Multimedia Systems 7: 467–476 (1999)

Multimedia Systems

Curvature scale space image in shape similarity retrieval

Sadegh Abbasi, Farzin Mokhtarian, Josef Kittler

Centre for Vision Speech and Signal Processing, Department of Electronic & Electrical Engineering, University of Surrey, Guildford GU2 5XH, UK; e-mail: {S.Abbasi,F.Mokhtarian,J.Kittler}@surrey.ac.uk

Abstract. In many applications, the user of an image database system points to an image, and wishes to retrieve similar images from the database. Computer vision researchers aim to capture image information in feature vectors which describe shape, texture and color properties of the image. These vectors are indexed or compared to one another during query processing to find images from the database. This paper is concerned with the problem of shape similarity retrieval in image databases. Curvature scale space (CSS) image representation along with a small number of global parameters are used for this purpose. The CSS image consists of several arch-shape contours representing the inflection points of the shape as it is smoothed. The maxima of these contours are used to represent a shape. The method is then tested on a database of 1100 images of marine creatures. A classified subset of this database is used to evaluate the method and compare it with other methods. The results show the promising performance of the method and its superiority over Fourier descriptors and moment invariants.

Key words: Multi-scale analysis – Shape similarity – Curvature scale space – Image database retrieval – Performance characterisation

1 Introduction

Advances in memory technologies and processing speed have made it feasible to store a large number of images in computers. This has given rise to the problem of organising them for a rapid access to their content. An *image database system* aims to help people in this regard and enable them to find their desired images as quickly as possible. In many applications such as medicine, journalism, advertising and fashion, the user of image database remembers something about the content of his desired image or wishes to find similar images to an existing image. In content-based image database systems, intrinsic properties of images are captured in some feature vectors which are indexed or compared to one another during query processing to find similar images from the database.

Considerable amount of information exists in two dimensional boundaries of objects which enables us to recognise objects without using further information. As a result, shape similarity retrieval plays an important role in content based image database systems.

Different shape representation methods are employed for shape similarity retrieval. In elastic (Del Bimbo et al. 1996) and modal (Sclaroff 1996) matchings, it is assumed that each object of the database is the deformed version of the query. The similarity between the two is then measured without introducing an explicit representation for the shape. These approaches are computationally expensive. A remedy is introduced in (Sclaroff 1996) by choosing a few prototype shapes as the representatives of different categories of the database. A sketch-based method which uses an abstract of the image edges is presented in (Hirata and Kato 1993). In another approach (Eggleston 1992), a set of shape features like centroid, pixel count, percentage of bounding rectangle fill, number of holes, perimeter etc.; are used to represent an object. Since local information is vital for shape similarity retrieval, it seems that representing an object with a set of global parameters does not lead to very good results. In the QBIC project (Niblack et al. 1993), shape features are based on a combination of heuristic shape features such as area, circularity, eccentricity, major axis orientation and a set of algebraic moment invariants.

In conclusion, although the number of proposed methods is increasing rapidly, there are still a number of shortcomings associated with each method. While the robustness of some methods is doubtful, other methods which exhibit a reasonable degree of robustness are often computationally expensive. In this paper, we introduce an efficient and robust shape representation method for shape similarity retrieval. The maxima of curvature scale space (CSS) image contours together with a small number of global parameters are used to represent a shape. The properties of the method are reviewed and the results of our experiments on a database of 1100 images of marine animals are presented. A comparison between the performance of the method with two other well-

Correspondence to: S. Abbasi

known methods, Fourier descriptors and moment invariants, are also presented.

The following is the organisation of the remainder of this paper. In Sect. 2, the construction of the CSS image and the properties of using its maxima as the shape representation are explained. Section 3 is devoted to the CSS matching algorithm. In Sect. 4, we explain the way we use the global parameters prior to the CSS matching. In Sect. 5, special factors which must be considered in using the CSS image maxima as the shape representation are presented. Section 6 is concerned with the problem of evaluation of the performance of the method which is carried out through a set of classified shapes. The experimental results including the effects of global parameters are presented in Sect. 7. Section 8 deals with the results of comparison of the method with Fourier descriptors and moment invariants. And finally, Sect. 9 offers concluding remarks.

2 The CCS representation

2.1 Curvature

The curvature of a curve is defined as

$$\kappa(s) = \lim_{h \to 0} \frac{\phi}{h} \ ,$$

where ϕ is the angle between $\mathbf{t}(s)$ and $\mathbf{t}(s + h)$. $\mathbf{t}$ represents the tangent vector and s is the arc length parameter. *Curvature-zero crossings* of a curve are points where the sign of curvature changes.

Consider a parametric vector equation for a curve:

$$\boldsymbol{\Gamma}(u) = (x(u), y(u)) \ ,$$

where u is an arbitrary parameter. The formula for computing the curvature function can be expressed as

$$\kappa(u) = \frac{\dot{x}(u)\ddot{y}(u) - \ddot{x}(u)\dot{y}(u)}{(\dot{x}^2(u) + \dot{y}^2(u))^{3/2}} \quad . \tag{1}$$

If we convolve each component of $\boldsymbol{\Gamma}$ with $g(u, \sigma)$, a 1D Gaussian kernel of width σ, then $X(u, \sigma)$ and $Y(u, \sigma)$ represent the components of the resulting curve, $\boldsymbol{\Gamma}_\sigma$:

$$X(u, \sigma) = x(u) \star g(u, \sigma)$$

$$Y(u, \sigma) = y(u) \star g(u, \sigma).$$

According to the properties of convolution, the derivatives of every component can be calculated easily:

$$X_u(u, \sigma) = x(u) \star g_u(u, \sigma) \ ,$$

$$X_{uu}(u, \sigma) = x(u) \star g_{uu}(u, \sigma) \ ,$$

and we will have a similar formula for $Y_u(u, \sigma)$ and $Y_{uu}(u, \sigma)$. Since the exact forms of $g_u(u, \sigma)$ and $g_{uu}(u, \sigma)$ are known, the curvature on $\boldsymbol{\Gamma}_\sigma$ can be computed easily:

$$\kappa(u, \sigma) = \frac{X_u(u, \sigma)Y_{uu}(u, \sigma) - X_{uu}(u, \sigma)Y_u(u, \sigma)}{(X_u(u, \sigma)^2 + Y_u(u, \sigma)^2)^{3/2}} \ . \tag{2}$$

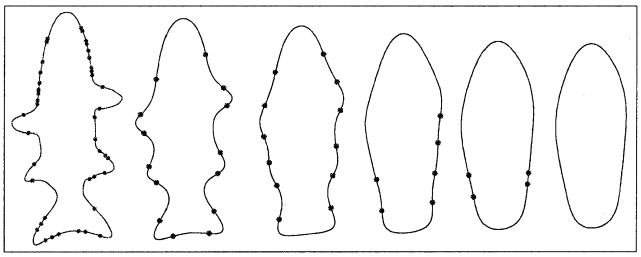


Fig. 1. Shrinkage and smoothing of the curve and decreasing of the number of curvature zero-crossings during the evolution, from left: σ = 1, 4, 7, 10, 12, 14

2.2 The CSS image

Following the preprocessing stage, every object is represented by the x and y coordinates of its boundary points. The number of these points varies from 400 to 1200 for images in our prototype databases. To normalise the arc length, the boundary is resampled and represented by 200 equally distant points. The curve is then smoothed by a Gaussian function. The smoothed curve is called $\boldsymbol{\Gamma}_\sigma$, where σ denotes the width of the Gaussian kernel, $g(u, \sigma)$. The locations of curvature zero-crossings on $\boldsymbol{\Gamma}_\sigma$ are determined at different levels of scale using Eq. 2. The process starts with σ = 1, and at each level, σ is increased by $\Delta\sigma$, chosen as 0.1 in our experiments. As σ increases, $\boldsymbol{\Gamma}_\sigma$ shrinks and becomes smoother, and the number of curvature zero crossing points on it decreases. Finally, when σ is sufficiently high, $\boldsymbol{\Gamma}_\sigma$ will be a convex curve with no curvature zero-crossings (see Fig. 1). The process of creating ordered sequences of curves is referred to as the *evolution* of $\boldsymbol{\Gamma}$.

If we determine the locations of curvature zero-crossings of every $\boldsymbol{\Gamma}_\sigma$ during evolution, we can display the resulting points in (u, σ) plane, where u is the normalised arc length and σ is the width of the Gaussian kernel. The result of this process can be represented as a binary image called the *CSS image* of the curve (see Fig. 2a). The intersection of every horizontal line with the contours in this image indicates the locations of curvature zero-crossings on the corresponding evolved curve. For example, by drawing a horizontal line at σ = 10.0, it is observed that there are 6 zero-crossing points on $\boldsymbol{\Gamma}_{10}$. These points can also be found on the boundary of object in Fig. 1 for σ = 10.

As seen in Fig. 1, there are two curvature zero-crossings on every concave or convex part of the shape, and as the curve becomes smoother, these points approach each other and create a contour in the CSS image of the shape. When the segment is filled, the two points join and represent the maximum of the relevant contour. The height of this contour then reflects the depth and size of the concavity or convexity. The deeper and larger the segment, the higher the maximum. In other words, a contour maximum in the CSS image, represents a segment of the shape.

2.3 Extracting maxima of CSS contours

We represent every image in the database with the locations of its CSS contour maxima. For example, in Fig. 2 there are seven maxima, and therefore the image will be represented

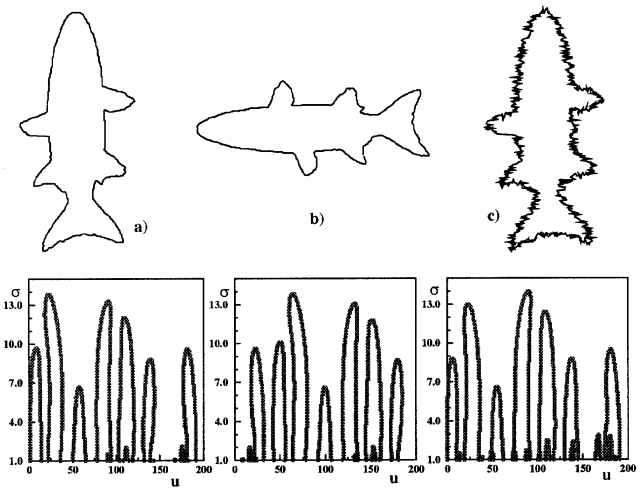


Fig. 2. a A boundary and its CSS image. **b** Change in orientation causes a circular shift in CSS image. **c** Noise creates small contours in CSS image

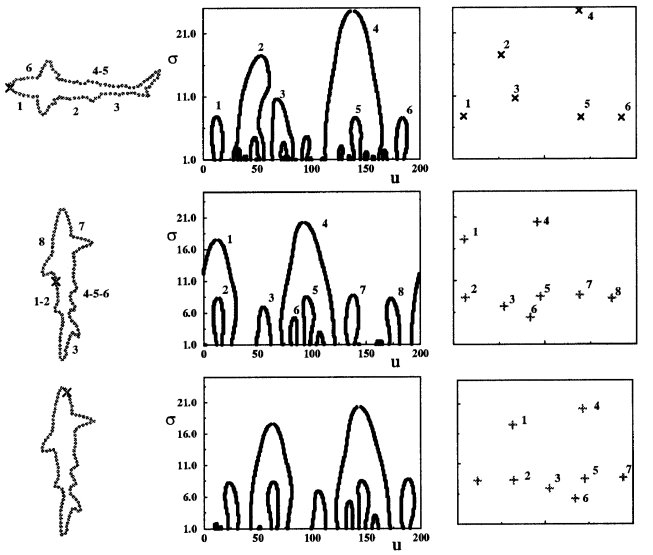


Fig. 3. CSS image and its maxima, *left*: re-sampled boundary with the marked starting point, *middle*: CSS image, *right*: normalised maxima of CSS images

by seven pairs of integer numbers. The locations of maxima are not readily available and must be extracted from the image (Mokhtarian et al. 1996a). The CSS contours are usually connected everywhere except in a neighbourhood of their maxima. We find the peaks of both branches of a contour in the CSS image and consider the midpoint of the line segment joining the pair as a maximum of the CSS image.

Small contours of the CSS image are related to noise or small ripples of the curve. In order to avoid complicated and inefficient matching, small maxima are not included in the representation. In most cases, even if small contours are considered, they do not play a major role in the final matching value between an input shape and a model from the database. In our system, if a maximum is less than $\frac{1}{6}$ of the largest maximum of the same CSS image, it is considered as noise. As a result, only major concavities and convexities of a shape will contribute to the representation.

2.4 Properties of the CSS image

The CSS representation is robust with respect to scale, noise and change in orientation. A rotation of the object usually causes a circular shift on its representation, which is easily determined during the matching process (compare Fig. 2a and b). Note that the effect of a change in the starting point is also the same. Due to arc length normalisation, scaling does not change the representation, and as Fig. 2 shows, noise may create some small contours on the CSS image, but the main contours and therefore the corresponding maxima remain unaffected.

Compactness is another aspect of the CSS representation. A shape is represented by about less than ten pairs of integer values which can be determined without any ambiguity. The matching algorithm which compares two sets of representations and assigns a match value as the measure of similarity between the shapes is also simple and fast.

Another property of the CSS image is that it retains the local properties of the shape. Every contour of the CSS image corresponds to a concavity or a convexity of the shape. A local deformation of the shape mainly causes a change in the corresponding contour of the CSS image. Using this property, one can include more local information about the shape in the CSS image. For example, it is possible to use the average curvature of a segment together with the maximum of its CSS contour. Alternatively, one can find the corresponding corners of each pair of curvature zero-crossings.

In shape similarity retrieval, the task is not to accurately recognise the input among the existing models, but it is to find the most similar models to the input and rank them in terms of a similarity measure. The CSS representation is a *reliable* tool to handle this task.

3 CSS matching

As mentioned before, every object in the database is represented by the locations of the maxima of its CSS image. In this section, we first explain the basic concepts of our matching algorithm, which compares two sets of maxima and assigns a matching value to them. The matching value represents the similarity measure between the actual boundaries of objects. A more complete description of the CSS matching algorithm then follows.

Each contour of the CSS image corresponds to a concavity or a convexity of the relevant object as presented in Fig. 3. It is obvious that regions 6 and 1 of the first object must be matched with regions 7 and 8 of the second object, respectively. Looking at the locations of the relevant maxima on the first and second row of Fig. 3, we realize that they are in quite different positions. This is due to different starting points. If we change the starting points properly, then the locations of corresponding maxima on CSS images will be close to each other. This can be observed on the third row of Fig. 3.

Therefore, the first step in CSS matching is to shift one of the two sets of maxima so that the effect of randomly selected starting point is compensated. Since the exact value of required shift is not available, we choose several values

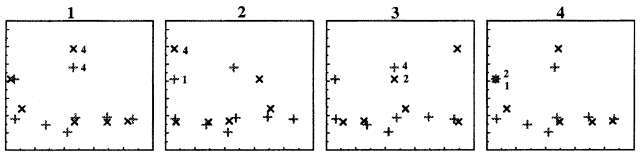


Fig. 4. Four possible choices for matching of the two sets of maxima related to first and second rows of the previous figure

for it and then find the best match among them. The best choice is a value that shifts one CSS image so that its major maximum covers the major maximum of the other CSS image. Other possible choices are those values which accomplish the same with the second and possibly the third major maxima.

For the two sets of maxima shown in Fig. 3, four choices are shown in Fig. 4. Considering this figure, one can quickly realize that the first one is the best. Every maximum of the first CSS image is matched with a maximum of the second one, and two maxima remain unmatched. The matching value will be the summation of the the straight-line distances between the matched pairs plus the vertical coordinates of the unmatched maxima.

For convenience, from now on, we call the input *image* and the images in the database *models*. The maxima of every model are sorted according to their σ-coordinates during the process of maxima extraction.

It should be noted that, in addition to the CSS maxima, we also use several *global parameters* to discard dissimilar shapes, prior to the CSS matching. These parameters and the way we use them are described in Sect. 4.

The complete matching algorithm which compares the two sets of maxima, one from the image and the other from the model is as follows.

1. Create a node consisting of the largest scale maximum of the image and the largest scale maximum of the model. Initialise the *cost* of this node to the absolute difference of σ-coordinates of the image and the model. Compute a CSS shift parameter α for each node:

   $$\alpha = U_m - U_i \ ,$$

   where U is the horizontal coordinate of a maximum, and i and m refer to image and model, respectively. This parameter is used to compensate the effect of different start points or change in orientation.
2. If there are more than one maximum in the model which have a σ-coordinate close (within 80%) to the largest scale maximum of the image, create extra nodes consisting of the largest scale maximum of the image and that respective additional maximum of the model. Also create the same nodes for the second largest scale maximum of the image and the respective maxima of the model. Initialise the cost and compute the CSS shift parameter for each node accordingly.
3. Create two lists for each node obtained in steps 1 and 2. The first list will contain the image curve maxima and the second list will contain the model curve maxima matched within that node at any point of the matching procedure. Initialise the first and second list of each node by the corresponding maxima determined in the first two steps.
4. Expand each node created in steps 1 and 2 using the procedure described in step 5.
5. To expand a node, select the largest scale image curve CSS maximum (which is not in the first list) and apply that node's shift parameter α to map that maximum to the model CSS image. Locate the nearest model curve CSS maximum (which is not in the second list). If the two maxima are in a reasonable horizontal distance (0.2 of the maximum possible distance), define the cost of the match as the straight-line distance between the two maxima. Otherwise, define the height of the image curve CSS maximum as the cost of the match.
   If there are no more image curve CSS maxima left, define the cost of match as the height of the highest model curve CSS maximum *not* in the node's second list. Likewise, if there are no more model curve CSS maxima left, define the cost of match as the height of the selected image curve maximum. Note that this cases may occur when the number of image maxima is *different* from the number of model maxima.
   Add the match cost to the node cost. Update the two lists associated with the node.
6. Select the lowest cost node. If there are no more model or image curve CSS maxima that remain unmatched within that node, then return that node as the lowest cost node. Otherwise, go to step 5 and expand the lowest cost node.
7. Reverse the place of the image and the model and repeat steps 1 to 6 to find the lowest cost node in this case.
8. Consider the lowest node as the final matching cost between the image and the model.

Using this algorithm and considering its amendment which follows immediately in Sect. 3.1, the system associates a matching value to every candidate and then displays the n best matched as its output, where n has already been selected by the user.

3.1 The problem of mirror-images in CSS matching

If a model in the database is similar to the mirror-image of the input, the CSS image of the model may also be similar to the mirror-image of the CSS image of the input. Since, by just a circular shift, it is not possible to map the corresponding maxima in this case, the above mentioned algorithm will fail to discover the similarity between the input and the model. Therefore, the mirror-image of the input should also be compared to the existing models of the database. Using the input maxima, we can easily calculate a new set of maxima which belongs to the CSS image of the mirror-image of the input. We can then either repeat steps 1 to 8 for the new set and consider the lowest matching cost between the two, or construct new nodes in step 1 for the new set and expand all nodes simultaneously.

4 Global parameters

The matching algorithm is not applied to all models of the database. We use a number of global parameters to reject dissimilar shapes to the input prior to the matching process.

To reject the dissimilar images based on the global parameters, we first calculate α_e, α_c and α_a as follows:

$$\alpha_e = \frac{|\, e_i - e_m \,|}{max(e_i, e_m)} \quad \alpha_c = \frac{|\, c_i - c_m \,|}{max(c_i, c_m)}$$

$$\alpha_a = \frac{|\, a_i - a_m \,|}{max(a_i, a_m)}$$

where *e* and *c* represent the eccentricity and circularity of the boundary and *a* represents the aspect ratio of the CSS image, while *i* and *m* stand for image and model, respectively.

According to their definition, α_e, α_c and α_a are between zero and one. We need to choose a threshold for each of these parameters so that, if one of them is above the relevant threshold, the corresponding model is rejected. For our system, we chose all threshold values, α_{et}, α_{ct} and α_{at} as 0.3. Lower threshold values may result in missing similar shapes, while , with higher values, the number of candidates increases, and using global parameters will become senseless. However, it should be noted that the performance of the system is not sensitive to small changes of these values, and this will be demonstrated later in Sect. 7.2.

5 Comments on the CSS representations

As the results of our experiments show, the proposed method has shown a very good performance in shape similarity retrieval. In this section, we explain some aspects of the representation that should be considered in this application of the CSS image.

Global information. In Subsect. 8.1, we mention that the largest magnitude pair of components (F_1 and F_{-1}) of Fourier descriptors represent an ellipse quite similar to the outline of the contour. Such information does not exist in the CSS representation, which consists of the maxima of the CSS image contours. As a result, two shapes with the same local deformations will have the same representations, even if one of them looks like a circle and the other looks like a rectangle. This is the main reason for using global parameters along with the maxima of the CSS image to represent a shape.

At the same time, it should be noted that one of these parameters is the aspect ratio of the CSS image. This, in turn, indicates that there is some implicit global information in the CSS image maxima.

The problem of shallow concavities. Since the process of smoothing can be approximated by heat equation deformation (Kimia and Siddiqi 1996), it can be shown that the shallow concavities can create the same large contours as the deep ones in the CSS image. Therefore, a shallow concavity may be matched with a deep one during the CSS matching. The problem can be solved by adding more information to the CSS image maxima (Abbasi et al. 1997).

Normalisation of rotation and starting point. The normalisation is applied by a circular shift to the maxima of the CSS image during the CSS matching. The measure of this shift is determined by the horizontal distance between the highest maximum of one CSS and the highest or the second highest maximum of the other one. This is symbolised by α in item 1 of the matching algorithm. Applying a circular shift to the CSS maxima is equivalent to a change in orientation of the curve. In fact, we change the orientation of one curve, so that the major segments of the two curves cover each other.

Due to difference in shape of major segments, it may happen that the best circular shift which matches the two objects cannot be achieved by this method. As a result, the matching value may be more than expectation, and this affects the output ranking. A remedy is to generate more nodes with slightly different shift parameters. However, if the number of nodes is too large, the efficiency of the matching algorithm and the speed of the system are affected. We have slightly increased the number of nodes and have achieved better results.

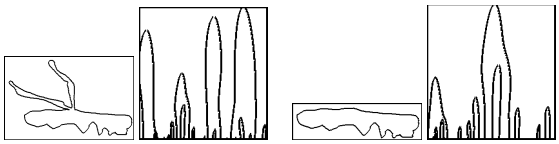


Fig. 5a–d. Occlusion may cause a change in the size of the remaining CSS contour

Occlusion. Minor occlusions do not dramatically change the CSS image of a shape and can be detected by the matching algorithm. Particularly, when major concavities of the shape remain intact. However, if several major segments of a shape are covered, the relevant contours in the CSS image will disappear. As a result, the remaining contours occupy a larger portion of the CSS image and their corresponding maxima will also be larger. An example is shown in Fig. 5. The first three maxima of the CSS image in Fig. 5b are related to the upper part of the shape in Fig. 5a. These maxima do not exist in the CSS image presented in Fig. 5d, due to covering the upper part of the shape. This part is presented by a small contour at the extreme left of the CSS image in Fig. 5d. It is observed that the configurations of the similar contours of the two CSS images are the same. The similar part consists of two larger contours, one inside the other, accompanied by one smaller contour at each side. However, the maxima in Fig. 5d are significantly larger than the maxima in Fig. 5b. This is due to the fact that a segment in Fig. 5c is represented by larger number of samples in comparison with the same segment in Fig. 5a.

The problem of noise, and ripples. As mentioned before, small contours in the CSS image which are not considered in representation, are related to ripples on the boundary of shape. As a result, two shapes with the same sets of concavities will have the same representations, even if one of them contains small ripples. In other words, the ripples are considered as noise, even if they carry some useful information. Circularity is the ratio of perimeter squared to the area. It is used to distinguish between rippled boundaries and smoothed ones, If two shapes are similar, but one of them includes ripples, then the area of both shapes will be in the same range, whereas the perimeters will be different.

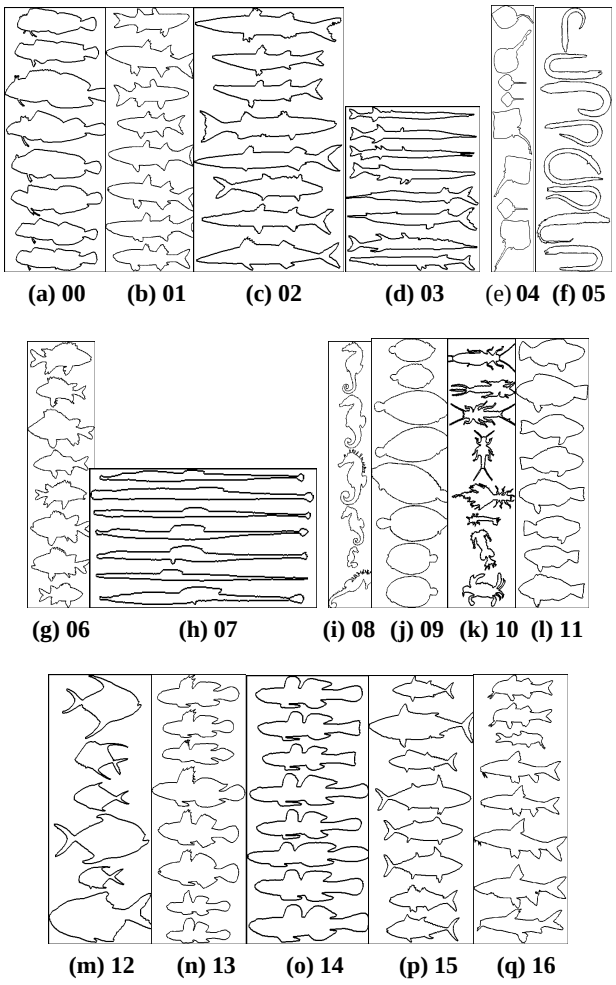


Fig. 6a–q. Classified database used for objective evaluation

6 Performance evaluation

Shape similarity retrieval is involved with the notion of similarity which cannot be measured. As a result, the evaluation of the system performance turns out to be a difficult task. A subjective evaluation involving human subjects is presented in (Mokhtarian et al. 1996b). The subjects are asked to find similar shapes to a number of queries from a small database. The results are then compared to the results of the system.

Here, we present an objective evaluation, which involves a small classified subset of our database. There are 17 classes in this database, each consisting of about 8 objects. The whole database is presented in Fig. 6. These objects are selected carefully so that the within-class similarity is reasonably high. There are also particular characteristics in each group to distinguish it from other groups. We have made every effort to perform a fair classification, and have paid more attention to the whole appearance of shapes, rather than taking into account the shape features which are used by our method. We also believe that our classified database provides a good base to compare the performance of different methods we have tried.

The procedure of evaluating and marking the performance of the system is as follows.

- Choose one of the objects in class one as the input query, and determine the first n outputs of the system. These are the most similar images of the database to the input according to the system. n = 15 is chosen for this test.
- Count the number of outputs which are in the same class as the input. Divide this number by the number of members of this class, multiply it by 100 and let the result be the performance measure of the system for that particular object.
- Repeat the previous steps for all members of class one. Determine the performance of the system for class one by averaging the performance measure of all members of this class.
- Determine the performance measures for all classes, repeating the above steps.
- Finally, find the performance measure of the system for the whole classified database by averaging the performance measures of all classes.

Using this method, we measure the performance of the system on different approaches and compare them in the following sections.

7 Results

In this section, the results of our experiments are presented and discussed. We start with the original CSS matching, which uses the maxima of the CSS image to represent the shape. It will then be compared to a modified version which includes the mirror-image considerations. These are in Subsect. 7.1. In the first step of the new matching algorithm, we also create more nodes. As a result, the chance for a better normalisation in the starting point and change in orientation increases. We present the results of our experiments on using the CSS representation with the global parameters in shape similarity retrieval in Subsect. 7.2.

7.1 CSS without global parameters

This is the first version of our system and is called *reference method*. We modified the CSS matching (Mokhtarian 1995) by considering additional nodes and solving the problem of mirror-image. Two examples are presented in Fig. 7a and b. In both examples, the input query has appeared as the first output of the system. In Fig. 7a, the seventh output is not as similar to the query as the other outputs. The dissimilarity between the two shapes is mainly due to the global appearance of them. We can deal with this problem by using global parameters. This argument also applies to the seventh output of Fig. 7b.

The results of objective evaluation for this approach is presented in the first row of Table 1. The result for group 04 is 100%. This means that, whenever one of the members of this group is used as the query, all other members appear in the first 15 outputs of the system. The same results are achieved for groups 05, 11 and 14. Apart from groups 03, 07 and 10, the results for other groups are more or less acceptable. Groups 03 and 07 include shapes with shallow concavities (Abbasi et al. 1997), and group 10 consists of strange shapes which are not quite similar to each other.

Table 1. Effects of not using the mirror-image consideration. WM and RM stand for reference method and without mirror-image respectively. The *third row* shows the difference in percent caused by cancelling the mirror-image consideration

G	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	T
RM	88	86	59	31	100	100	78	33	72	95	31	100	78	89	100	86	59	76
WM	78	53	44	30	98	100	44	35	69	95	25	61	78	69	86	72	52	64
dif	-10	-33	-15	-1	-2	0	-34	2	-3	0	-6	-39	0	-20	-14	-14	-7	-12

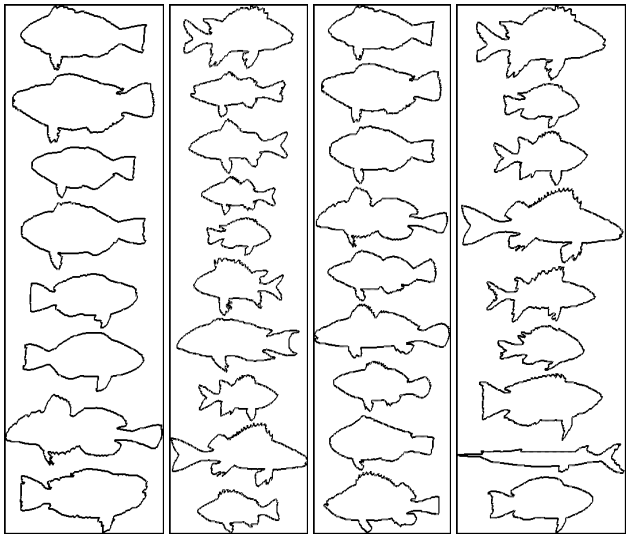


Fig. 7. a and **b** reference method. **c** and **d** without mirror-image consideration

However, considering objects of other groups, we believe that these objects can be considered as a group.

Note that, for this test, we have not used any global parameters except the aspect ratio of the CSS images.

7.1.1 CSS without mirror-image

The problem of mirror-image in CSS matching is explained in Sect. 3.1. Here we repeat the queries of Fig. 7a and b by disabling the mirror-image consideration. The results are presented in Fig. 7c and d, respectively. It is observed that the orientation of outputs is the same as that of the input queries in these cases. Some good models like the first three outputs of Fig. 7b have not appeared in Fig. 7d. The same comment applies to the fifth and the sixth output of Fig. 7a. The problem with the eighth output of this example is related to its shallow concavities described in Sect. 5.

The results of the objective evaluation is presented in the second row of Table 1. A dramatic drop in performance measure is observed when we remove the modifications. This drop is more considerable for groups like 06 and 11, which include shapes with different orientations.

7.2 CSS with global parameters

Eccentricity and circularity contain considerable information about the global shape of an object. When this information is used in conjunction with the maxima of the CSS image, which basically contain local information of the shape, the best results are achieved.

We study the effects of these parameters in two stages. In stage one, we observe the improvement caused by each of these parameters. Then we study the effects of using them together. Using circularity alone may increase the performance measure of the method by up to 7% which is achieved by setting α_{ct} to any value in the wide range of 0.28–0.46. This is shown in Fig. 8a. Note that we have not used eccentricity in this experiment. Also note that, when we choose a value for α_{ct}, it is fixed for all images of the database.

Since the aspect ratio of the CSS image was used with the reference method, it is also used in the experiments of this section. This makes the comparison easier and more meaningful. The total performance measure for the CSS matching plus aspect ratio as the global parameter was 76%, as it is shown in Table 1.

The better results are achieved when we use eccentricity alone. Any value for α_{et} in the range of 0.28–0.42 leads to 10% improvement in the performance measure, which is quite considerable. The total performance measure is 86%. Figure 8b represents the results of this experiment.

Now, we choose α_{et} as 0.33 and study the performance of the system by changing the value of α_{ct}. This time, the increase in the performance measure is 14% when we choose α_{ct} between 0.37 and 0.55. This is shown in the plot of Fig. 8c. Note that, for the values in the range of 34%–67%, this figure will be 13%.

If α_{ct} is fixed as 0.4, and α_{et} is chosen and fixed at any value between 0.28 and 0.4, then the same results, ie 14% increase is achieved. This is shown in Fig. 8d.

We can conclude that, by using these three global parameters, the performance measure of the system will be more than 90% in a wide range of threshold values.

8 Comparison with other methods

8.1 Fourier descriptors (FDs)

A closed curve, $\Gamma(t) = (x(t), y(t))$, can be considered as a complex periodic function of t, where $-\infty \leq t \leq +\infty$. This function can then be sampled by N equidistant points. The discrete Fourier transform of $\Gamma(t)$ is then defined as

$$F_k = \sum_{n=0}^{N-1} e^{-j2\pi nk/N} \Gamma_n \quad -N/2+1 \leq k \leq N/2-1 \,,$$

where Γ_n is the nth sample of $\Gamma(t)$.

As stated in Wallace and Wintz (1980), F_1 always has the highest magnitude among F_ks (F_0 is not considered), provided that the contour is traced in the counterclockwise manner and the contour does not cross itself. We will observe that this may not be true in some special cases.

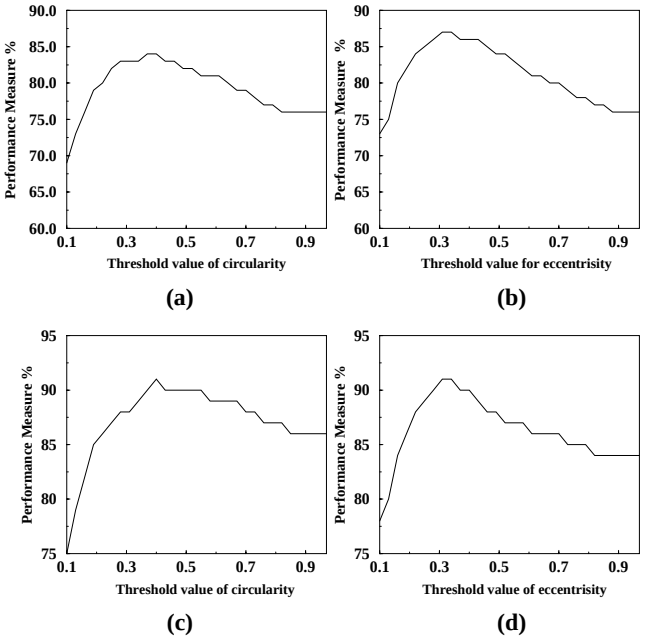


Fig. 8a–d. Effects of using global parameters. **a** Circularity alone, α_{et} = 1. **b** Eccentricity alone, α_{ct} = 1. **c** Both but α_{et} is fixed at 0.33. **d** Both but α_{ct} is fixed at 0.4. Note: For all cases, α_{at} is fixed at 0.25

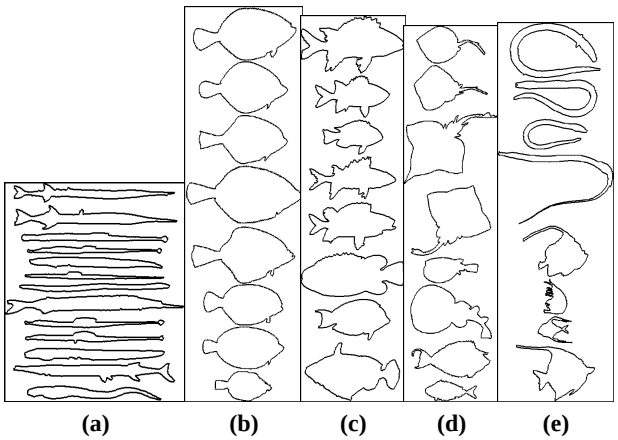


Fig. 9a–e. Examples of query results based on Fourier descriptors method

Provided that the starting point, position, orientation and scale have been chosen properly, the degree of similarity between two sets of FDs can be measured by the sum of Euclidean distances of corresponding components. This will be proportional to the sum of Euclidean distances of the contour points. A method suggested in Wallace and Wintz (1980) is supposed to normalise the FDs. All contours of the database are turned so that the orientation and the starting points of all similar shapes become similar, and therefore the Euclidean distance is minimised.

Applying the inverse Fourier transform, we can recreate a contour from its FDs. In particular, F_1 and F_{-1} create an ellipse which is quite similar to the outline of the contour without any details. Each pair, like F_i and F_{-i}, will create a particular ellipse which is traced i times. The superposition of these ellipses recreates the contour.

Using 20 pairs of FDs of every contour of our database, we employed the method explained in Wallace and Wintz (1980) to normalise them. We then used the Euclidean distance between the FDs of the input query and those of the models to find the most similar shapes.

A number of examples are given in Fig. 9. We can learn the following points from these examples.

- In the first three examples, the input query can be covered by an ellipse which is presented by F_1 and F_{-1}, and therefore the results are globally similar to the input query.
- The main reason for the good results of the second example is that the input query is symmetrical. This is not true for Fig. 9d and e, and therefore the results of these examples are not good enough.
- The problem of mirror-image is not considered in the method. Looking at group 06, we realise that there are several other shapes similar to the mirror-image of the input query of the third example, which have not appeared as outputs.
- For the input of Fig. 9e, F_1 does not have the largest magnitude as claimed in Wallace and Wintz (1980), and therefore the method fails to apply a proper normalisation.

The same conclusions can be made by considering the results of the objective evaluation presented in Table 2, where they are compared to the CSS with global parameters.

We have very good results for groups 00, 02 and 07. On the other hand, very poor results are observed for groups 05, 06, 08, 10, and 11. In group 00, the orientations and the starting points are the same for all contours. This is a result of using the same algorithm for extracting these contours from the original color images. Shapes in group 02 are also symmetric and shapes in group 07 are unique in terms of the largest ellipse. For more complex shapes, like group 08 and shapes with different orientations like groups 06 and 11, we observe a poor performance. For some of the contours of group 05, F_1 does not have the largest magnitude component, and therefore we come across very poor results.

In conclusion, the FD method introduced by Wallace and Wintz (1980) is simple, quite fast, easy to implement and contains good global information of the shape. On the other hand, lack of local support and ambiguities in starting point and orientation are the most important shortcomings of the method. Moreover, we observed that Wallace and Wintz (1980) have not considered the problem of mirror-image and their assumption over F_1 as the component with the largest magnitude among FDs is not always correct.

8.2 Moment invariants

The geometric moment of order $p+q$ of the boundary points of a shape is defined as follows:

$$\mu_{pq} = \sum_x \sum_y x^p y^q \ .$$

The same definition also applies to the points of the solid shape. The original *moment invariants* introduced by Hu

Table 2. FD results in comparison with the CSS + global parameters

G	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	T
CSS	89	86	81	95	100	100	91	98	75	92	81	100	94	89	100	95	69	91
FD	100	84	100	75	78	42	80	100	36	98	13	70	78	98	73	97	48	75
dif	-11	2	-19	20	22	68	9	-2	39	-6	68	30	16	-9	27	-2	21	16

Table 3. Moment invariants results in comparison with the CSS + global parameters

G	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	T
CSS	89	86	81	95	100	100	91	98	75	92	81	100	94	89	100	95	69	91
MI	34	78	91	100	88	41	72	86	47	53	23	91	53	45	72	73	58	65
dif	55	8	-10	-5	12	59	19	12	28	39	58	9	41	43	28	22	11	26

(1962) are seven functions of μ_{pq}, where p and q are between 0 and 3. These functions, called M_1 to M_7, are invariant to rotation, reflection or a combination of them. Scale invariance can be achieved by normalising the functions using *radius of gyration*, which is defined as follows:

$$r = (\mu_{20} + \mu_{02})^{\frac{1}{2}}.$$

This parameter is proportional to the size of object boundary. A set of six normalised moment invariants is obtained as follows:

$$M'_2 = \frac{M_2}{r^4}, \quad M'_3 = \frac{M_3}{r^6}, \quad M'_4 = \frac{M_4}{r^6},$$

$$M'_5 = \frac{M_5}{r^{12}}, \quad M'_6 = \frac{M_6}{r^8}, \quad M'_7 = \frac{M_7}{r^{12}}.$$

We use the latter to represent the boundary as well as solid shape, and experimentally compare this method to our proposed method. Each object is then represented by a 12-dimensional feature vector, including two sets of moment invariants, one from object boundary and the other from solid shape. The Euclidean distance is used to measure the similarity between different shapes.

The results of objective evaluation is presented in Table 3. The best results are for groups 02, 03, 07. These are globally different from the other groups.

It seems that moment invariants may be used to represent some global properties of the shape, but they cannot be used as the shape representation.

9 Conclusion

We introduced a method for shape similarity retrieval from large image databases. The maxima of the CSS image together with a small number of global parameters were used to represent the closed planar shapes. The representation and its associated matching algorithm were explained. The method was tested on a database of 1100 images of marine animals.

With regard to evaluation of the method, we examined an objective test involving a classified subset of the database. We then compared our method with a couple of the most well-known methods in shape representation, namely Fourier descriptors and moment invariants. Different individual examples as well as the objective results showed the superiority of our method over these methods.

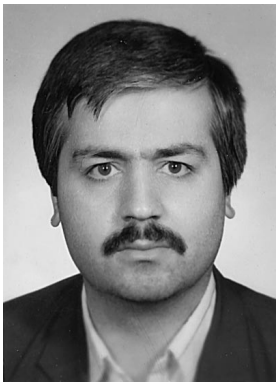
In comparison with other methods discussed in Sect. 1, the CSS representation and its associated matching algorithm provide a clear, straightforward and robust method which can be used in shape similarity retrieval.

A demo of this work is available at the following web-site: http://www.ee.surrey.ac.uk/Research/VSSP/imagedb/demo.html.

Acknowledgements. Sadegh Abbasi is grateful to the Ministry of Culture and Higher Education of Iran for its financial support during his research studies.

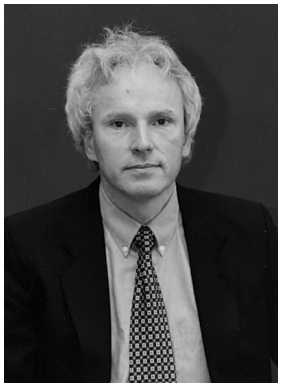
References

1. Abbasi S, Mokhtarian F, Kittler J (1997) Shape similarity retrieval using a height adjusted curvature scale space image. In: Proceedings of 2nd International Conference on Visual Information Systems, 1997, San Diego, CA, USA, pp 173–180
2. Del Bimbo A, Pala P, Santini S (1996) Image retrieval by elastic matching of shapes and image patterns. In: Proceedings of the 1996 International Conference on Multimedia Computing and Systems, 1996, Hiroshima, Japan, IEEE, Los Alamitos, CA, USA, pp 215–218
3. Eggleston P (1992) Constraint-based feature indexing and retrieval for image databases. Proc SPIE 1819: 27–39
4. Hirata K, Kato T (1993) Rough sketch-based image information retrieval. NEC Res Dev 34(2): 263–273
5. Hu MK (1962) Visual pattern recognition by moments invariants. IRE Trans Inf Theory IT-8: 179–187
6. Kimia BB,Siddiqi K (1996) Geometric heat equation and nonlinear diffusion of shapes and images. Comput Vision Image Understanding 64(3): 305-332
7. Mokhtarian F (1995) Silhouette-based isolated object recognition through curvature scale space. IEEE Trans Pattern Anal Mach Intell 17(5): 539–544
8. Mokhtarian F, Abbasi S, Kittler J (1996) Robust and efficient shape indexing through curvature scale space. In: Fisher RB, Trucco E (eds) Proceedings of the 6th British Machine Vision Conference (BMVC), volume 1, 1996, Edinburgh, Scotland, British Machine Vision Association, pp 53–62
9. Mokhtarian F, Abbasi S, Kittler J (1996) Efficient and robust retrieval by shape content through curvature scale space. In: Smeulders AWM, Jain R (eds) Proceedings of the First International Workshop on Image Database and Multimedia Search, 1996, Amsterdam, The Netherlands, Intelligent Sensory Information Systems, pp 35-42
10. Niblack W et al (1993) The QBIC project; querying images by content using color texture and shape. Proc SPIE 1908: 173–187
11. Sclaroff S (1996) Encoding deformable shape categories for efficient content-based search. In: Smeulders AWM, Jain R (eds) Proceedings of the First International Workshop on Image Databases and Multimedia Search (IDMS), 1996, Amsterdam, The Netherlands, Intelligent Sensory Information Systems, pp 107–114
12. Wallace TP, Wintz P (1980) An efficient three-dimensional aircraft recognition algorithm using normalised Fourier Descriptors. Comput Graphics Image Process 13: 99–126



SADEGH ABBASI is currently a research fellow at the Centre for Vision, Speech and Signal Processing (CVSSP) at the University of Surrey. He is an on leave Lecturer from the University of Guilan, Rasht, Iran. He received his BSc and MSc in Telecommunication Systems from the University of Tehran, Iran and worked for 2 years at Iran Telecommunication Research Centre. From October 1994, he joined the CVSSP where he started his PhD in shape similarity retrieval. He has been the author or co-author of a number of papers in this area. His research interests include image database systems, shape representation and pattern recognition.

FARZIN MOKHTARIAN is currently a Lecturer at the Centre for Vision, Speech and Signal Processing at the University of Surrey in Guildford, England. He received his MSc and PhD in Computer Vision from the University of British Columbia, Vancouver, Canada and spent 2 years at NTT Basic Research Labs in Tokyo as a Research Scientist. He recently served on the Conference Board and program Committee of the First International Conference on Scale Space Theory in Computer Vision. His research interests include shape representation, object recognition, multi-scale shape analysis, and image database retrieval by shape content.



JOSEF KITTLER is currently the director of the Centre for Vision, Speech and Signal Processing at the University of Surrey in Guildford. He has worked on various theoretical aspects of pattern recognition and machine vision. He gained experience in many applications including automatic inspection, remote sensing, robotics, speech recognition, character recognition and document processing. His current research interests include pattern recognition, neural networks, image processing and computer vision. He has co-authored a book with the title 'Pattern Recognition: a statistical approach' published by Prentice-Hall. He has published more than 300 papers. He is a member of the Editorial Boards of Pattern Recognition Journal, Image and Vision Computing, Pattern Recognition Letters, Pattern Recognition, Artificial Intelligence, and Machine Vision and Applications.