values, $\mathbf{p1}$ and $\mathbf{p2}$ are invariant perimeters, and \mathbf{p} is a positive constant.

In order to speed up the matching process, we exploit the signature histogram. This feature is independent on the starting point. The low-dimensional histogram allows for using of efficient data structures, like R-trees, for the database organization. First, signature histograms are compared. Only items close enough to the input curve are matched using the whole signature.

The total difference between the images is taken to be the average of the distances between the corresponding curves,

$$d(\mathbf{im}_1, \mathbf{im}_2) = \frac{\sum_{\{\text{Maximal curves}\}} \sqrt{\text{diff}(c_{1i}, c_{2i})}}{\mathbf{N}},$$

where \mathbf{N} is the number of compared curves and \mathbf{c}_{1i} and \mathbf{c}_{2i} are corresponding curves of \mathbf{im}_1 and \mathbf{im}_2 , respectively. In order to find pairs of corresponding curves we sort the curves of each image with respect to some invariant criterion. In case of similarity transformation, the curve's area serves as an ordering criterion. Curves that cannot be sorted by area are ordered by their perimeters. Under affine transformation, perimeters cannot be used any longer; however, the ratio of areas is still invariant and areas can be used for ordering.

The following example illustrates extraction and matching of invariant signatures. The input image (Fig. 3a) is an affine transformed version of the library image (Fig. 3b). The extracted contours are presented on Fig. 3c. The contours were extracted by a tracing and collection process which runs on the results of Difference Recursive Filter [27–29]. The contours were

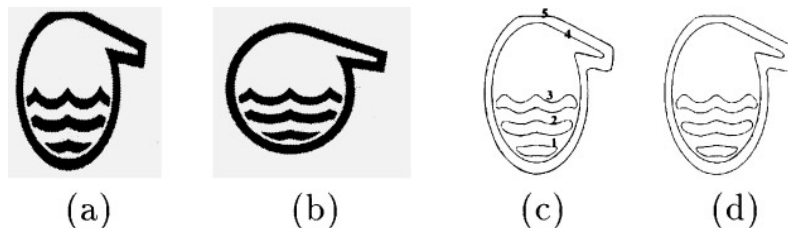


FIG. 3. (a) The input image. (b) The source library image. (c) The contours of the input image (numbered). (d) The fitted contours.

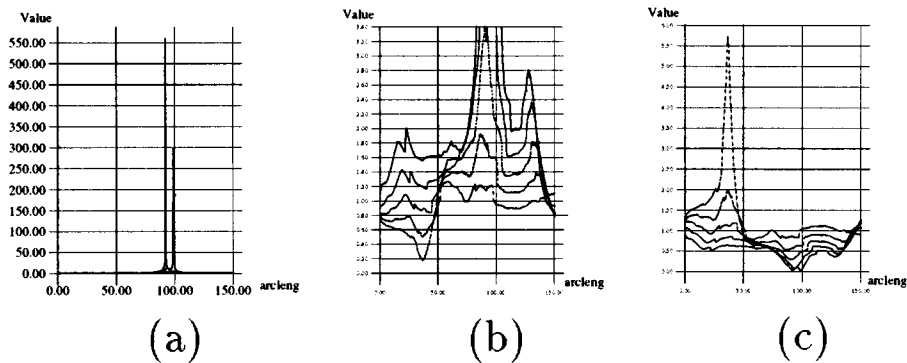


FIG. 4. Numerical instability of an affine invariant signature. (a) Multivalued affine invariant signature $f(t)$ for curve No. 5 from Fig. 3. The signature is based on the areas ratio. (b) $f(t)$ after putting a threshold. (c) The signature $1/f(t)$.

ordered (by area) and invariant signatures were extracted. We use the invariant signature described above—the ratio of areas, with four parameters, each having five sample points. While for similarity invariants we used the angle, which is relatively stable characteristic, the ratio of areas can suffer from numerical instability. If the points $\mathbf{P}(\tau - \tau_{B_2})$, $\mathbf{P}(\tau)$, and $\mathbf{P}(\tau + \tau_{F_2})$ (see Fig. 1(right)) become near collinear, the signature becomes unstable. Figure 4 presents the affine invariant signature for curve No. 5 from Fig. 3. The instability points force us to put a threshold on the signature values. While the resulting signature (see Fig. 4b) can still be used for the matching process, some of the information apparently is lost. Another possibility is to exchange the values of τ_{B_1} with τ_{B_2} , and τ_{F_1} with τ_{F_2} , respectively, and use the signature $\frac{1}{f(t)}$ instead of $f(t)$ (Fig. 4c). The matching results for the trademarks presented in Fig. 3 are presented in Fig. 5. Each grey-level map, consisting of five strips, corresponds to the multivalued signature, while each strip corresponds to the singular signature. Each pair of the grey-level maps presents the signatures of the input (above) and library (below) contours at the shift position giving the best match (the numbering of the contours is as in Fig. 3c). Good match is achieved for all the curves. Note, that the signatures of curves No. 4 and 5 are very similar as the curves are almost identical when going under affine transformation.

3.2. Indexing while Some Shape Information Is Missing

One of the main advantages of the proposed approach is its ability to handle situations in which part of shape information

is missing. We handle separately two different cases: partial occlusion and user-generated sketch-based queries. In both cases the indexing is based on the same geometric features which are used regularly. However, the indexing algorithms will be used in accordance with the appropriate context.

3.2.1. Partial occlusion. Partially occluded images are treated in the same way as regular images. Since both the reparameterization and signature extraction stages are absolutely local, the signature values for the part of the contour should match the values computed for the whole curve. The only difference is the need for *partial matching*, i.e., matching of the occluded contour to the part of the library curve. If the input curve is occluded, we map it to a matrix with the number of columns proportional to the width of the signature domain. The domain of the multivalued signature is the minimal domain of its components (for open curve, the larger the locality parameter, the smaller the signature domain). Thus, the number of columns in the matrix is

$$\widetilde{\text{Lev}} = \text{Lev} * \min_{i=0}^{\text{Num}-1} \{ \|\text{domain}(S_i)\| \} / \text{Per}_{\text{lib}},$$

where Per_{lib} is the invariant perimeter of the library signature and Lev is the number of columns of the library matrix. The matrix of the occluded signature is then matched to a part of the library matrix.

Figure 6a is a rotated, scaled, and occluded version of the library image presented in Fig. 6b. We use the angle $\phi(\tau)$, formed by the points $\mathbf{P}(\tau - s_B)$, $\mathbf{P}(\tau)$ and $\mathbf{P}(\tau + s_F)$, as an invariant

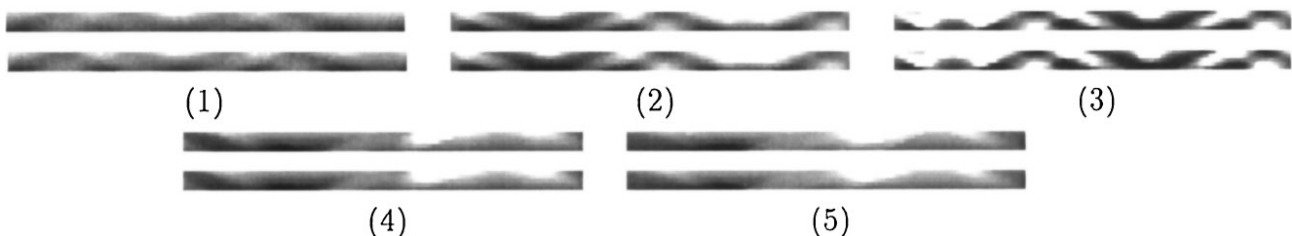


FIG. 5. The best match for each curve presented in Fig. 3. Grey-level maps are used for signatures. The curves are numbered as in Fig. 3. Good match is achieved for all the curves.

signature (as in Fig. 1(left)). Figure 7 presents the beast matchings between the input and the library curves. The signature domain is reduced as a result of the occlusion.

In the next example we took a real image, photographed under partial occlusion. The extracted trademark part of the image served as an input to the query against the trademarks database. The input image and the retrieval results are presented on Fig. 8.

3.2.2. Sketch-Based Queries. We consider sketch-based retrieval as retrieval that uses the gross shape structure, while the fine details can be omitted. For sketch-based retrieval we exploit a topological invariant which we define here that is based on the following simple and effective invariant property:

Given the same contour decomposition for an image P and a transformed image $\mathbf{T}_\psi(P)$, the contour C_1 of P resides inside the contour C_2 if and only if the same relation holds for the corresponding contours in the transformed image. This holds for any projective transformation \mathbf{T}_ψ .

In other words, the property that one contour is an inner contour of another is a projective invariant given the same contour representation. This property allows us to represent images exploiting the relations of internal–external between contours. Each image is represented as a tree which we term the *containment tree* (This definition is similar to the adjacency graph [33]). The vertices of the tree are curves. For two curves, C_1 and C_2 , the edge ($C_1 \rightarrow C_2$) exists if C_2 is inside C_1 , and there is no C_3 such that C_2 is inside C_3 and C_3 is inside C_1 . The root of the tree is a “dummy” contour which includes all the contours. This representation can easily be obtained after curves extraction (see Fig. 9). The representation is unique up to the order between the vertices on the same level. Thus tree-matching algorithms can be used to compare the representation of the input image with the library images. This problem of tree matching has a polynomial solution [34]. The discrimination power of the containment tree is illustrated on Fig. 10.

Algorithm 1

Sketch matching.

Given: An input sketch S , and a database image I ,

1. Build a containment tree T_1 for S , and a containment tree T_2 for I . In each containment tree the contours on the same level are ordered in descending order using the area – affine invariant geometric criterion.
2. Check if T_1 matches T_2 .

The match is defined in the following manner:

The containment tree T_1 for the sketch S matches the containment tree T_2 for the image I if:

1. T_1 is a leaf **or**
2. The number of subtrees of T_1 equals that of T_2 and the corresponding sons of T_1 and T_2 match.

We exploit the containment tree for sketch-based retrieval. The matching algorithm is presented as Algorithm 1. The algorithm requires that for each contour of the specified sketch all its sub-contours should be either omitted or specified. In this way

we allow for queries that uses gross structure while omitting details (from some level of the tree).

Algorithm 2

Flexible sketch matching.

Given: An input sketch S , and a database image I ,

1. Build T_1 and T_2 as in Alg. 1.
2. Check if T_1 flexibly matches T_2 .

We define a flexible match in the following manner:

The containment tree T_1 for the sketch S flexibly matches the containment tree T_2 for the image I if:

1. T_1 is a leaf **or**
2. The number of subtrees of T_1 equals that of T_2 and the corresponding sons of T_1 and T_2 flexibly match **or**
3. The number of subtrees of T_1 , n , is less than that of T_2 , but the $(n+1)^{th}$ son of T_2 corresponds to a contour which is much smaller than the contour corresponding to the n^{th} son of T_2 (that is, the ratio of their areas is less than some predefined threshold R), and the first n corresponding sons of T_1 and T_2 flexibly match.

A flexible version of this algorithm which allows us to omit small sub-contours of the specified sketch is presented as Algorithm 2. Changing a threshold R allows us to control the retrieval flexibility. As the flexibility threshold R gets larger, more candidates will pass the threshold and will match the input sketch, giving higher retrieval time. Setting R , therefore, presents a tradeoff between false alarms and misses. If the containment trees of the sketch and the image match according to the appropriate algorithm, the final verification is performed using multivalued signatures.

Figure 11 presents an example of sketch-based query. The lower row presents the distance values for the database items. One can see that the values confirm the intuitive measure of the similarity between the sketch and the images.

4. SHAPE RETRIEVAL

Our shape retrieval scheme consists of two main phases: filtering the database in order to drop the irrelevant images, and ranking the *candidates subset* according to the distance from the input image. For indexing we use geometric entities, such as circles and ellipses. In our system we detect geometric entities from image contours. Circle detection is based on the fact that the characteristic ratio, which equals the quotient of the area of the curve to the square of its perimeter, is known to be minimal

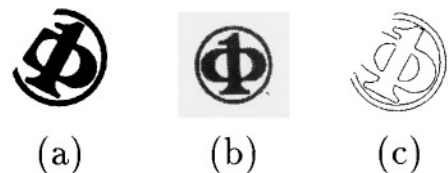


FIG. 6. (a) The input image (occluded). (b) The source library image. (c) The contours of the input image.

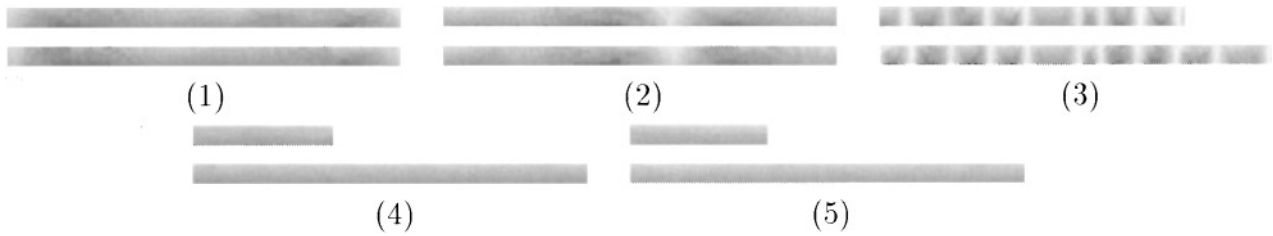


FIG. 7. Best matches for the curves presented in Fig. 6a. In (1) and (2) best matches for the signatures of the inner contours of the Φ letter are presented. In (3) the best match for the external contour of the Φ letter is presented. (4) The best match for the inner circle. (5) The best match for the external circle.

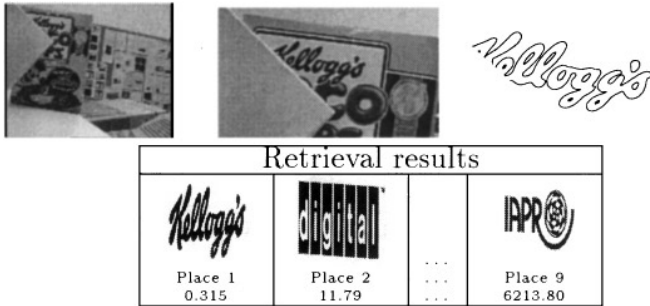


FIG. 8. Example of real occluded scene. (Above) The photographed scene, the region containing the trademark, and the extracted contours. (Below) Retrieval results.

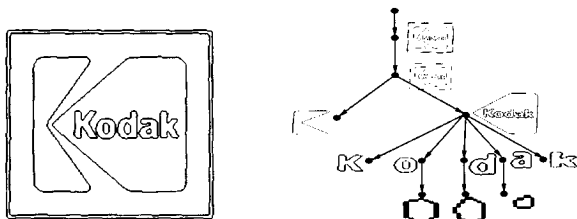


FIG. 9. The input image after curves extraction is presented on the left. The containment tree, representing the containment relationships between the curves, is presented on the right. For each subtree, the corresponding curve and all its internal curves are presented.

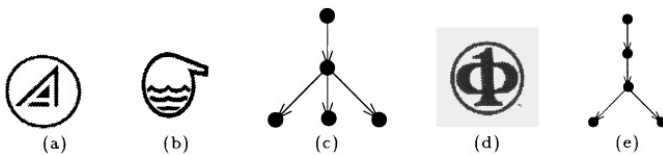


FIG. 10. The discrimination power of a containment tree. (a) and (b) Two images having the same containment tree. (c) The containment tree of (a). (d) The image has the same number of curves as in (a), but a different containment tree. (e) The containment tree of (d).

	0.598	0.604	11.00	15.386

FIG. 11. Sketch query example. The input sketch is on the left.

($1/4\pi$) for a circle. The curve is detected as a circle if its characteristic ratio is close to $1/4\pi$ up to a predefined threshold. The detection of ellipses is based on the fact that any ellipse can be transformed to a circle by appropriate affine transformation. So, if we calculate affine invariant signature for an ellipse, it should be equal to that of a circle, which should be constant since a circle is a fully symmetrical image. Thus, we can calculate the affine-invariant signature described in Section 3.1, setting its parameters to some predefined values, and check the difference between the obtained signature and the *a priori* known constant value. The curve is detected as an ellipse if the difference value is close to zero. The number of these entities can serve for efficient filtering of the image collection. Final ranking within the set is based mainly on the semi-local multivalued invariant signatures. The number of geometric entities in the image instances provides an accurate filtering. In order to avoid miss-matches, we allow the database image to have more entities (circles, ellipses) than the input image. In addition, the threshold used for entity detection in the input image is tighter than those used for database images. Thus, correct database images are not pruned out.

The general scheme of query processing is presented on Fig. 12. After edge detection and curve extraction from the input image, geometric entities are detected. This global feature extraction is used as a basis for the filtering process. Relational database (see Appendix A) is used in this stage in order to retrieve the candidates subset. Features used for indexing to the relational database and eventually for pruning the candidate set vary from the number of the curves and the geometric entities, their relative dimensions, etc. The relational database includes alphanumeric data as well. The query may contain this kind of information, like the organization name (for the trademarks database), and this part of the query will be processed as a regular query (further pruning the set). The candidates obtained at this stage go to a matching with the input image. The final set, ordered according to the similarity measure, is given as an answer.

4.1. Computational Considerations

In this section we discuss the computational complexity of feature extraction and present typical processing times. Table 2 presents average reparameterization time, average time spent for computing signatures, average depth of a containment tree, and average fan out. The quantities are averaged over groups of images having a certain number of curves. The number of

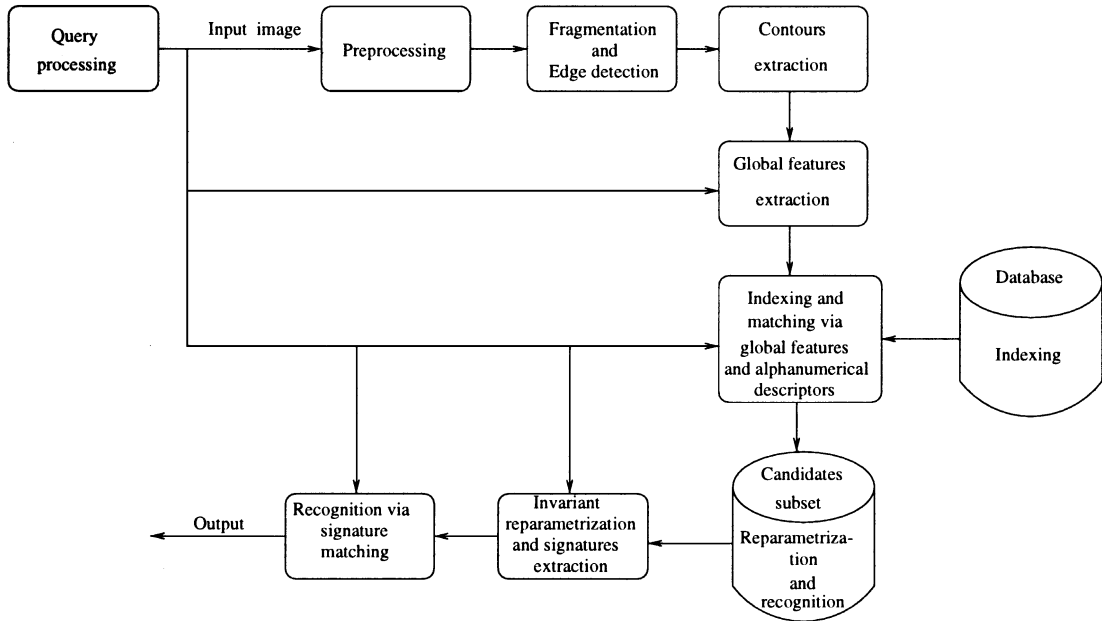


FIG. 12. Query processing.

TABLE 2
Computational Considerations

No. curves	Average reparameterization time	Average signature computation time	Average depth	Average fan out
1	2.60	0.12	1.00	0.00
2	2.74	0.14	1.38	1.00
3	2.90	0.19	1.60	1.50
4	5.74	0.24	1.50	2.00
5	3.43	0.27	2.25	1.93
6	4.76	0.32	2.43	1.50
7	6.10	0.39	2.17	2.67
8	5.71	0.39	2.57	2.75
9	4.89	0.43	2.40	2.56
10	5.25	0.46	2.33	2.23
11	10.98	0.49	2.17	3.33
12	8.40	0.59	4.00	2.62
13	8.07	0.64	1.50	2.67
14	6.09	0.52	2.33	3.00
15	7.05	0.65	2.75	4.67
16	6.20	0.55	2.67	3.67
18	6.28	0.53	3.00	2.50
19	7.76	0.75	3.00	2.50
20	9.72	0.88	2.46	4.16
Overall	5.89	0.44	2.26	2.86

Note. For the trademark database sorted by the number of curves, average reparameterization time, average time spent for computing signatures (both in seconds), average containment tree depth, and average fan out are presented. The quantities are averaged over all the images having the same number of curves. The last row presents the results averaged over the whole database.

nodes in the containment tree is equal to the number of curves in the image. Note that both the average depth and fan out are low, providing efficient containment-tree-based retrieval. Most of the computing time in the feature extraction process is spent on the reparameterization, which involves B-spline calculation. However, for the database images, the computation is done off-line (see Fig. 26, Appendix A). The only reparameterization that is done on-line is of the input image, and this is done only once.

The computational complexities for various stages in the process are as follows. The reparameterization complexity is $O(N \times ord^2)$, where N is the number of points over the curve and ord is the spline order (typically 6). The signature calculation complexity is $O(N * Num)$, where Num is the number of signatures in the multivalued signature. The containment tree matching algorithms (Algorithms 1 and 2) have $O(C)$ complexity, where C is the number of curves in the image.

5. EXPERIMENTS

To show the applicability of our approach we tested it first on a database of 500 constrained 3D objects, namely surfaces of revolution. Using the KBS bottles collection [32] we present queries showing good retrieving ability for a specific, and similar, objects. The effectiveness of the filtering process and the ability to handle different transformations are demonstrated in the second part in which a UMD [33] database of 100 trademarks is tested. We give results for sketch-based queries on a UMD [33] database of more than 300 “road signs.” To show the scalability of our approach, as well as retrieval results for a combination of various databases, we added a database of about

1100 contours of images of marine creatures used by SQUID system [34], and tested the system on combination of the four databases together. The section is concluded with performance evaluation. For all four databases we used the same system. One may note that our system performs best with structured domains such as trademarks and road signs images. The system is implemented using the ORACLE/SQL environment and has a WWW interface. The user is allowed to query images by example, logical description, or the combination of both. The example image can be the database image or the image provided by the user.

5.1. Invariant Signatures for Surfaces of Revolution

An interesting practical application field for multivalued signatures is recognition in a database of 3D objects which are surfaces of revolution (see also Mundy *et al.* [35]). We can treat such objects as planar under a controlled change of the view-point (which is common in industrial settings, e.g., in assembly lines). In this case, we can approximately describe the view-point change by affine transformation and apply our algorithms. In our experiment, the database contains more than 500 items some of which are photographed objects that are surfaces of revolution and the rest are bottles from the KBS bottles collection [32]. Given an input image (one of the database objects under rotation and zoom), we compared it with the database by using multivalued signatures. Figure 13 presents an input image, its external contour, and its multivalued invariant signature. This input image was checked against all the images in the database. A good match was achieved retrieving the right database image in the first place. Figure 14a presents the grey-level maps of the input (above) and the database (below) contours at the shift position giving the best match. Retrieval results are presented next (first four of the set). Figure 15 presents a number of queries performed using the surfaces of revolution database.

Figure 16a presents our HTML interface. The input scene and the processing results are presented in Fig. 16b.

5.2. Filtering the Candidates Set

Next we illustrate the effective filtering obtained by using invariant features. We run our queries on a database which contains

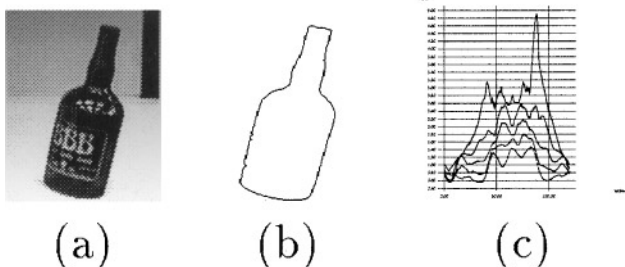


FIG. 13. (a) The input image. (b) The external contour of the input image. (c) The invariant signature for the input image.

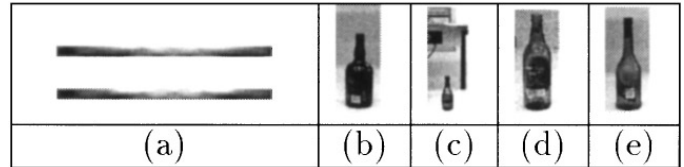


FIG. 14. (a) The best match for the input image. (b)–(e) Retrieval results.

more than 100 trademarks. In the first example (Fig. 17(1)), the input image is a rotated and scaled version of one of the database trademarks. The user operated with the query

**select title ordered by dist(logo30-trans.tiff)
where (context = text).**

The query looks for images having textual strings in them. Both alphanumerical information and geometric features are used for filtering the database. Exploiting geometric features is especially effective for images with a large number of curves, because of their complicated structure. One can see that the number of candidates is rather small. In the following example (Fig. 17(2)), the user operated with

**select title where (N_circles = 1 and N_curves
≥ N_curves (logo55.tiff)),**

directly specifying the number of geometric entities (circles). The input image is one of the database trademarks. One can see that the candidates comply with the given condition. The user has the freedom to limit the transformation which the images are allowed to undergo by stating the transformation explicitly in the query. In the next example (Fig. 17(3)), the user limited the transformations to the affine case, operating with the query

select title ordered by dist(logo50-transf.tiff) under affine.

Both the filtering and the ranking are based on features that are invariant under the affine transformation. The input image is an occluded version of one of the database trademarks.

The next example presents sketch-based retrieval (Fig. 17(4)). The sketch was drawn using curves-drawing software (Xfig). Here we allow the omission of some small objects within the image, while preserving its general structure. In this case we used Algorithm 2 for filtering the database. The various stages for creating the sketch are presented in Fig. 18.

Note that in all the examples the number of the images in the candidates set is sufficiently smaller than the number of the items in the database. The filtering is still effective even where occlusion is present (see, for example, Fig. 17(3)). When the filtering is based on the containment tree we expect it to be more effective (see, for example, Fig. 17(4), where only four candidates passed the filtering).



FIG. 15. Results of queries on the bottle database.

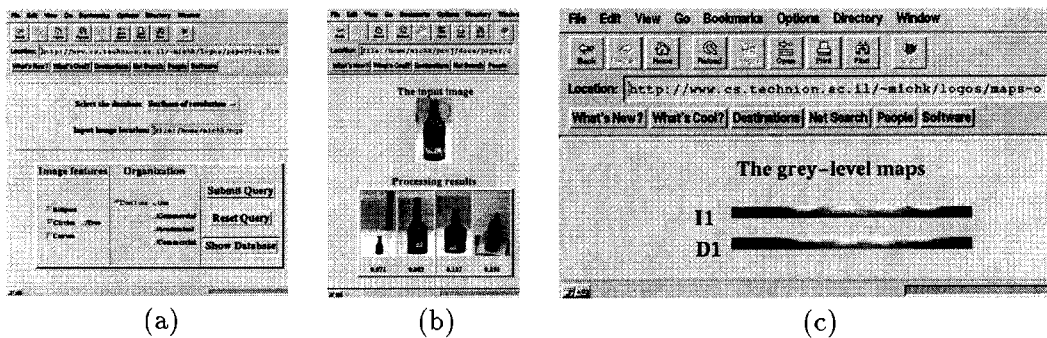


FIG. 16. (a) Querying the database using HTML interface. (b) The input image and processing results. (c) The grey-level maps, presenting multivalued signatures: above, the map corresponding to the input image; below, the map, corresponding to the image which took the first place.


















	Input image	Retrieval results			
1		 Place 1 0.128	 Place 2 9.57	...	 Place 5 1492.57
2		 Place 1	 Place 2	...	 Place 10
3		 Place 1 0.233	 Place 2 0.804	...	 Place 23 1347.148
4		 Place 1 0.526	 Place 2 0.919	 Place 3 0.961	 Place 4 0.961

FIG. 17. Query results.

5.3. Sketch-Based Queries

Figure 19 presents a number of sketch-based queries and retrieval results. In addition, we present the queries in SQL-like notation (see Appendix A). The queries are performed on the database of “road signs,” containing about 300 images. After filtering based on Algorithm 1 or 2, the candidates passed matching with the input sketch, based on invariant signatures. For each query, the resulting candidates set contains no more than 15 images. One can see that the distance measure reflects the similarity between the sketch and the candidate image.

In the next experiment we checked sensitivity of the sketch-based queries to deformations. Figure 20 (left) shows the behavior of the retrieval process when the input sketch (the same as in Fig. 17(4)) passes graduate deformation. One can see that the results are quite stable, and the first two places don't change. Figure 20 (right) shows how the error behaves as the input sketch deforms.

5.4. Experiments with a Combination of Different Databases

In these experiments we illustrate the scalability of our approach. For this purpose we added a database of about 1100 contours of images of marine creatures used by the SQUID system [33]. Figure 21 presents the results of a simple query on this database. Next we present an experiment of retrieval using all the images from our four different databases (a combination that includes more than 2000 images). Figure 22a presents the input image. Figures 22b–22e present retrieval results obtained against the union of the four databases.

Figure 23 presents an example of retrieval using two databases, namely the trademark and sign databases.

5.5. Efficiency Analysis

In this section we used the combined database which included both the database of the trademarks and the “road signs.”

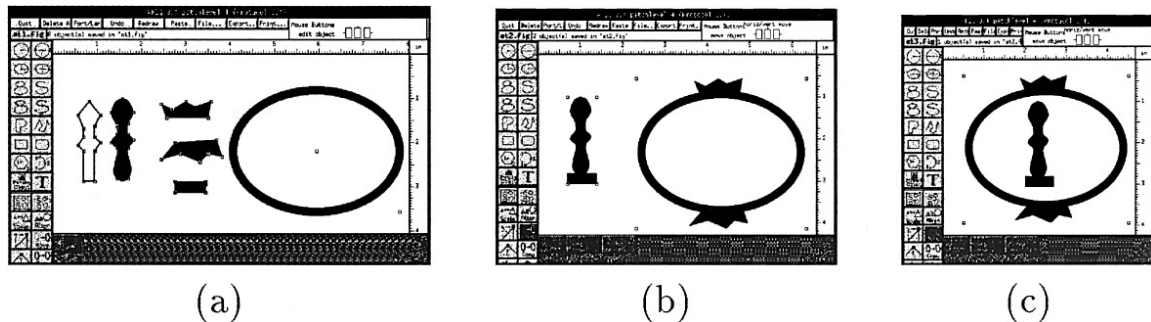


FIG. 18. Creating an input sketch for the query in Fig. 17(4) using Xfig. (a) Basic elements, including a control polygon for the central element. (b) Intermediate stage. (c) The final sketch.




























Input sketch	Retrieval results									
	 0.072	 0.251	 0.816	 6.08	 20.71	 22.19	 106.24	 928.02	select image sorted by sketch(093X) where (method = flexible)	
	 0.048	 0.321	 1.97	 2.52	 40.17	 85.09	 121.93	 442.10	select image sorted by sketch(093R)	
	 0.053	 0.821	 8.23	 10.01	 30.68	 35.12	 90.14	 112.65	select image sorted by sketch(097Q) where (method = flexible and flexibility=0.1)	

FIG. 19. Results of queries on the “road signs” database.

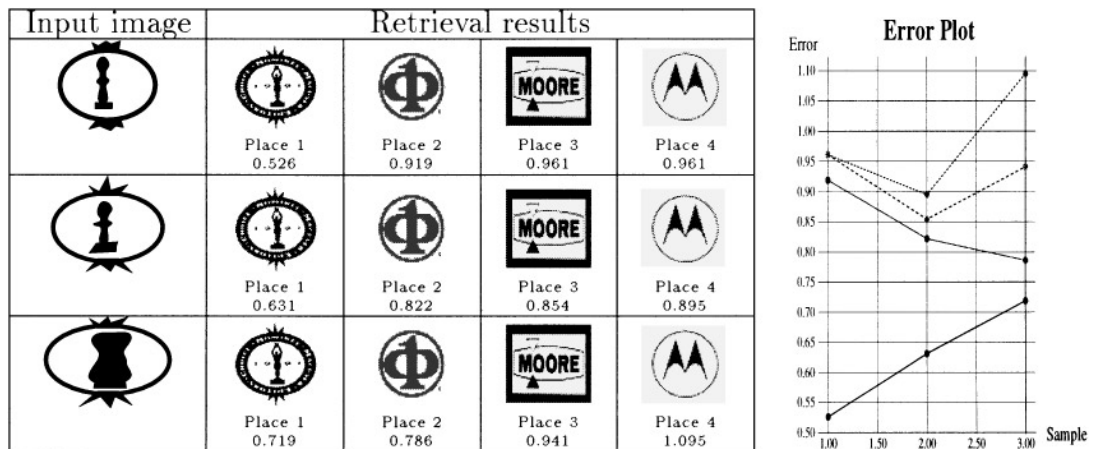


FIG. 20. Sensitivity of the matching to shape deformation. (Left) Retrieval results. (Right) The error plot. The lowest graph represents the error for the first place, the second graph represents the error for the second place, etc.



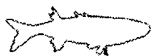


Input image	Retrieval results				
					
(a)	(b)	(c)	(d)	(e)	

FIG. 21. Results of a query on the marine creature database. (a) The input image. (b)–(e) Retrieval results.

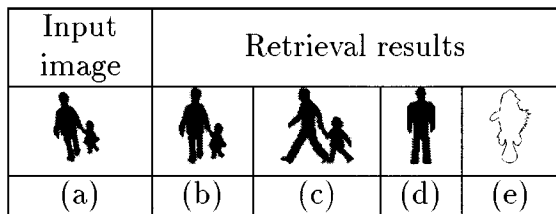


FIG. 22. Experiment with four databases. (a) Input image. (b)–(e) Retrieval results.

Figure 24a presents the average processing time (averaging on 100 queries) for a query as a function of the number of the curves in the input image. The upper curve corresponds to a sequential matching, i.e., a full signatures matching against the entries in the database. The middle presents average processing time for a query using the number of geometric entities for pruning the database. The lower curve corresponds to retrieval using the containment tree. The middle and lower curves are nonmonotonic, a direct consequence of a tradeoff between a shorter query execution time as a result of the pruning and the extra time needed for matching more signatures. The graphs demonstrate that geometric entities and especially the containment tree are effective means for database pruning and their efficiency grows with the number of contours.

Figure 24b demonstrates the scalability of our approach. Queries have been run to retrieve an image from a subset of the database. The queries used the number of geometric entities for pruning the subset. When the subset grows, processing time increases approximately as a linear function.

Table 3 illustrates the influence of the flexibility threshold R (see Algorithm 2) on the number of retrieved images for sketch-based retrieval. The number of the answers depends on the complexity of the input sketch.

6. CONCLUSIONS

In this paper we have addressed the problem of shape-based retrieval from image databases. Our approach emphasizes the use of invariants as shape descriptors. Specifically, we have used geometric invariant features for efficient indexing, while local

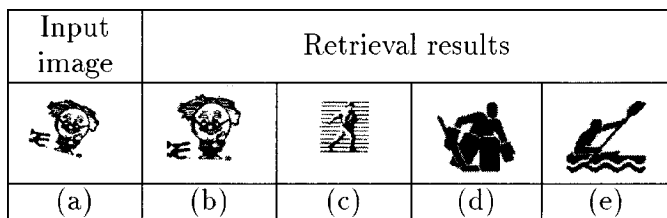


FIG. 23. Experiment with trademark and sign databases. (a) Input image. (b)–(e) Retrieval results.

TABLE 3
The Number of Retrieved Images as a Function of the Flexibility Threshold R

R	XI	☒	🚗
0.01	2	4	10
0.02	3	6	12
0.04	6	15	18
0.08	10	16	18
0.16	14	25	20

Note. For the presented input sketch, the number of retrieved images is given for different values of R .

multivalued invariant signatures have been used for ranking the answers. The substantial reduction of the candidates set due to the filtering stage guarantees the efficient retrieval. The approach supports image retrieval while part of the shape is missing, can handle images distorted by different viewpoint transformations, and can flexibly answers queries based on logical descriptions, shape (query by example), or combination of both.

We have implemented our approach in a heterogeneous database system having a SQL-like user interface augmented with sketch-based queries. The system is built on top of a commercial database system (Oracle) and can be activated from the Web. We have presented experimental results demonstrating the effectiveness of the proposed approach under various conditions using three different databases.

APPENDIX A: THE DATABASE

We implemented the database as an *object-oriented*, in accordance with the definition in [36]. A database entry has a *complex structure*, a direct result of the image features we use. An *image* data type is a complex object, which includes, in a nested structure, different data types. Some of these objects are themselves complex, like a *containment tree* or a *curve*. Each database entry includes, in addition to the geometric features,

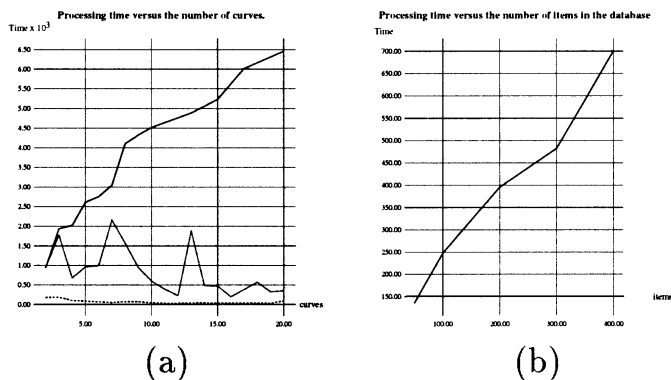


FIG. 24. (a) Average evaluation time versus a number of CURVES. (b) Scalability: Processing time versus the number of database items.

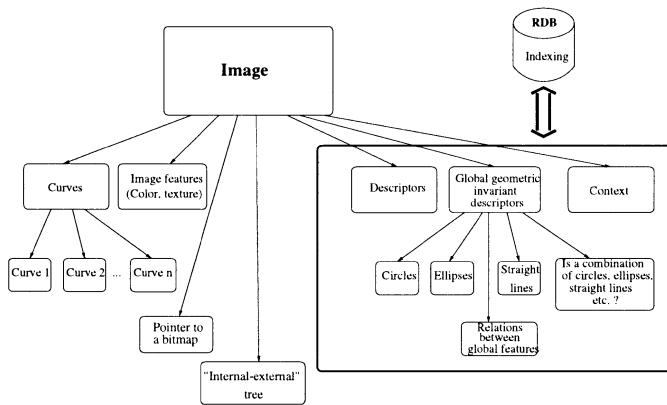


FIG. 25. The image data model.

some alphanumerical information. This information allows for textual-based querying, in addition to, or in combination with, content-based retrieval. The alphanumerical data and the low-dimensional geometric descriptors reside in a relational database (RDB, in our system Oracle). The RDB covers part of the object-oriented database (OODB) and gives efficient performance and convenient user interface.

The data model is presented in Fig. 25. The model contains three types of fields:

1. Low-dimensional geometric descriptors and alphanumerical fields:

—*Descriptors* fields store alphanumerical descriptors of the image that the user assigns when the image is inserted (i.e., data on a company, etc.).

—*Context* can include a verbal description of the image or its parts (“bird,” “apple” etc.), characters appearing in the image (“Kodak” in Fig. 9) etc.

—*Global geometric invariant descriptors*.

2. High-dimensional geometric and nonshape features:

—*Image features*: color, texture, etc.

—*Bitmap* field: pointer to a file.

—*Containment tree*.

3. Curves:

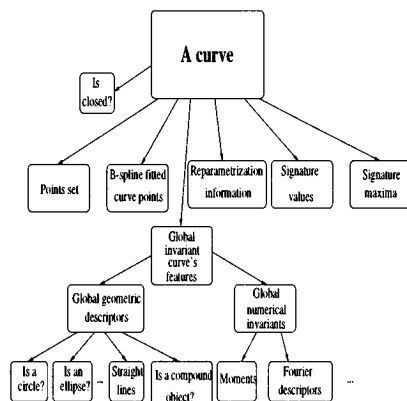


FIG. 26. The curve data model.

Each curve is a complex object represented according to the curve data model. For each curve the data model includes (see Fig. 26):

- an ordered set of points,
- whether the curve is closed or not,
- the points of the B-spline which was fitted to the curve,
- the information about the invariant reparameterization (i.e., the values of the invariant parameter for each point),
- the invariant signature values—for each point, *Num* values are stored, where *Num* is a number of signatures (usually 1, higher for the multivalued case),
- auxiliary information, i.e., signature maxima, global curve features, etc.

REFERENCES

1. S.-F. Chang and J. Smith, Extracting multi-dimensional signal features for content-based visual query, in *SPIE Int. Symp. on Visual Communications and Image Processing*, 1995.
2. A. Jain and A. Vailaya, Image retrieval using color and shape, *Pattern Recognition* **29**, 1996, 1233–1244.
3. A. D. Bimbo and P. Pala, Visual image retrieval by elastic matching of user sketches, *IEEE Trans. Patt. Anal. Mach. Intell.* **19**, 1997, 121–132.
4. R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
5. D. Forsyth, J. Munday, A. Zisserman, and C. Brown, Projectively invariant representations using implicit algebraic curves, *Image Vision Comput.* **8**, 1990, 130–136.
6. G. Taubin and D. Copper, Object recognition based on moment (or algebraic) invariants, in *Geometric Invariance in Machine Vision* (J. Mundy and A. Zisserman, Eds.), pp. 375–397. Cambridge, MA, MIT Press, 1992.
7. O. Weiss, Projective invariants of shapes, in *Proc. DARPA Image Understanding Workshop*, Cambridge, MA, 1988, pp. 1125–1134.
8. A. Bruckstein and A. Netravali, On differential invariants of planar curves and the recognition of partially occluded planar shapes, Tech. Rep., AT&T Technical Memo, 1990.
9. L. V. Gool, P. K. A., and Oosterlinck, Recognition and semi-differential invariants, in *Proc. CVPR, 1991*, pp. 454–460.
10. E. Barrett, P. Payton, N. Haag, and M. Brill, General methods for determining projective invariants in imagery, *Computer Vision Image Understand.* **53**, 1991, 45–65.
11. E. Rivlin and I. Weiss, Local invariants for recognition, *IEEE Trans. Patt. Anal. Mach. Intell.* **17**, 1995, 226–238.
12. A. Bruckstein, E. Rivlin, and I. Weiss, Recognizing objects using scale space local invariants, in *Int. Conf. on Pattern Recognition*, Vienna, 1996, p. A9M.6.
13. H. Asada and M. Brady, The curvature primal sketch, *IEEE Trans. Patt. Anal. Mach. Intell.* **8**, 1986, 2–14.
14. F. Mokhtarian and A. Mackworth, Scale-based description and recognition of planar curves and two-dimensional shapes, *IEEE Trans. Patt. Anal. Mach. Intell.* **8**, 1986, 34–43.
15. R. Mehrotra and J. Gary, Similar-shape retrieval in shape data management, *IEEE Computer* **28**, 1995, 57–62.
16. A. Pentland, R. Picard, and S. Sclaroff, Photobook: Tools for content-base manipulation of image databases, in *Proceedings, SPIE Conf. on Storage and Retrieval of Image Databases II*, 1994, pp. 37–50.
17. C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovich, and W. Equitz, Efficient and effective querying by image content, *J. Intelligent Information Systems* **3**, 1994, 231–262.

18. J. Eakins, Retrieval of trade mark images by shape feature, in *Proceedings, I Int. Conf. on Electronic Library and Visual Information Research, De Montfort University, Milton Keynes, 1994*, pp. 101–109.
19. K. Shields, J. Eakins, and J. Boardman, Automatic image retrieval using shape features, *New Rev. Document Text Manage.* **1**, 1995, 183–198.
20. M. Ubel, The montage extensible datablade architecture, in *Proc. of the ACM SIGMOD Int. Conf. on Management of Data, Minneapolis, 1994*, p. 482.
21. S. Chang, Q. Shi, and C. Yan, Iconic indexing by 2-D strings, *IEEE Trans. Patt. Anal. Mach. Intell.* **9**, 1987, 413–428.
22. V. Gudivada and V. Raghavan, Design and evaluation of algorithms for image retrieval by spatial similarity, *ACM Trans. Inf. Systems* **13**, 1995, 115–144.
23. A. Soffer and H. Samet, Pictorial queries by image similarity, in *Proc. of ICPR, 1996*, pp. 114–119.
24. A. Bruckstein, N. Katzir, M. Lindenbaum, and M. Porat, Similarity-invariant signatures for partially occluded planar shapes, *Int. J. Comput. Vision* **7**, 1992, 271–285.
25. A. Bruckstein, J. Holt, A. Netravali, and T. Richardson, Invariant signatures for planar shape recognition under partial occlusion, *CVGIP: Image Understanding* **58**, 1993, 49–65.
26. T. Moons, E. Pauwels, L. V. Gool, and A. Oosterlinck, Recognition of planar shapes under affine distortion, *Int. J. Comput. Vision* **14**, 1995, 49–65.
27. J. Shen and S. Castan, An optimal linear operator for edge detection, in *Proceedings, IEEE Comput. Society Conference on Comput. Vision and Pattern Recognition, Miami, 1986*, pp. 109–114.
28. J. Shen and S. Castan, Edge detection based on multi-edge models, in *SPIE, Cannes, 1987*.
29. J. Shen and S. Castan, Further results on DRF method for edge detection, in *9 Int. Conf. on Pattern Recognition, Rome, 1988*, pp. 223–225.
30. A. Rozenfeld, *Picture Languages*. Academic Press, New York, 1979.
31. J. Köbler, U. Schöning, and J. Torán, *The Graph Isomorphism Problem*. Birkhäuser, Boston, 1993.
32. KBS Finland beer bottle collection, URL: <http://tuoppi oulu.fi/kbs/beer/kbsbeer.htm>.
33. UMD logos database, Laboratory for Language and Media Processing, Institute for Advanced Computer Studies, University of Maryland at College Park, URL: ftp://documents.cfar.umd.edu/pub/contrib/databases/UMD/logo_database.tar.
34. Search for similar shapes in the SQUID system, URL: <http://www.ee.surrey.ac.uk/Research/VSSP/imagedb/demo.html>. University of Surrey, United Kingdom.
35. A. Zisserman, D. Forsyth, J. Mundy, C. Rothwell, J. Liu, and N. Pillow, 3d object recognition using invariance, *Artificial Intelligence* **78**, 1995, 239–288.
36. J. Ullmann, *Principles of Database and Knowledge-base systems, I*. Computer Science Press, Rockville, Maryland, 1989.