# Shape Matching by Curve Modelling and Alignment

CECILIA DI RUBERTO
University of Cagliari
Dept. of Math. and Comp. Sc.
Via Ospedale 72, 09124 Cagliari
ITALY
dirubert@unica.it

MARCO GAVIANO
University of Cagliari
Dept. of Math. and Computer Science
Via Ospedale 72, 09124 Cagliari
ITALY
gaviano@unica.it

ANDREA MORGERA
University of Cagliari
Dept. of Math. and Comp. Sc.
Via Ospedale 72, 09124 Cagliari
ITALY
andrea.morgera@unica.it

*Abstract:* Automatic information retrieval in the field of shape recognition has been widely covered by many research fields. Various techniques have been developed using different approaches such as intensity-based, model-based and shape-based methods. Whichever is the way to represent the objects in images, a recognition method should be robust in the presence of scale change, translation and rotation. In this paper we present a new recognition method based on a curve alignment technique, for planar image contours. The method consists of various phases including extracting outlines of images, detecting significant points and aligning curves. The dominant points can be manually or automatically detected. The matching phase uses the idea of calculating the overlapping indices between shapes as similarity measures. To evaluate the effectiveness of the algorithm, two databases of 216 and 99 images have been used. A performance analysis and comparison is provided by precision-recall curves.

*Key–Words:* Curve alignment, Information retrieval, Object recognition, Image indexing, Precision-recall.

## 1 Introduction

Because of recent advances in computer science, content-based retrieval has received considerable attention and, consequently, improving the technology for content-based querying systems becomes more appealing. For content-based image retrieval, of course shape contains the most attractive visual information for human perception. Four important feature components are usually considered for content-based image retrieval: color, texture, shape and spatial relationship. Among these features, shape contains the most attractive visual information for human perception. Shape recognition arises in a variety of contexts, examples of which include not only retrieval by content from image databases, but also document image analysis, automated industrial inspection, analysis of medical images as well as vision-based robotics and visual tracking. Shape recognition is also a very challenging problem because of the large degree of variations of objects in 2D images: within class variations, articulations, variations in position, viewing direction, illumination, occlusion and others. Most recognition approaches tend to restrict the representation space of objects by focusing on the main variations in the projected intensity pattern (intensity-based), on the within category variations of shape when restricted to a set of models (model-based), and on the variations of the apparent contour (shape-based). The problem of recognition when shapes are encoded by their contours is interesting for many reasons: contours are easy to obtain and can be used for shape representation regardless of shape convexity; they are one of the most commonly used shape descriptors; since they may be considered as high order curves, algorithms for contour-based recognition are potentially extendible to more generic shape classes including cursive words and hand-drawn sketches. Boundary-based representations of shapes include unorganized point sets [4, 7], curves [3, 24], and medial axis or shock graphs [9, 17, 20]. Outlines of shapes have been represented as unorganized point sets, which are typically matched using an assignment algorithm [5, 7, 13]. In [13] graduated assignment is used to match image boundary features. In [5, 14] the Hungarian method is used to match ”shape contexts”, where a coarse histogram of the relative location of the remaining points is used as feature. A related approach [15] uses the generalized Hausdorff distance to compare edge maps of 2D images. These methods have the advantage of not requiring ordered boundary points, but the match does not necessarily preserve the coherence of shapes and may have problems when dealing with noisy or partial data. In many object recognition and shape-based indexing applications, the objects are represented by their outline curves and matched [12, 16, 24]. Matching typically involves finding a mapping from one curve to the other that minimizes an ”elastic” performance functional, which penalizes ”stretching” and ”bending” [3, 24]. The

minimization problem in the discrete domain is transformed into one of matching shape signatures with curvature, bending angle, or orientation as attributes [3, 12, 16]. Generally, the curve-based methods suffer from one or more of the following drawbacks: asymmetric treatment of the two curves, lack of rotation and scaling invariance, and sensitivity to articulations and deformations of parts.
Generally speaking, in order to be effective, a recognition measure should satisfy the following properties proposed by Arkin et al. [2]:

- The measure should be a metric.
- It should be invariant under translation, rotation, and scale change.
- It should be easy to compute.
- It should match intuitive notions of shape resemblance.

Region-based techniques seem effective but arduous and domain-dependent. On the contrary, representing a shape with its contour is much more efficient than with the overall region. However, the contour captured by a camera under arbitrary orientation may have been distorted by certain geometric transformations. In [19] the notion of alignment curve is introduced in order to ensure a symmetric treatment of the curves, and rotation-invariance is guaranteed by the use of intrinsic properties. Current curve alignment methods can be classified into two categories: methods based on rigid transformations [1], and those based on non-rigid deformations [3, 24]. Methods based on rigid transformations rely on matching feature points by finding the optimal rotation, translation, and scaling parameters. They are sensitive to articulations, deformations of parts, occlusion, and other variations in the object form. Methods based on non-rigid deformations model articulation and other deformations by finding the mapping from one curve to another that minimizes a performance functional consisting of stretching and bending energies. These methods suffer from one or more of the following drawbacks: asymmetric treatment of the two curves, sensitivity to sampling of the curves, lack of rotation and scaling invariance, and sensitivity to occlusion and articulations.
Our approach to find the optimal correspondence between 2D curves relies on using the intrinsic properties of the curves in an energy minimization framework by overlapping the curves and calculating the degree of alignment as a measure of similarity of the curves. In Section 2 we present the mathematical background used by the proposed alignment method. Section 3 describes the algorithms for finding the optimal alignment and shape matching. Section 4 illustrates the proposed curve alignment performance in several applications including prototype formation, object recognition and indexing into image databases.

## 2 The mathematical formulations

In this section we write down the mathematical results that will provide the guidelines of the method described in section 3.

**Definition 2.1** (Image contour or curve). *A set $I$ of $n$ points*

$$I : \{P_i \equiv (x_i, y_i) \in R^2, i = 1, \ldots, n \mid P_1 = P_n; \\ P_i \neq P_j,\ i = 1, \ldots n-2,\ j = i+1, \ldots n-1\}$$

*is said to be a* image contour *or* curve *if, for any line segment with endpoints two successive points in $I$, the following*

$$\begin{aligned} \overline{P_iP_{i+1}} \cap \overline{P_jP_{j+1}} &= \emptyset,\ i = 1, \ldots n-3, \\ j = i+2, \ldots, n-1 & \\ \overline{P_iP_{i+1}} \cap \overline{P_{i+1}P_{i+2}} &= P_{i+1},\ i = 1, \ldots n-2 \end{aligned}$$

*holds.*

We can state that an *image contour* or *curve* $I$ is the set of the successive vertices of a closed polygon that is not self intersecting. In the sequel we denote by $\bar{I}$ the closed polygon associated with $I$.

**Definition 2.2** (Translation). *Given $(x_t, y_t) \in R^2$, a translation mapping $T : R^2 \to R^2$ is defined by*

$$T : \begin{cases} x' &= x + x_t \\ y' &= y + y_t. \end{cases} \tag{2.1}$$

**Definition 2.3** (Rotation). *Given $(x_r, y_r) \in R^2$ and $\theta \in [0, \pi]$, a rotation mapping $R : R^2 \to R^2$ through the angle $\theta$ and the rotation center $(x_r, y_r)$ is defined by*

$$R : \begin{cases} x' = \cos(\theta) * (x - x_r) - \sin(\theta) * (y - y_r) + x_r \\ y' = \sin(\theta) * (x - x_r) + \cos(\theta) * (y - y_r) + y_r. \end{cases}$$

**Definition 2.4** (Scaling). *Given two real numbers $s_1$ and $s_2$, a scaling mapping $S : R^2 \to R^2$ is defined by*

$$S : \begin{cases} x' &= s_1 * x \\ y' &= s_2 * y. \end{cases} \tag{2.2}$$

**Proposition 2.5** *Let $I'$ be a* curve *obtained from a* curve *$I$ by a chain application $S * R * T$ of a translation $T$, a rotation $R$ and a scaling $S$ with parameters $(x_t, y_t)$, $(\theta, x_r, y_r)$ and $(s_1 = s_2 = s)$, respectively.*

*Let $I^b$ be the* image contour *obtained by the chain application to $I'$ of the mapping $map_1$*

$$map_1 : \begin{cases} x' &= x + x_c - x'_c \\ y' &= y + y_c - y'_c \end{cases} \quad (2.3)$$

*where $C \equiv (x_c, y_c)$ and $C' \equiv (x'_c, y'_c)$ denote the centroid of $I$ and $I'$, respectively. Further, let $I''$ be obtained by the application to $I^b$ of the mappings $map_2$ and $map_3$ given by*

$$map_2 : \begin{cases} x' &= \alpha * (x - x_c) - \beta * (y - y_c) + x_c \\ y' &= \beta * (x - x_c) + \alpha * (y - y_c) + y_c \end{cases} \quad (2.4)$$

$$map_3 : \begin{cases} x' &= \bar{s} * (x - x_c) + x_c \\ y' &= \bar{s} * (y - y_c) + y_c \end{cases} \quad (2.5)$$

*where*

$$\alpha = \frac{(x_1 - x_c)(x_0 - x_c) + (y_1 - y_c)(y_0 - y_c)}{((x_1 - x_c)^2 + (y_1 - y_c)^2)^{0.5}((x_0 - x_c)^2 + (y_0 - y_c)^2))^{0.5}}$$

$$\beta = \frac{(y_1 - y_c)(x_0 - x_c) - (x_1 - x_c)(y_0 - y_c)}{((x_1 - x_c)^2 + (y_1 - y_c)^2)^{0.5}((x_0 - x_c)^2 + (y_0 - y_c)^2))^{0.5}}$$

$$\bar{s} = \frac{((x_1 - x_c)^2 + (y_1 - y_c)^2)^{0.5}}{((x_0 - x_c)^2 + (y_0 - y_c)^2))^{0.5}}$$

*and $P_{max} \equiv (x_1, y_1)$ and $Q_{max} \equiv (x_0, y_0)$ the points in $I$ and $I^b$, respectively, with maximum distance from $(x_c, y_c)$, then $I'' = I$.*

**Proof.** The chain application of $S * R * T$ maps the centroid $C$ of $I$ into the centroid $C'$ of $I'$ and maps $P_{max}$ into $P'_{max}$ with $P'_{max}$ the point in $I'$, with maximum distance from $C'$. The application of $map_1$ maps, through a translation, $C'$ into $C$ and the application of $map_2$ cancels the rotation done in $R$. Finally, $map_3$ maps $P'_{max}$ into $P_{max}$ by a scaling. Hence, we can state $I = I''$. □

**Proposition 2.6** *Let $I'$ be a* curve *obtained from a* curve *$I$ by a chain application $S * R * T$ of a translation $T$, a rotation $R$ and a scaling $S$ with parameters $(x_t, y_t)$, $(\theta, x_r, y_r)$ and $(s_1, s_2)$, respectively. Let $I^b$ be the* curve *obtained by the application to $I'$ of the mapping $map_1$*

$$map_1 : \begin{cases} x' &= x + x_c - x'_c \\ y' &= y + y_c - y'_c \end{cases} \quad (2.6)$$

*where $C \equiv (x_c, y_c)$ and $C' \equiv (x'_c, y'_c)$ denote the centroid of $I$ and $I'$, respectively. Further, let $I^{bb}$ be obtained by the application to $I^b$ of the mapping $map_2$*

$$map_2 : \begin{cases} x'(\theta) = \cos\theta * (x - x_c) \\ -\sin\theta * (y - y_c) + x_c \\ y'(\theta) = \sin\theta * (x - x_c) \\ +\cos\theta * (y - y_c) + y_c \end{cases} \quad (2.7)$$

*with $\theta \in [0, 2\pi]$. Further, let $I''$ be obtained by the application to $I^{bb}$ of the mapping $map_3$ given by*

$$map_3 : \begin{cases} x'(\theta) &= \bar{s}_1 * (x(\theta) - x_c) + x_c \\ y'(\theta) &= \bar{s}_2 * (y(\theta) - y_c) + y_c \end{cases} \quad (2.8)$$

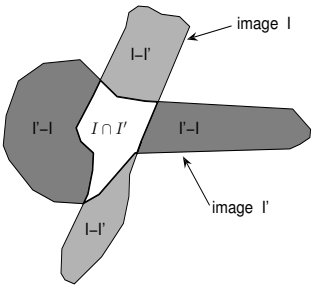


Figure 1: The dark regions contribute to define $ind_1$

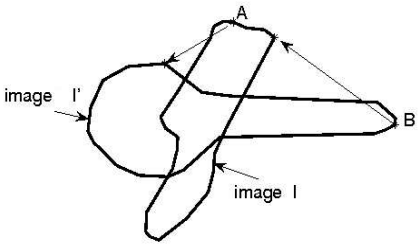


Figure 2: The maximum of the distances of A from $I'$ and B from $I$ gives $ind_2$.

*where*

$$\bar{s}_1 = \frac{x_0}{x_1}, \quad \bar{s}_2 = \frac{y_0}{y_1}$$

*with $P_x \equiv (x_1, y_c)$ and $P_y \equiv (x_c, y_1)$ the intersection points, with the smallest distance from $(x_c, y_c)$, of the boundary of $\bar{I}$ with the straight lines $y = y_c$ and $x = x_c$, respectively, and $Q_x \equiv (x_0, y_c)$ , $Q_y \equiv (x_c, y_0)$ the intersection points of the boundary of $\bar{I}^{bb}$ with the same straight lines. Then, there exists $\theta^* \in [0, 2\pi]$ such that $I'' = I$.*

**Proof.** The chain application of $S * R * T$ maps the centroid $C$ of $I$ into the centroid $C'$ of $I'$. The application of $T * R$ can be rewritten as an application of a rotation $R'$ through a suitable angle $\theta'$ with the centroid of $I$ as rotation center and of a suitable translation $T'$. The transformation $map_1$ maps $C'$ into $C$. The application of $map_2$ with angle $-\theta'$ and the scaling in $map_3$ moves $I''$ to the original curve. □

In order to compare shapes, we give now a definition that will provide an *overlapping index* between two curves $I$ and $I'$. Such an index measures the degree of alignment or overlapping of two curves and it does represent a measure of similarity between two shapes.

**Definition 2.7** (Overlapping index 1). *Let* $I \equiv (P_i,\ i = 1,\ldots,n)$ *and* $I' \equiv (P'_j,\ j = 1,\ldots,m)$ *be two* curves *and* $\bar{I}$ *and* $\bar{I}'$ *the associated polygons with vertices* $I$ *and* $I'$*, respectively. Let*

$$ind_1(I,I') = max\left(\frac{area(\bar{I}-\bar{I}')}{area(\bar{I})},\ \frac{area(\bar{I}'-\bar{I})}{area(\bar{I})}\right)$$

*where* $area(\bar{I})$ *stands for the area of the set* $\bar{I}$*, then* $ind_1(I,I')$ *will evaluate the overlapping degree between* $I$ *and* $I'$*.*

Fig.1 explains the definition, the slightly dark region and the darker region contribute to define $ind_1$. That value goes to zero as the areas of both regions reduce to zero, i.e. the regions are identical. In the sequel we shall make use of a second definition of the *overlapping index* that will turn out to be computationally less expensive.

**Definition 2.8** (Overlapping index 2). *Let* $I \equiv (P_i,\ i = 1,\ldots,n)$ *and* $I' \equiv (P'_j,\ j = 1,\ldots,m)$ *be two* image contours *and* $\bar{I}$ *and* $\bar{I}'$ *the associated polygons. The overlapping index* $ind_2$ *is defined as*

$$ind_2(I,I') = \max(\max_{i=1,\ldots,n}(distance(P_i,\mathfrak{F}(\bar{I}')),\ \max_{j=1,\ldots,m}(distance(P_j,\mathfrak{F}(\bar{I})))$$

*where* $\mathfrak{F}(S)$ *denotes the boundary of the set* $S$ *and* $distance$ *refers to the euclidian distance.*

In Fig.2 $A \in I'$ and $B \in I$ are the points with the maximum distance from $I$ and $I'$ respectively. The maximum of these distances gives $ind_2$. Note that given two image contours $I$ and $I'$ such that $ind_1(I,I') = 0$ then the associated polygons fully overlap, i.e., $\bar{I} = \bar{I}'$; that is not true if $ind_2(I,I') = 0$. The following example

$$\begin{aligned} I &\equiv \{(-1,1),(0,2),(0,0.5),(1,1),(1,-1),(0,-2),\\ &\quad (0,-0.5),(-1,-1),(-1,1)\}\\ I' &\equiv \{(-1,1),(0,0.5),(0,2),(1,1),(1,-1),(0,-2),\\ &\quad (0,-0.5),(-1,-1),(-1,1)\} \end{aligned}$$

shows that. Whenever we don't make reference to a specific definition, by $ind(I,I')$ we shall denote any overlapping index between $I$ and $I'$.

# 3 The alignment method for shape matching

The mathematical definitions given in section 2 suggest the main features of our method: to compare two shapes $I$ and $I'$ we assume that $I'$ is derived from $I$ by some linear transformations. That is in order to compare two curves $I$ an $I'$ we proceed as follows:

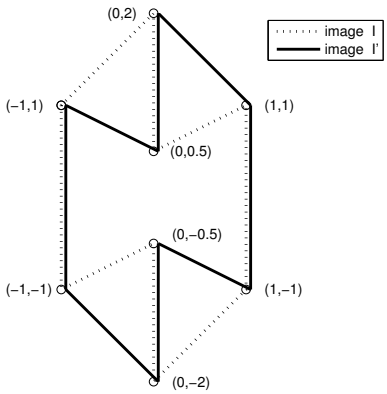


Figure 3: Image contours $I$ and $I'$ with $ind_2(I,I') = 0$ but $\bar{I} \neq \bar{I}'$

1. overlap by a translation $I$ and $I'$ so that $C = C'$, with $C$ and $C'$ being the centroids of $I$ and $I'$, respectively;
2. carry out a rotation on $I'$ with rotation center $C$;
3. perform a scaling;
4. calculate the overlapping index $ind(I,I')$ .

We assume that two cases can take place: $I$ and $I'$ may differ either $(a)$ by a equal scaling along both axis, either $(b)$ by non equal scaling along the axis.
To handle cases (a) and (b) we shall consider two algorithms $OA\_1$ and $OA\_2$; in the first case we rewrite the four steps sketched above as

**Algorithm OA_1** (single scaling)

1. find centroids $C$ and $C'$ and translate $I$ and $I'$ so that their centroids are the axis origin $O \equiv (0,0)$;
2. find points $P_{max}$ and $P'_{max}$ in $I$ and $I'$, respectively, whose distance from $O$ is maximum;
3. rotate $I$ by the angle $(P_{max} - O)^T(P'_{max} - O)/(\|P_{max}\|\|P'_{max}\|)$;
4. scale $I$ by $(\|P_{max}\|/\|P'_{max}\|)$.
5. calculate $ind(I,I')$ .

In the second case we rewrite the four steps as follows:

**Algorithm OA_2** (double scaling)

1. find centroids $C$ and $C'$ and translate $I$ and $I'$ so that their centroids are the axis origin $(O \equiv (0,0)$; let $I^a$ and $I^b$ be the new curves;
2. let $\theta(i)$ $i = 1,\ldots,n$ a vector of n points equally spaced in $[0,2\pi]$;
3. find $\theta(i^*)$ such that

$$ind(I^a, I''(\theta(i^*))) = \min_i ind(I^a, I''(\theta(i)))$$

with $I''(\theta)$ obtained as result of a chain application of 2.7, and 2.8 to $I^b$.

### 3.1 Shape representation and modelling

In our approach shapes are represented by their contours. Given a set $S$ of assigned shapes in order to find all the similar shapes in $S$ or that share a similar contour, we consider a model $M$ which all the shapes of $S$ must be compared to. Such a model is made up of one ore more curves $m_i, i = 1 \dots m$ and we measure the shape similarity of an element $I$ of $S$ to $M$ by calculating the overlapping indices between $I$ and each element $m_i$ of $M$. The smallest value of these indices is assumed as similarity measure of $I$ to the model and is denoted by $ind(I, M)$:

$$ind(I, M) \equiv \min_{i=1...m} ind(I, m_i)$$

The shape models have been drawn from real image contours by detecting the most significant points of the contours and joining them by straight lines. There exist a wide variety of ways to achieve a polygonal approximation of a contour [6, 11, 22]. The significant points are used both to reserve the shape of curve and to reduce the amount of data. In order to make the method more efficient, the elements of $M$ will be approximated contours, usually, of real images. Indeed, this allows to define a image contour of a model by a number of points smaller than the number of points that would define the original image. In Fig.4 two real image contours and approximated contours are depicted. In Fig.4(a) the original contour, represented by a smaller solid line, is defined by 379 points while the approximated contour, represented by a larger solid line, is defined by 40 points. In Fig.4(b) we have 302 points and 30 points, respectively. In our experiments we have detected the dominant points both manually and automatically.

Let's describe our automatic approach for dominant point detection [10]. In order to reach a complete set of dominant points of a given a contour, we need of a initial group of starting points. By the result of the four vertex theorem [8] we choose to locate four initial points by searching for local maxima and minima in the signature of the shape. A point is considered a maximum peak if it has the maximal value, and was preceded (to the left) by a value lower by $\Delta$. In our case we use $\Delta$ = 0.5. In such way, searching the minimal values, we find minimum peaks. We use the first and second maximum and minimum peaks so we have four starting points for our method.

After the initial points setup the method builds a set of dominant points in the following way:

1) Let $D = \{d_i, \quad i = 1, ..., m\}$ the intial dominant points set. For each pair of points $d_i d_{i+1}, \quad i = 1, 2, ....m$ where $d_{i+1}$ is a neighbor of

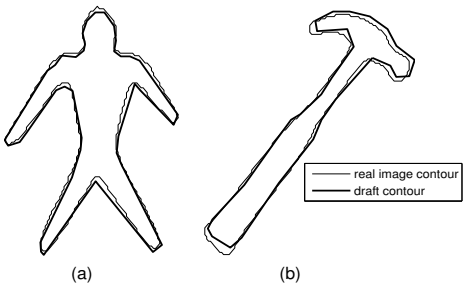


Figure 4: Image contours of real images and approximated contours.

$d_i$ (modulo $m$) it is possible to calculate the sum of distance from each contour point $p_j \quad (j = 1, ..., n)$ between $d_i$ and $d_{i+1}$ and the line that connects the two points:

$$dist_j = \frac{|(x_{i+1} - x_i)(y_i - y_j) - (x_i - x_j)(y_{i+1} - y_i)|}{\sqrt{((x_{i+1} - x_i))^2 + ((y_{i+1} - y_i))^2}}$$

where $(x_{i+1}, y_{i+1})$, $(x_i, y_i)$ and $(x_j, y_j)$ are coordinates of $d_{i+1}$, $d_i$ and $p_j$ respectively. A global distance is calculated as :

$$T = \sum_j dist_j \quad , \quad j = 1, ..., n \qquad (3.9)$$

2) In order to find dominant points the value $T$ of the previous step is compared to a treshold $s$ defined as $s = p * l$, where $p$ is an input parameter of the algorithm and $l$ is the number of points of the line that connects $d_i$ and $d_{i+1}$. If $T > s$ we add an intermediate point between $d_i$ and $d_{i+1}$ looking for the point with maximum distance to the chord. The current set of points and the new dominant points added are the set of points for the next iteration.

3) The algorithm stops when, after two following iterations, the number of the current set of points is equal to the number of the previous set.

Given a dominant points set $D$ we adopt a refinement technique to suppress dominant point which are near enough between them by a predefined threshold $\tau$ = 0.007.

### 3.2 Shape matching and timing analysis

We now give a remark allowing to reduce substantially the computation of the overlapping index 2.

In comparing an element $I$ of $S$ to all elements in $M$, the calculation of $ind_2(I, M)$ has to be carried out by calculating first $ind_2(I, m_1)$ and then $ind_2(I, m_i), \quad i = 2, \dots m.$ Once a

$ind_2(I, m_j), j \in \{1, \ldots, m\}$ has been carried out we can start to compute $ind_2(I, m_i), i > j$. However, if for a point $(\tilde{x}, \tilde{y}) \in I$ the overlapping index $ind_2((\tilde{x}, \tilde{y}, \mathfrak{F}(m_i))$ is greater than or equal to $ind_2(I, m_j)$ then $ind_2(I, m_i)$ will be greater or equal to $ind_2(I, m_j)$. As a consequence we don't need to complete the calculation of $ind_2(I, m_i)$. This feature will be present in the implementation of our algorithms.

Finally, for algorithm $OA\_1$ we shall consider an option that carries out a sort of suboptimization procedure. In comparing contour $I$ to the contour of each element $m_j$ of $M$ we perform rotations of $m_j$ through angles $\theta(i)$ and calculate the best $ind(I, m_j)$. We proceed much the same as in algorithm $OA\_2$.

The *overlapping* algorithm $OA\_1$ is now written in *pseudo-code* language. It calls four procedures: *find_centroid*, *find_max_point*, *rotate*, and *find_ind* and, eventually, it uses *option_A*

**Algorithm, single scaling (OA_1)** m≡ number of contours in model $M$.

**input_model**: [x_s,y_s,n_s,m];
[xb,yb]=find_centroid (x_s,y_s,n_s,m);
*translate model centroids to (0,0)*
**for** $j = 1 : m$
x_s(j,:)=x_s(j,:)-xb(j); y_s(j,:)=y_s(j,:)-yb(j);
**end**
[x1,y1,n1]=input_data $I \in S$;
[xb1,yb1]=find_centroid(x1,y1,n1,1);
*translate image centroid to (0,0)*
x1=x1-xb1; y1=y1-yb1
nn1=find_max_point(x1,y1);
best=$\infty$;
**for** i_m=1:m
n=n_s(i_m);
x=x_s(i_m,1:n); y=y_s(i_m,1:n);
best_buf=find_ind(x,y,n,x1,y1,n1,best)
**if** best_buf $<$ best best=best_buf; **end**
nn=find_max_point(x,y);
[xx1,yy1,sca]= rotate_scale(x1,y1,x1(nn1),
y1(nn1),x(nn),y(nn));
best_buf=find_ind(x,y,n,sca*xx1,sca*yy1,n1,best);
**if** best_buf $<$ best best=best_buf; **end**
Option A
**end**
**output**: best.

We give the main features of the procedures called by the $OA\_1$ algorithm.

**Find_centroid.** This function reads the contours of the $m$ images that made up the model. Then it calculates and returns their centroids values. The centroid $(x_c, y_c)$ of $I \equiv \{P_i \equiv (x_i, y_i), i = 1, \ldots, n\}$ is given by

$$x_c = \frac{1}{6A}\sum_{i=0}^{n-1}(x_i + x_{i+1})(x_i y_{i+1} - x_{i+1}y_i)$$

$$y_c = \frac{1}{6A}\sum_{i=0}^{n-1}(y_i + y_{i+1})(x_i y_{i+1} - x_{i+1}y_i)$$

$$A = \frac{1}{2}\sum_{i=0}^{n-1}(x_i y_{i+1} - x_{i+1}y_i).$$

**Find_max_point.** This function reads the coordinates of a contour $I$ and, then, finds and returns the index of the element of $I$ whose distance from the $(0, 0)$ is the largest one.

**Rotate_scale.** This function reads the coordinates of a contour I, then applies to $I$ a transformation so that it is rotated by the angle

$$\theta = \frac{(x(nn), y(nn))^T (x1(nn1), y1(nn1))}{\|(x(nn), y(nn))\| \, \|(x1(nn1), y1(nn1))\|}$$

and scaled by

$$sca = \frac{|(x(nn), y(nn)\|}{\|(x1(nn1), y1(nn1))\|}.$$

It returns the transformed $I$.

**Find_ind.** This function reads the coordinates of a contours $I$ and $I'$ and a real value $best$, then calculates and returns the overlapping index $ind(I, I')$ between the two contours. If $ind(I, I')$ is $ind_2(I, I')$ the procedure is stopped as soon as any point $P$ of $I$ has a distance from $\mathfrak{F}I'$ greater than $best$ either any point $P$ of $I'$ has a distance from $\mathfrak{F}I$ greater than $best$.

**Option_A for OA_1.** It consists of the following program segment:

% option: find the best rotation with rot_no
a positive integer
teta=[0 : 2 * $\pi$/(rot_no − 1) : 2 * pi];
x_start=x; y_start=y;
**for** j= 1: rot_no
x=cos(teta(j))*x_start-sin(teta(j))*y_start;
y=sin(teta(j))*x_start+cos(teta(j))*y_start;
best_buf=find_ind($x, y, n, scal * x1,$
$scal * y1, n1, best$);
**if** best $>$ best_buf best=best_buf; **end**
**end**

The *overlapping* algorithm $OA\_2$ is now written in *pseudo-code* language. It calls procedures already introduced for $OA\_1$ and *find_zero*.

**Algorithm, double scaling (OA_2)**

input_model: [x_s,y_s,n_s,m];
[xb,yb]=find_centroid(x_s,y_s,n_s,m);
*translate model centroids to (0,0)*
**for** j=1:m

```
    x_s(j,:)=x_s(j,:)-xb(j); y_s(j,:)=y_s(j,:)-yb(j);
end
[x1,y1,n1]=input_data I ∈ S;
[xb1,yb1]=find_centroid(x1,y1,n1,1);
translate image centroid to (0,0)
x1=x1-xb1; y1=y1-yb1
best=∞;
for i_m=1:m
  n=n_s(i_m);
  x=x_s(i_m,1:n); y=y_s(i_m,1:n);
  best_buf=find_ind(x,y,n,x1,y1,n1,best)
   if best_buf < best best=best_buf; end
  x1_0=find_zero(x1,y1,n1);
  y1_0=find_zero(y1,x1,n1);
  find the best rotation
  teta=[0:2*π/(rot_no-1):2*π];
  x_start=x; y_start=y;
  for j= 1:rot_no
    x=cos(teta(j))*x_start-sin(teta(j))*y_start;
    y=sin(teta(j))*x_start+cos(teta(j))*y_start;
    x_0=find_zero(x,y,n);
    y_0=find_zero(y,x,n);
    scal1=abs(x_0/x1_0);
    scal2=abs(y_0/y1_0);
    best_buf=find_ind(x,y,n,scal*x1,scal*y1,n1,best);
    if  best > best_buf best=best_buf; end
  end
end
output: best.
```

**Find_zero.** This reads the coordinates of a contour $I \equiv (x, y)$ then calculates and returns the smallest distance from $O \equiv (0, 0)$ to the intersection of the boundary of $I$ with the $x\ axis$.

# 4 Experimental results

In this section we illustrate our shape recognition approach with some experiments. We have evaluated the matching ability of the proposed method by indexing two databases of shapes, both constructed by Kimia's group [18, 17]. The two proposed algorithms, $OA\_1$ and $OA\_2$, have been first tested on a set $S^1$ of 216 images $I_i, i = \ldots, 216$ (see Fig.5). The database consists of twelve shapes each from eighteen families (or classes) for a total of 216 shapes. In order to simplify the presentation of our procedure, we first assume that we want to identify the shapes of just one family $F \subset S^1$; for instance, we want to identify all screwdrivers. We assume $F$ made up of $n_F = n_2 - n_1 + 1$ images and $F \equiv \{I_j, j = n_1, \ldots, n_2\}$, where $I_j$ is the $j^{th}$ image in $S^1$. The indexing process is achieved by first defining a model $M$ for the family and then by comparing each element

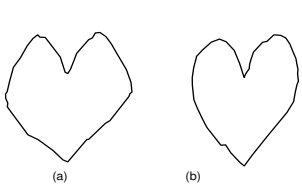


Figure 6: A model made up of two image contours.

of $S$ with the model and calculating the overlapping indices $ind(I_j, M),\ j = 1, \ldots, 216$. The overlapping indices are then sorted in ascending order and stored as a vector named $res\_sort$. The algorithm is considered fully effective if the overlapping indices of the images in $F$ are in the first $n_F$ positions of $res\_sort$. Further, whenever this does not take place, we consider how much the positions of the overlapping indices of the images in $F$ which are not in the first $n_F$ positions are far from the $n_F$ position. Hence, we evaluate the effectiveness of the algorithm by defining two accuracy values $acc\_1$ and $acc\_2$. Given

$$\bar{F} = \{j | n_1 \leq j \leq n_2,\ pos(j) \leq n_F\},$$

$$\underline{F} = \{j | n_1 \leq j \leq n_2,\ pos(j) > n_F\},$$

$$acc\_1 \equiv \frac{card(\bar{F})}{n_F},$$

$$acc\_2 \equiv \frac{\sum_{j \in \underline{F}} (pos(j) - n_F)}{card(\underline{F})}$$

where by $pos(j)$ and $card(\bar{F})$ we denote the position of image $I_j$ in $res\_sort$ and the cardinality of $\bar{F}$ respectively. In Fig.7 we show the results obtained by comparing the set $S^1$ with the model made up of the two images given in Fig.6. The filled circles represent contours in the family to be identified; the empty ones represent those not in the family. In such a case all the contours in the family are identified ($acc\_1$=0, $acc\_2$=0). We now present the overall results of our experiments. The set $S^1$ is partitioned into 18 families $F_i, i = 1, \ldots 18$: bones, glasses, and so on. Since our task is to identify the shapes of each family (or class), we define four sets $M_{1,1}$, $M_{1,2}$, $M_{2,1}$, and $M_{2,2}$ each made up of 18 models, one for each family of images to identify. For each family, both the models in $M_{1,1}$ and $M_{1,2}$ consist of one image model; each model in $M_{1,1}$ is an approximated contour chosen such that its mean overlapping $ind_1$ calculated with respect to all the images of the family to be identified provides the minimum value. In the same way we have chosen the models in $M_{1,2}$, but calculating $ind_2$. That is, only one image model has been selected as the most representative approximated contour of a family with respect to $ind_1$ or to $ind_2$. The models in $M_{2,1}$ and

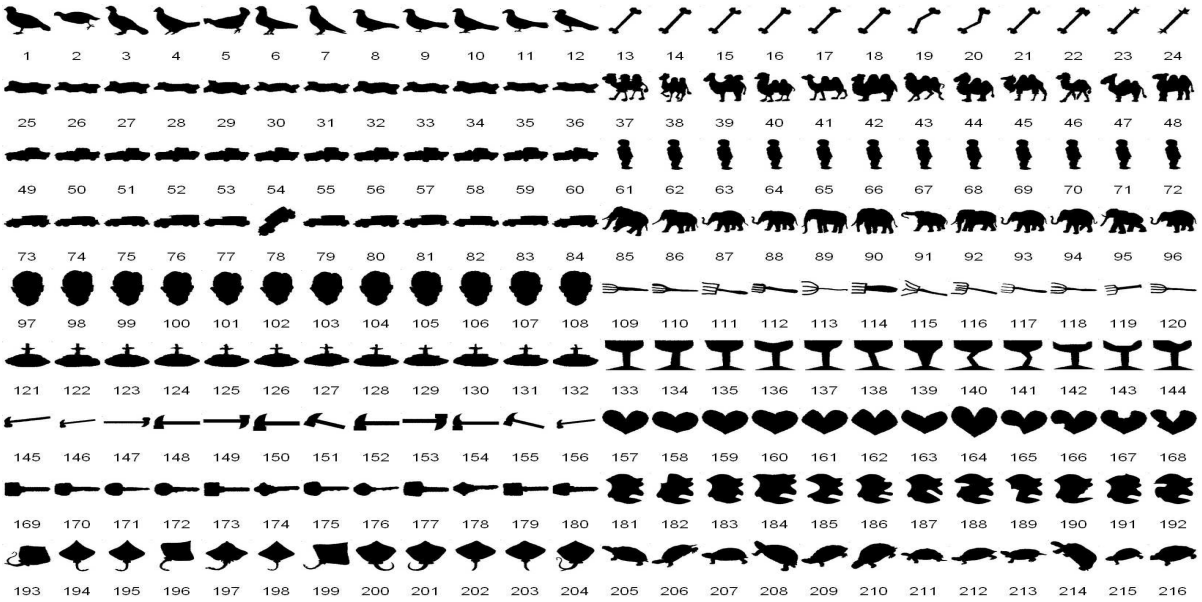

Figure 5: A database of 216 shapes: set $S^1$.

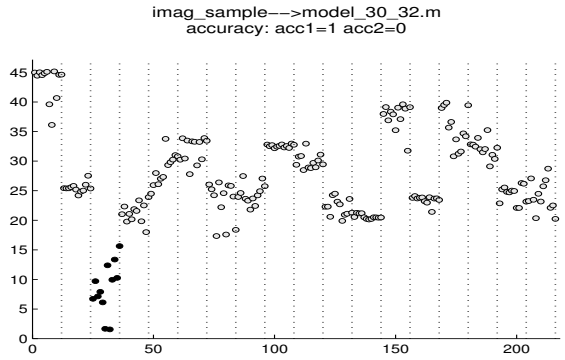


Figure 7: Plot of the overlapping indices of all images.

$M_{2,2}$ consist of from one to two images. They have been chosen trying to minimize the overlapping indices $ind_1$ and $ind_2$. In our case we have

$$M_{1,1} \equiv \{3, 13, 25, 44, 57, 67, 79, 89, 103, 109, 129, 142, 145, 168, 173, 187, 194, 207\}$$

$$M_{1,2} \equiv \{1, 13, 25, 37, 57, 67, 84, 89, 102, 109, 123, 135, 145, 168, 173, 187, 195, 207\}$$

$$M_{2,1} \equiv \{3, 13, 25, 44, 54\_56, 67\_68, 75\_82, 89, 103, 109, 128\_129, 142, 146\_152, 168, 173\_180, 186\_190, 193\_197, 208\_216\}$$

$$M_{2,2} \equiv \{1, 13, 30\_32, 38\_42, 49\_52, 67, 75\_84, 89, 102, 109, 123\_125, 135, 147\_155, 168, 171\_175, 187, 195\_201, 205\_207\}.$$

Each element in the model sets corresponds with the shape number in the image database. We run algorithms $OA\_1$ and $OA\_2$ with various settings:

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | $M_{1,1}$ | 0.83 | 8.23 | 23 |
| 0 | 1 | 40 | $M_{1,1}$ | 0.86 | 8.18 | 596 |
| 0 | 2 | 0 | $M_{1,2}$ | 0.86 | 6.64 | 3 |
| 0 | 2 | 40 | $M_{1,2}$ | 0.89 | 6.30 | 18 |
| 1 | 1 | 0 | $M_{1,1}$ | 0.71 | 15.47 | 21 |
| 1 | 1 | 40 | $M_{1,1}$ | 0.79 | 10.28 | 587 |
| 1 | 2 | 0 | $M_{1,2}$ | 0.75 | 16.12 | 2 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.87 | 5.84 | 18 |

Table 4.1: Results of algorithm $OA\_1$ with models of one image (set $S^1$).

1. the overlapping index can be $ind_1$ or $ind_2$;
2. the contour image of $S$ may be transformed by a chain application of the map $R * S * T$ with $R$ a rotation through an angle $\theta$ chosen at random in the interval $[0, \pi]$, $S$ a scaling and $T$ a translation;
3. the option A in $OA\_1$ may be or not be present; whenever is present $rot\_no = 40$;
4. $OA\_2$ is run with $rot\_no = 40$.

The scaling $S$ is assumed equal along both axis for $OA\_1$, while is not equal in the case of $OA\_2$.
The results of the numerical experiments referring to $OA\_1$ are summarized in tables 4.1 and 4.2; while those referring to $OA\_2$ are in 4.3 and 4.4. The $acc\_1$ and $acc\_2$ appearing in the tables are mean values of the values calculated for each family.

We now report the results gathered by testing algorithms $OA\_1$ and $OA\_2$ for another set of images

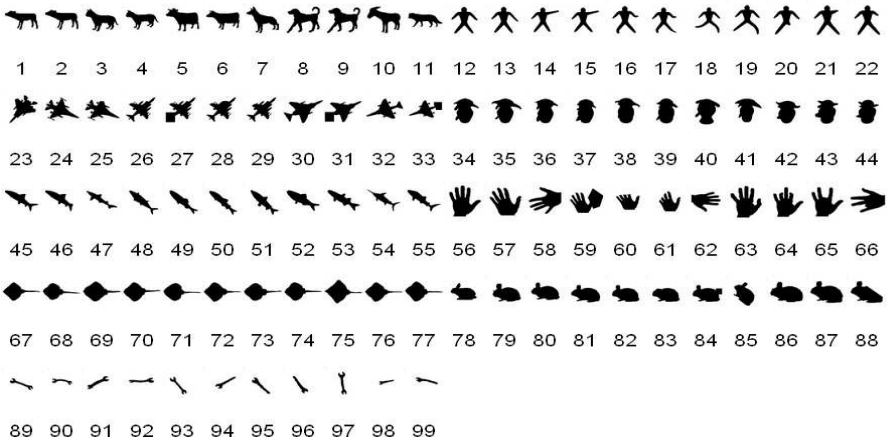


Figure 8: A database of 99 shapes: set $S^2$.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | $M_{2,1}$ | 0.93 | 6.92 | 35 |
| 0 | 1 | 40 | $M_{2,1}$ | 0.89 | 7.37 | 926 |
| 0 | 2 | 0 | $M_{2,2}$ | 0.97 | 10.83 | 4 |
| 0 | 2 | 40 | $M_{2,2}$ | 0.92 | 3.96 | 25 |
| 1 | 1 | 0 | $M_{2,1}$ | 0.79 | 13.71 | 32 |
| 1 | 1 | 40 | $M_{2,1}$ | 0.86 | 10.07 | 919 |
| 1 | 2 | 0 | $M_{2,2}$ | 0.80 | 21.67 | 3 |
| 1 | 2 | 40 | $M_{2,2}$ | 0.92 | 8.11 | 24 |

Table 4.2: Results of algorithm $OA\_1$ with models of up to two images (set $S^1$).

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 1 | 40 | $M_{1,1}$ | 0.80 | 24.20 | 637 |
| 0 | 2 | 40 | $M_{1,2}$ | 0.84 | 14.61 | 24 |
| 1 | 1 | 40 | $M_{1,1}$ | 0.69 | 39.90 | 637 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.79 | 29.48 | 19 |

Table 4.3: Results of algorithm $OA\_2$ with models of one image (set $S^1$).

$S^2$, again created by Kimia's group [18, 17](Fig.8). There are 99 shapes in this database from nine families, which were collected from a variety of sources: fish, and sea-animals from Farzin Mokhtarian, "greebles" from Mike Tarr, and tools from Stan Sclaroff. Eleven shapes are included from each family so that shapes have variations in form, and as well transformations such as occlusion, articulation and missing parts. The numerical experiments have been carried out as for the set $S^1$. In such a case the model set are given by

$$M_{1,1} \equiv \{3, 17, 30, 39, 51, 56, 72, 80, 99\}$$
$$M_{1,2} \equiv \{3, 17, 32, 38, 53, 56, 76, 87, 98\}$$

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 1 | 40 | $M_{2,1}$ | 0.83 | 20.31 | 917 |
| 0 | 2 | 40 | $M_{2,2}$ | 0.89 | 8.85 | 25 |
| 1 | 1 | 40 | $M_{2,1}$ | 0.72 | 44.60 | 989 |
| 1 | 2 | 40 | $M_{2,2}$ | 0.80 | 25.50 | 25 |

Table 4.4: Results of algorithm $OA\_2$ with models of up to two images (set $S^1$).

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | $M_{1,1}$ | 0.84 | 6.33 | 5 |
| 0 | 1 | 40 | $M_{1,1}$ | 0.87 | 7.50 | 131 |
| 0 | 2 | 0 | $M_{1,2}$ | 0.86 | 8.78 | 0 |
| 0 | 2 | 40 | $M_{1,2}$ | 0.83 | 13.80 | 5 |
| 1 | 1 | 0 | $M_{1,1}$ | 0.71 | 10.03 | 4 |
| 1 | 1 | 40 | $M_{1,1}$ | 0.84 | 9.57 | 131 |
| 1 | 2 | 0 | $M_{1,2}$ | 0.65 | 11.14 | 0 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.82 | 13.40 | 4 |

Table 4.5: Results of algorithm $OA\_1$ with models (mod=1) of one image (set $S^2$).

and

$$M_{2,1} \equiv \{1\_8, 17, 30\_31, 34\_44, 48\_54, 58\_66, 72, 84\_88, 93\_97\}$$
$$M_{2,2} \equiv \{1\_8, 17\_21, 32, 38, 48\_53, 57\_58, 76, 84\_86, 91\_98\}.$$

An overall analysis of the results given in the tables allows to state that the overlapping algorithm is effective in identifying the images of a each family. Furthermore, the results highlight the features of each setting taken into account. We summarize the main points in the following remarks

1. the use of overlapping $index\_1$ does produce any clear advantage with respect to that of $index\_2$,

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | $M_{2,1}$ | 0.97 | 4.44 | 8 |
| 0 | 1 | 40 | $M_{2,1}$ | 0.94 | 5.33 | 232 |
| 0 | 2 | 0 | $M_{2,2}$ | 0.96 | 8.44 | 0 |
| 0 | 2 | 40 | $M_{2,2}$ | 0.85 | 15.58 | 8 |
| 1 | 1 | 0 | $M_{2,1}$ | 0.86 | 6.07 | 8 |
| 1 | 1 | 40 | $M_{2,1}$ | 0.92 | 4.72 | 232 |
| 1 | 2 | 0 | $M_{2,2}$ | 0.80 | 11.91 | 0 |
| 1 | 2 | 40 | $M_{2,2}$ | 0.84 | 14.00 | 6 |

Table 4.6: Results of algorithm $OA\_1$ with models (mod=2) of up to two images (set $S^2$).

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 1 | 40 | $M_{1,1}$ | 0.76 | 13.92 | 135 |
| 0 | 2 | 40 | $M_{1,2}$ | 0.73 | 15.83 | 4 |
| 1 | 1 | 40 | $M_{1,1}$ | 0.63 | 22.33 | 138 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.64 | 29.00 | 4 |

Table 4.7: Results for algorithm $OA\_2$ with models (mod=1) of one image (set $S^2$).

but the computer execution time is dramatically favorable to the use of $index\_2$;

2. option A gives usually best results but is computationally expensive;
3. the use of models made up of more images improves considerably the algorithm efficiency.

In order to compare the performance of the algorithms the *precision recall curve* has been used. Precision and recall are two classical performance evaluation metrics in the field of information retrieval. As described above results of the comparison of shapes are sorted in ascending order and stored in a vector. We have obtained a ranking of database images based on similarity with a given model; a particular item in the ranking is considered as relevant if the image belongs to the class of the model, otherwise it is not relevant. Building a precision-recall curve requires another important parameter, called *scope*. It is defined as the number of retrieved items from the top of the ranking list. We have set a scope size value equal to 12 for the set $S^1$ and 11 for the set $S^2$. This choice is due to the fact that in [18] the summary of rank-ordered results is given only for the first 12 and 11 shapes, respectively. For each family, given a sample shape and the relative rank results within the fixed scope size, precision and recall values are calculated. In Fig.9 and Fig.10 the comparative precision recall curves for the set $S^1$ are shown. Each image is referred to OA_1 and OA_2 algorithm, respectively. Both figures display the comparison between performance of [18], our best result and our worst result. In the same way (Fig.11 and Fig.12) the precision-recall curves are plotted for the set $S^2$.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 1 | 40 | $M_{2,1}$ | 0.84 | 7.83 | 240 |
| 0 | 2 | 40 | $M_{2,2}$ | 0.82 | 10.24 | 6 |
| 1 | 1 | 40 | $M_{2,1}$ | 0.70 | 16.83 | 242 |
| 1 | 2 | 40 | $M_{2,2}$ | 0.71 | 20.86 | 8 |

Table 4.8: Results for algorithm $OA\_2$ with models (mod=2) of up to two images (set $S^2$).

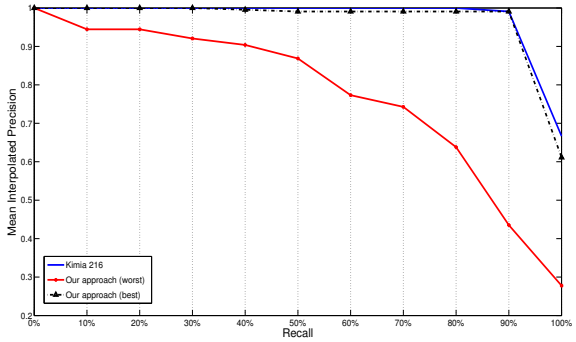


Figure 9: Precision-recall curve for algorithm OA_1 and set $S^1$.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 2 | 0 | $M_{1,2}$ | 0.86 | 6.54 | 0.29 |
| 0 | 2 | 40 | $M_{1,2}$ | 0.88 | 6.44 | 3.00 |
| 1 | 2 | 0 | $M_{1,2}$ | 0.77 | 16.71 | 0.31 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.87 | 5.54 | 3.6 |

Table 4.9: Results of algorithm $OA\_1$ with models of one image (set $S^1$) created automatically.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 2 | 0 | $M_{2,2}$ | 0.96 | 8.22 | 0.35 |
| 0 | 2 | 40 | $M_{2,2}$ | 0.94 | 6.64 | 4.11 |
| 1 | 2 | 0 | $M_{2,2}$ | 0.85 | 18.71 | 0.37 |
| 1 | 2 | 40 | $M_{2,2}$ | 0.93 | 5.87 | 4.59 |

Table 4.10: Results of algorithm $OA\_1$ with models of up to two images (set $S^1$) created automatically.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 2 | 40 | $M_{1,2}$ | 0.85 | 17.07 | 3.37 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.77 | 38.35 | 3.43 |

Table 4.11: Results of algorithm $OA\_2$ with models of one image (set $S^1$) created automatically.

In the showed experiments the image models have been created manually. We have repeated the experiments by using our automatic method for dom-

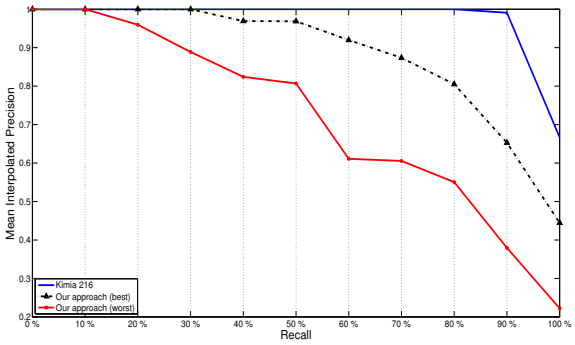


Figure 10: Precision-recall curve for algorithm OA_2 and set $S^1$.

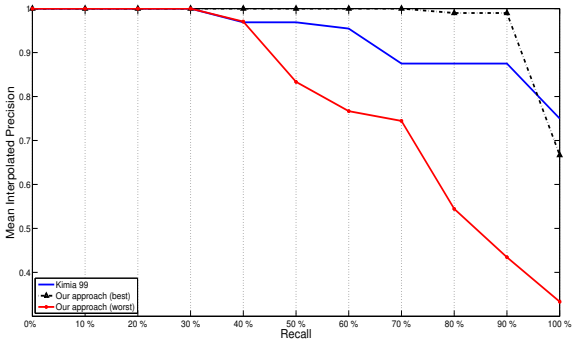


Figure 11: Precision-recall curve for algorithm OA_1 and set $S^2$.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 2 | 40 | $M_{1,2}$ | 0.87 | 9.76 | 5.15 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.82 | 19.92 | 5.34 |

Table 4.12: Results of algorithm $OA\_2$ with models of one image (set $S^1$) created automatically.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 2 | 0 | $M_{1,2}$ | 0.82 | 9.56 | 0.8 |
| 0 | 2 | 40 | $M_{1,2}$ | 0.77 | 9.74 | 0.47 |
| 1 | 2 | 0 | $M_{1,2}$ | 0.74 | 11.55 | 0.7 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.77 | 9.89 | 0.49 |

Table 4.13: Results of algorithm $OA\_1$ with models of one image (set $S^2$) created automatically.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 2 | 0 | $M_{2,2}$ | 0.91 | 6.89 | 0.9 |
| 0 | 2 | 40 | $M_{2,2}$ | 0.89 | 9.91 | 1.11 |
| 1 | 2 | 0 | $M_{2,2}$ | 0.87 | 6.56 | 0.9 |
| 1 | 2 | 40 | $M_{2,2}$ | 0.89 | 6.66 | 1.13 |

Table 4.14: Results of algorithm $OA\_1$ with models of up to two images (set $S^2$) created automatically.

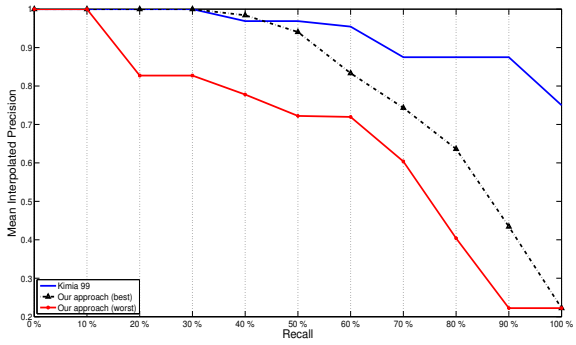


Figure 12: Precision-recall curve for algorithm OA_2 and set $S^2$.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 2 | 40 | $M_{1,2}$ | 0.80 | 12.30 | 0.57 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.66 | 24.07 | 1.0 |

Table 4.15: Results of algorithm $OA\_2$ with models of one image (set $S^2$) created automatically.

| transf | overl. index | rot_no Opt. A | model set | acc_1 | acc_2 | time min. |
|---|---|---|---|---|---|---|
| 0 | 2 | 40 | $M_{1,2}$ | 0.77 | 9.55 | 1.30 |
| 1 | 2 | 40 | $M_{1,2}$ | 0.69 | 22.45 | 1.52 |

Table 4.16: Results of algorithm $OA\_2$ with models of one image (set $S^2$) created automatically.

inant point detection as described in subsection 3.1 for creating the approximated contours. The numerical results are reported in tables 4.9-4.12 for the first database ($S^1$) and in tables 4.13-4.16 for the second database ($S^2$). The overlapping index is $ind_2$.

# 5 Conclusions

In this paper we have proposed a new recognition method based on a curve alignment technique. The method consists of various phases including extracting outlines of images, detecting significant points and aligning curves. The dominant points can be manually detected or by using a very fast and efficient approach described in subsection 3.1. The last phase uses the idea of calculating the overlapping indices between shapes as similarity measures. The proposed algorithm has given satisfying results in terms of accuracy, although a reasonable performance degradation in the $OA\_2$ variation, compared to similar approaches. Such results justify the computational effort required to shapes matching. Also, by using an automatic method for creating approximated models we achieve a very impressive improvement in terms of required computation in spite of a negligible lack

in accuracy in few circumstances.

*References:*

[1] N.J. Ayache and O.D. Faugeras, "HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.8, no.1, pp.44-54, 1986.

[2] E.M. Arkin, L.P. Chew, D.P. Huttenlocher, K. Kedem, J.S.B. Mitchell, "An Efficiently Computable Metric for Comparing Polygonal Shapes", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.13, no.3, pp.209-216, 1991.

[3] R. Basri, L. Costa, D. Geiger, and D. Jacobs, "Determining the Similarity of Deformable Shapes", *Vision Research*, vol.38, pp.2365-2385, 1998.

[4] S. Belongie and J. Malik, "Matching with Shape Contexts", *IEEE Workshop on Content-based Access of Image and Video Libraries*, 2000.

[5] S. Belongie, J. Malik, and J. Puzicha, "Matching Shapes", in *Proc. Intl. Conf. Computer Vision (ICCV01)* pp.454-461.

[6] S.-G. Chen and M.-H. Chi and J.-Y. Wu, "Curvature estimation and curvature flow for digital curves", *WSEAS Transactions on Computers*, vol.5, pp.804-809, 2006.

[7] H. Chui and A. Rangarajan, "A New Algorithm for Non-rigid Point Matching", *IEEE Conference on Computer Vision and Pattern Recognition*, 2000, II:pp.44-51.

[8] D. DeTurck, H. Gluck, D. Pomerleano, D. S. Vick, "The four vertex theorem and its converse", *Notice of the AMS*, vol.54, n.2, pp. 192-207, Feb. 2007

[9] C. Di Ruberto, "Recognition of Shapes by Attributed Skeletal Graphs", *Pattern Recognition*, vol. 37, pp.21-31, 2004.

[10] C. Di Ruberto and A. Morgera, "A fast iterative method for dominant points detection of digital curves", *8th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases (AIKED'09)*, Cambridge, UK, February 21-23, 2009, pp.271-278.

[11] M. Ghanbari and A.M.G. Pinheiro, "Scale space contour approximation for shape similarity evaluation", *WSEAS Transactions on Computers*, vol.3, no.3, pp.819-824, 2004.

[12] Y. Gdalyahu and D. Weinshall, "Flexible Syntactic Matching of Curves and its Application to Automatic Hierarchical Classification of Silhouettes", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.21, no.12, pp.1312-1328, 1999.

[13] S. Gold and A. Rangarajan, "A Graduated Assignment Algorithm for Graph Matching", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.18, no.4, pp.377-388, 1996.

[14] L. He and H. Liu, "Shape context for image understanding", *WSEAS Transactions on Information Science and Applications*, vol.2, no.11, pp.1846-1853, 2005.

[15] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing Images using the Hausdorff Distance", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.15, pp.850-863, 1993.

[16] E. Milios and E.G.M. Petrakis, "Shape Retrieval based on Dynamic Programming", *IEEE Trans. on Image Processing*, vol.9, no.1, pp.141-146, 2000.

[17] T.B. Sebastian and B.B. Kimia, "Curves vs. Skeletons in Object Recognition", *Signal Processing*, vol.85, no.2 pp.247-263, 2005.

[18] T.B. Sebastian, P.N. Klein, and B.B. Kimia, "Recognition of Shapes by Editing Shock Graphs", in *Proceedings of the Eighth International Conference on Computer Vision*, Vancouver, Canada, July 9-12 2001, pp.1755-762, IEEE Computer Society Press.

[19] T.B. Sebastian, P.N. Klein, and B.B. Kimia, "On Aligning Curves", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.25, no.1, pp.116-125, 2003.

[20] K. Siddiqi, A. Shokoufandeh, S.J. Dickinson, and S.W. Zucker, "Shock Graphs and Shape Matching", *Int. J. of Computer Vision*, vol.35, no.1, pp.13-32, 1999.

[21] C.C. Tappert, "Cursive Script Recognition by Elastic Matching", *IBM J. Research Development*, vol.26, no.6, pp.765-771, 1982.

[22] C.H. Teh and R.T. Chin. "On the detection of dominant points on digital curves", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.11, n.8, pp.859-871, 1989.

[23] B. Wirtz, "Average Prototypes for Stroke-Based Signature Verification", *Proc. Int. Conf. Document Analysis and Recognition*, Ulm, Germany, August 1997, pp.268-272.

[24] L. Younes, "Computable Elastic Distance between Shapes", *SIAM Journal of Applied Mathematics*, vol. 58, pp.565-586, 1998.