# Shape Recognition by Clustering and Matching of Skeletons

Hamidreza Zaboli
Amirkabir University of Technology, Tehran, Iran
Email: zaboli@aut.ac.ir

Mohammad Rahmati and Abdolreza Mirzaei
Amirkabir University of Technology, Tehran, Iran
Email: {rahmati,amirzaei}@aut.ac.ir

***Abstract*—We perform the task of shape recognition using a skeleton based method. Skeleton of the shape is considered as a free tree and is represented by a connectivity graph. Geometric features of the shape are captured using Radius function along the skeletal curve segments. Matching of the connectivity graphs based on their topologies and geometric features gives a distance measure for determining similarity or dissimilarity of the shapes. Then the distance measure is used for clustering and classification of the shapes by employing hierarchical clustering methods. Moreover, for each class, a median skeleton is computed and is located as the indicator of its related class. The resulted hierarchy of the shapes classes and their indicators are used for the task of shape recognition. This is performed for any given shape by a top-down traversing of the resulted hierarchy and matching with the indicators.**

**We evaluate the proposed method by different shapes of silhouette datasets and we show how the method efficiently recognizes and classifies shapes.**

***Index Terms*—object recognition, shape recognition, shape classification, skeleton, radius function, clustering.**

## I. INTRODUCTION

In the field of computer vision, recognition of objects in the images is a main and important task which should be performed for understanding the scene. This task may be done using different features of the objects with respect to related application. Some of these features are shapes of objects, color, brightness, motion vector, etc [1]. Based on the application, a feature or a combination of them may be useful and may be applied. Shape of object is a rich and effective feature of the object which has a key role for recognition of objects. In this work, we aim to recognize objects using their shapes. Using of this feature leads to another problem so-called "shape recognition". Typically, shape recognition interacts with similar tasks such as shape representation, shape matching and shape classification. One of them is result of previous one and another is input for subsequent one. For example, the task of shape matching can not be performed without a method for representation of shapes. Also recognition of a shape is performed by a combination of classification and matching of the shapes [1].

In this paper, we perform the task of shape recognition by first proposing a method for representation and description of shapes. Then we give a method for matching of the represented shapes. At last, clustering methods is used for classification of the shapes and we recognize the shapes by clustering and matching of them.

Our representation method is based on the skeleton of shape. Skeleton of a shape is a set of connected medial lines and curves along the different parts of the shape [2]. There are different methods for extracting skeleton of a shape. Since skeleton-based methods deal with the skeleton of shapes, they consider structure of them. Indeed, skeleton-based methods belong to the structural shape recognition methods [2]. Structural methods try to understand and explore the object which the shape has been derived from. In fact, they attempt to decompose shape into its meaningful and distinct parts in most cases.

Siddiqi et al. in [3] presented a skeleton-based method so-called shock graphs. They colored skeletal segments into 4 types and extracted a shock tree using a shock grammar. Sub-graph isomorphism was used for matching the shock trees and a distance measure was acquired. Similarity or dissimilarity of shapes may be determined using the distance measure. We have improved efficiency and recognition rate of this method in the presence of occlusion and missing parts in our previous work [4]. In this paper, we propose a different skeleton-based method from [3,4].

We consider extracted skeleton as a connectivity graph such that junctions are considered as graph nodes and skeletal curve segments is considered as graph edges[1]. Connectivity graph for each skeleton contains topological information of the skeleton and shape. Beside the topological information, we attend to geometric information of the shape along the skeletal curve segments. This is performed using radius function which

[1] Note that we use terms “skeletal curve segment” as a curve segment of the skeleton which connects two junctions or a junction to a terminal point or vice versa. Furthermore, there is no junction between the two end points of the curve segment.

is defined on the skeletal points and it captures convexities and concavities of the different parts of the shape along the skeletal curves.

Matching phase of the connectivity graphs and their topological and geometric information is performed based on mapping of the junction points and the edges. Moreover, a distance measure is defined in the matching phase and it is used to determine similar and dissimilar shapes. At this point, we are able to differ between similar and dissimilar shapes.

Clustering is defined as a method for exploring and grouping similar patterns within a set of patterns [5]. Clustering methods are applied to a set of patterns to partition the set into classes such that all patterns within a class are similar to each other and different from the members of other classes. We use clustering methods to categorize different classes of shapes and find groups of similar shapes. As a result of this process, classification and recognition of shapes are performed using a hierarchical clustering method which generates a hierarchical structure. Input shape is matched with different classes of shapes along the hierarchy in a top-down manner to reach to a similar class. At this point, we can recognize the input shape.

This paper is organized as follows. In Section II skeleton of the shape is extracted and it is transformed into a connectivity graph. Section III represents the method for matching the connectivity graphs and distance between the shapes. Section IV discusses how to cluster and classify the shapes using clustering methods. Experimental results and discussions are represented in Section V and in Section VI the paper is concluded.

## II. Skeleton and Extraction of Connectivity Graph

There are various methods to extract skeleton of a closed area. Superiority of skeleton is that it contains useful information of the shape, especially topological and structural information. To have skeleton of a shape, first boundary or edge of the shape is extracted using edge detection algorithms [6,7] and then its skeleton is generated by skeleton extraction methods [8,9]. Medial axis is a type of skeleton that is defined as the locus of centers of maximal disks that fit within the shape [1,8]. We use medial axis as the skeleton of shape.

In the following, we aim to extract two features from the shape with respect to its skeleton. One feature is topology of the shape and another is geometric information such as thickness and thinness of the different parts of the shape.

### *A. Topological Information*

We represent skeleton of a shape as a graph such that junction points are graph nodes and skeletal curve segments are graph edges. We call this graph as connectivity graph of the skeleton. Therefore, we may have for any given shape, its skeleton and its connectivity graph. Connectivity graph perfectly represents topology of the skeleton and structure of the shape. Fig. 1 shows a sample shape, its skeleton and its connectivity graph.

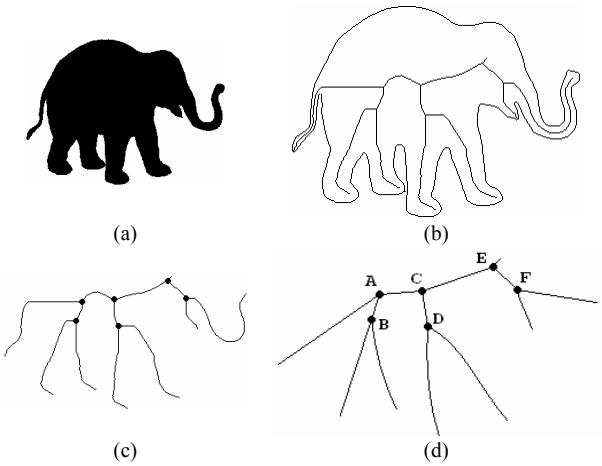



Figure 1. (a): shape of an elephant, (b): boundary and skeleton, (c): skeleton with junction points, (d): connectivity graph of the skeleton.

Fig. 1(a-d) show silhouette of a elephant, boundary and skeleton, skeleton with junction points and connectivity graph of the skeleton, respectively. Thus, for any given shape we may have a connectivity graph, representing topology of the skeleton and the shape. Connectivity graph contains arrangement of junctions relative to other junctions and edges. Also it includes information such as number of junctions and edges and how edges connect to junctions. Connectivity graph does not contain geometric information such as convexities or concavities, thickness or thinness of parts.

### *B. Geometric Information*

In order to include geometric information into our shape representation method, we need to a feature which captures convexities, concavities, thickness and thinness of different parts of a shape. This is achievable using "radius function". This function is defined and can be computed for all skeletal points. Radius function *R(p)* for point *p* on the skeleton is the radius of the maximal inscribed disc touches the boundary of the shape [2]. In fact, variations of this function along the skeletal points create convex or concave parts of a shape. This is shown in Fig. 2. As seen in this figure, fixed values of radii from A to B create a flat part and increases of radii values from B to C create a convex part in the shape.

However, it may be considered as a hen and egg problem i.e. convex and concave parts of a shape cause increases and decreases of radius function, respectively.

For each skeletal curve segment on the skeleton, values of radius function are computed starting from a junction point. For example, in Fig. 1(d), three skeletal curve segments branch out from junction point *A*. Values of radius function are computed for each one of the three curve segments starting from *A*.

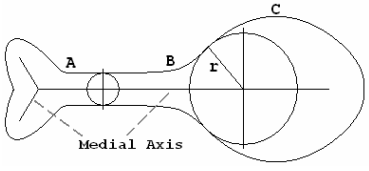



Figure 2. Variations of radius function on the skeleton create concave and convex parts of a shape.

In this way, we have at least a radius vector for each skeletal curve segment. Computed vector for each skeletal curve segment is assigned to its corresponding edge in the connectivity graph of the skeleton. Up to this point, we have both of topological and geometric information for any given shape.

### *C. Pruning of the Skeleton and Connectivity Graph*

Skeleton of a shape has different segments with different importance. Some skeletal curve segments are more important and have key role for forming of the shape. These segments cause that a shape differ from other shapes. In fact, these major differences create different objects and shapes. On the other hand, some skeletal curve segments have not key role for creating the related object and shape. These curve segments only create minor differences between the shapes which belong to a category. Moreover, skeleton of a shape is a sensitive structure which may be changed in the presence of boundary noises [10,11]. Boundary noises can cause that some minor skeletal curve segments be added to or deleted from the skeleton (and connectivity graph). Fig. 3. shows this sensitivity.

As seen in the figure, head of the elephant in (a), (the part within dashed ellipse) is a little different from the head in (b). This small difference causes that the skeletal curve segment 'a' and its junction point not to be exist in the skeleton of Fig. 3(b).

In order to overcome sensitivity of the skeleton to small changes on the boundary, we use of a pruning method. In this pruning method, we aim to delete minor and sensitive skeletal curve segments from the skeleton. Length of each skeletal curve segment is computed and average value of them is calculated. Then any skeletal curve which is smaller than a fixed portion of the average value is deleted from the skeleton. Therefore, we have (1) for pruning of sensitive skeletal curves segments:

$$\frac{1}{n\alpha}\sum_{i=1}^{n} length(i) \qquad (1)$$

where $i$ is a skeletal curve segment of the skeleton, $n$ is number of the skeletal curve segments and $\alpha$ is a fixed positive value in a matching process. Any skeletal curve segment with a length less than (1) is deleted from the skeleton. Note that by deleting a skeletal curve segment which has junction points at its two end points, the two junction points and their connections are transformed into a unique junction point. Also, if a skeletal curve segment has a junction point at one end and a terminal point at another end, then by deleting it, its junction point may be transformed into a typical point.

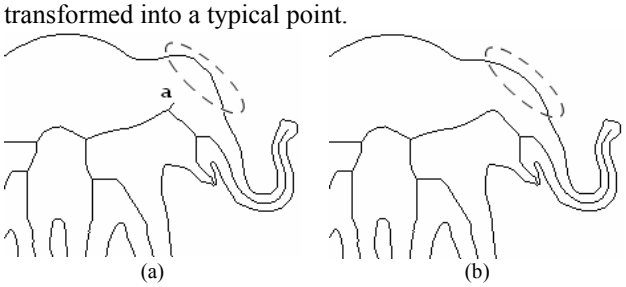



Figure 3. Sensitivity of skeleton to minor noises on the boundary.

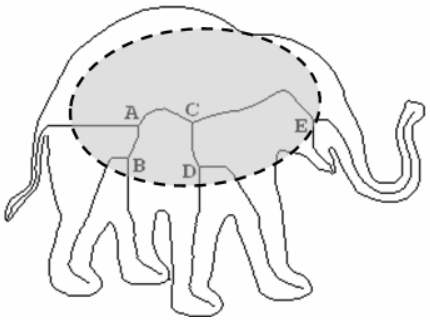



Figure 4. Shape of an elephant and its main parts inside the grayed area.

### *D. Weighting scheme for the Skeleton and Connectivity Graph*

Up to this point we have a well-structured skeleton for any shape and we may transform it into a connectivity graph containing both topological and geometric information. Before transforming into a connectivity graph, we apply a weighting scheme to the skeleton. The goal of this phase is to explore basic and more important parts of the skeleton and shape. In this way, similarity of the basic parts of the skeleton are taken more significant into consideration, while differences of the border and irrelevant parts are less attended.

For example, in the matching process and determining the similarity between two horses, presence/absence of the ears is not as important as presence/absence of stomach.

We decompose a shape into different parts such that internal parts of the shape and its skeleton are more significant and weightier than external parts. A simple example of this weighting method has been given in Fig. 4 for shape and skeleton of an elephant. As shown in the figure, the parts of the shape and skeleton which lie within the grayed area are of more importance. We give more weight to the skeletal curve segments which are located inside this area. A similar method was used in [12], but the method was not skeleton-based.

The following pseudo codes represent the procedure for weighting the skeleton. Assume skeleton $S$ with its junction points and skeletal curve segments. We represent it by connectivity graph $G(V,E)$, where set $V=\{v_1,v_2,\ldots,v_n\}$ includes vertices of $G$ and set $E=\{ e_1,e_2,\ldots,e_m \}$ includes edges of $G$.

```
Procedure Graph_Weighting(G)
  For all ei
    If ei has a terminal point at one of
        its ends ⇒ {
        Mark ei as 'external';
        Mark ei as 'weighted';
        Weight(ei)=1;
        }
    else Mark ei as 'internal';
  Endfor;
  For all ei check_number(ei)=0;
  For any external edge ei
    Internal_Weighting(ei);
  Endfor;
Endprocedure;

Procedure Internal_Weighting(G,ei)
```

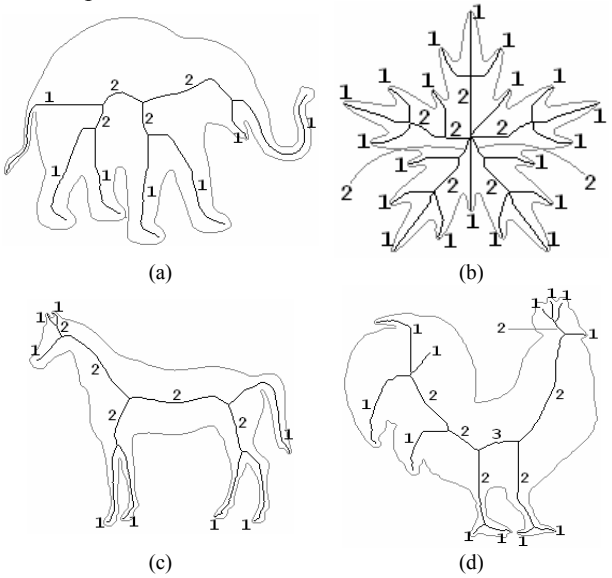
```
  For any edge e_j connected to e_i
    check_number(e_j)++;
    If e_j is 'weighted' ⇒ {
      If weight(e_j)>weight(e_i)+1 ⇒
        weight(e_j)=weight(e_i)+1;
      }
    else weight(e_j)=weight(e_i)+1;
    If e_j is 'internal' and
       check_number(e_j)<number of edges
       connected to e_j ⇒
       Internal_Weighting(G,e_j);
  Endfor;
Endprocedure;
```

Using our weighting method skeleton of Fig. 4 is weighted as shown in Fig. 5(a). As seen in this figure, the skeletal curve segments which have lied within the grayed area (i.e. internal parts) in Fig. 4 obtain more weight than other skeletal curve segments. Weights values are positive integers which start from one and increase with step value 1 by moving to the interior of skeleton and shape. However, it may be chosen as other weighting values. For example weights of each skeletal curve segment may also depend on its length, etc. Fig. 5 (b-d) shows more sample weighted skeletons using our weighting scheme.

## III. Matching of the Skeletons and Connectivity Graphs

Up to this point, we have a weighted connectivity graph with topological and geometric features for any given shape. Different methods have been proposed to match graphs and trees. These methods can be used for matching of the extracted connectivity graphs.

Sub-graph isomorphism has been used in [3,13] for the task of shape matching. We propose a tree matching method and we use it for matching of the skeletons and connectivity graphs. Besides matching of the topology, we take geometric features into consideration. Note that connectivity graphs which are extracted from the skeleton of a shape are trees.



(a) (b) (c) (d)

Figure 5. Four sample shapes and their skeletons weighted using our weighting scheme.

Thus, tree matching methods may be used to match these graphs. Result of the matching process is a distance value which denotes to the difference and distance between the two given shapes.

### A. *Method for Tree Matching*

Assume connectivity graph $G_1$ extracted from skeleton $S_1$, such that its edges connect to each other by nodes (junctions). Edges of graph $G_1$ are divided into two types internal and external. Both end points of internal edges are graph nodes while external edges have a node at one of these end points and a terminal point at another end point. Moreover, assume graph $G_2$ as another connectivity graph extracted from skeleton $S_2$. Graphs $G_1$,$G_2$ are free trees because they have no root.

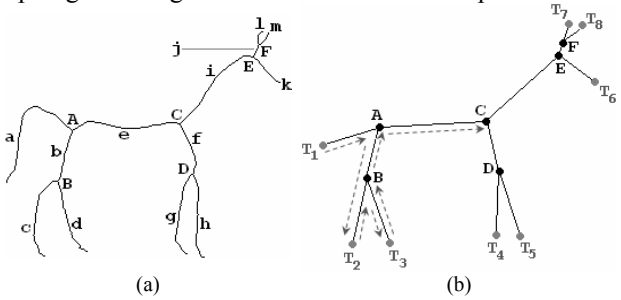
We traverse edges of the graphs in order of the appearance and a counter clock-wise preference at the junctions (nodes). A terminal point is chosen as start point and every edge is replaced with its radius vector. Note that the radius vectors are computed before and their directions can be reversed with respect to the two nodes at the ends of each edge. The traversing will terminates at the start point. Fig. 6 shows this traversing process.

Fig. 6(a) shows labeled skeleton of a horse and Fig. 6(b) shows connectivity graph of the horse skeleton. As seen in Fig. 6(b), terminal point $T_1$ has been chosen as start point. Direction of traversing is from $T_1$ to $A$. Therefore, corresponding radius vector to edge $T_1A$ is placed as the first element. At point $A$, the nearest edge which is appeared by a counter clock-wise rotation is edge $AB$. This edge is chosen as the next path. The traversing is continued in the specified path by dashed arrows and radius vector corresponding to each edge is replaced.

In this way, any given free tree is transformed into a sequence containing geometric features of all skeletal curve segments. The resulted sequence of traversing of the tree shown in Fig. 6(b) is as follows:

$$\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5},\overrightarrow{T_5D},$$
$$\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},\overrightarrow{AT_1}$$

Moreover, our traversing process keeps topological structure of the tree such that the tree can be constructed from the sequence. In this way, we may have an attributed sequence for any given skeleton and connectivity graph. Therefore, any given shape can be represented by an attributed sequence containing both topological and geometric features of the shape.



(a) (b)

Figure 6. (a): Skeleton of the shape of a horse, (b): connectivity graph of the skeleton and traversing it starting from $T_1$.

Length of the sequence depends on the length of the skeleton and number of the values of radius vectors.
As a result, the matching process for any two shapes is reduced to matching of two sequences. We perform this sequence matching using dynamic programming method. A cost function is considered for matching the elements of the sequences. Note that elements of the sequences are values of the radius vectors.

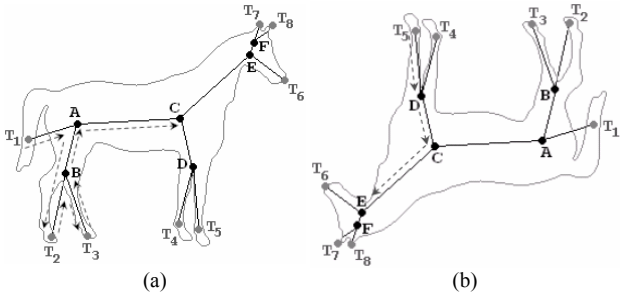
Furthermore, cost function takes weight of each edge into consideration. As we mentioned before, each edge of the connectivity graph is indicator of a skeletal curve segment. Also for each skeletal curve segment and its relative edge we have a radius vector containing values of the radius function. With respect to our weighting scheme and the weights computed for each edge of the graph, we apply the weights to the sequences and the cost function.
Assume two sequences $V,U$ resulted from two skeletons $S_1,S_2$ with length of $n,m$ , respectively. Function $Cost(i,j)$ is defined as cost of matching of the first i$^{th}$ and j$^{th}$ elements of the sequences $V,U$ . This function is given in (2).

$$Cost(i,j) = min\begin{Bmatrix} cost(i,j-1), \\ cost(i-1,j), \\ cost(i-1,j-1) \end{Bmatrix} + \left(weight(V_i)\times V_i\right) - \left(weight(U_j)\times U_j\right) \quad (2)$$

where $V_i$, $U_j$ refer to value of the $i^{th}$ , $j^{th}$ elements of sequences $V,U$ , respectively. Also, $weight(V_i)$ is the weight assigned to the skeletal curve segments which element $V_i$ belongs to. A similar notation is defined for $weight(U_i)$ and its relative skeletal curve segment.

To match whole of the sequences $V,U$ and computing total distance between them, $cost(n,m)$ should be computed. In fact, $cost(n,m)$ returns the distance between the two skeletons and the two given shapes. We use this cost as a distance measure and we may compute the distance between any two given shapes. As a result of this process, it is straightforward to determine similarity or dissimilarity of shapes.

One thing to be mentioned is that the traversing process and generation of the sequence from connectivity graph depends on the start point. It means that the generated sequences for two similar shapes with different start points are different. Fig. 7 shows this dependency. As seen in this figure, shape of the horse in Fig. 7(a) has been rotated by $\pi$ in Fig. 7(b).




(a) (b)

Figure 7. Shape of a horse and its connectivity graph in (a). rotated shape of (a) by 180 in (b).

Consequently, start point in the tail of the horse at the left shape is changed to another point in the leg at the right shape. Generated sequence from Fig. 7(b) starting from terminal point $T_5$ is as follows:

$$\overrightarrow{T_5D},\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},$$
$$\overrightarrow{AT_1},\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5}$$

This dependency to start point makes the method sensitive to rotation transformation. To make our method invariant to rotation, we transform resulted sequences into a cyclic sequence and we perform matching of the sequences with different start points. Different start points are chosen at terminal points. Moreover, the different start points are applied only to one of the two sequences. For example, in Fig. 7(b) terminal points (points $T_i$) are chosen as different start points and start point of another skeleton is chosen to be a fixed terminal point. By considering these different start points, it is simply observable that different start points can be applied directly to one of the resulted sequence. This may be performed by saving location of the terminal points in the traversing stage.

Therefore, matching process for any two given shapes using dynamic programming is performed starting from different terminal points in one of the two sequences. Different sequences generated from skeleton of the Fig. 7(b) are given in Table 1.

As seen in the table, 8 different sequences are generated starting from 8 different terminal points. All of these 8 sequences should be matched with resulted string from the second shape.

Table 1. Different sequences generated by starting from different start points of Fig 7(b).

| Start Point | Generated Sequence |
|---|---|
| $T_5$ | $\overrightarrow{T_5D},\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},$ $\overrightarrow{AT_1},\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5}$ |
| $T_6$ | $\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},\overrightarrow{AT_1},\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},$ $\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5},\overrightarrow{T_5D},\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6}$ |
| $T_7$ | $\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},\overrightarrow{AT_1},\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},$ $\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5},\overrightarrow{T_5D},\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7}$ |
| $T_8$ | $\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},\overrightarrow{AT_1},\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},$ $\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5},\overrightarrow{T_5D},\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8}$ |
| $T_1$ | $\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5},\overrightarrow{T_5D},$ $\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},\overrightarrow{AT_1}$ |
| $T_2$ | $\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5},\overrightarrow{T_5D},\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},$ $\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},\overrightarrow{AT_1},\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2}$ |
| $T_3$ | $\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4},\overrightarrow{T_4D},\overrightarrow{DT_5},\overrightarrow{T_5D},\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},\overrightarrow{T_6E},\overrightarrow{EF},$ $\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},\overrightarrow{EC},\overrightarrow{CA},\overrightarrow{AT_1},\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},\overrightarrow{T_2B},\overrightarrow{BT_3}$ |
| $T_4$ | $\overrightarrow{T_4D},\overrightarrow{DT_5},\overrightarrow{T_5D},\overrightarrow{DC},\overrightarrow{CE},\overrightarrow{ET_6},\overrightarrow{T_6E},\overrightarrow{EF},\overrightarrow{FT_7},\overrightarrow{T_7F},\overrightarrow{FT_8},\overrightarrow{T_8F},\overrightarrow{FE},$ $\overrightarrow{EC},\overrightarrow{CA},\overrightarrow{AT_1},\overrightarrow{T_1A},\overrightarrow{AB},\overrightarrow{BT_2},\overrightarrow{T_2B},\overrightarrow{BT_3},\overrightarrow{T_3B},\overrightarrow{BA},\overrightarrow{AC},\overrightarrow{CD},\overrightarrow{DT_4}$ |

### *B. Clustering of Shapes*

In this section we aim to cluster and group similar shapes together. The goal of the clustering is to group given shapes such that each shape in a group is similar to other shapes in the group and different from the shapes in other groups. In this way, we may classify given shapes into some classes and then we may use these classes for the task of recognition. There are different methods for clustering a set of patterns. Generally, these methods need a measure for determining the distance/similarity between patterns. In our application, the patterns to be clustered are the shapes. Thus, it is straightforward to use the measure proposed in the previous section as a distance measure for clustering of the shapes.

On the other hand, hierarchical clustering methods cluster patterns by making a hierarchy of clusters. This hierarchy represents how the similar groups of clusters link and join to each other, based on the defined distance measure. In this hierarchy more similar clusters join and unique at the low level of the hierarchy. Moreover, inside each cluster, there are many similar shapes. Fig. 8 shows a sample hierarchical clustering of animal shapes. As shown in this figure, different classes of shapes are evaluated. These animal shape classes are rabbit, mouse, dog, horse, deer, human and bird. Fig. 8 illustrates that in the first level, classes of rabbit and mouse are linked because they are more similar to each other than other classes.

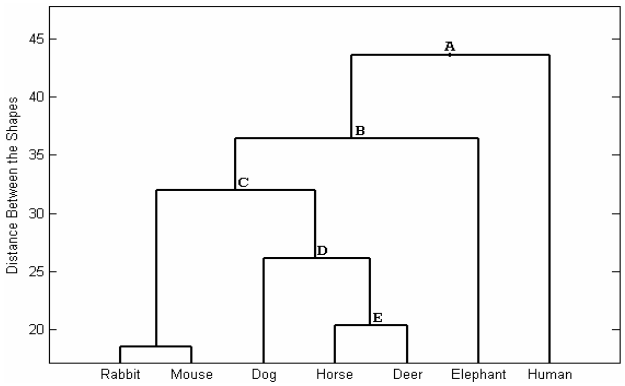



Figure 8. Applying of a hierarchical clustering method for clustering seven classes of shapes using the proposed distance measure.

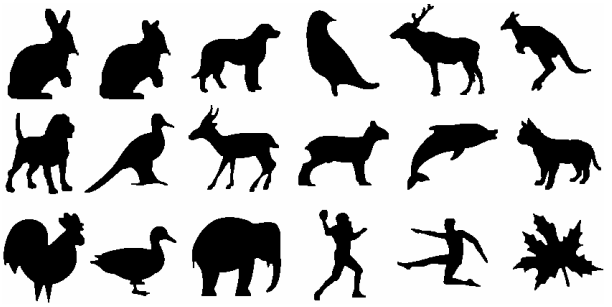


Figure 9. Sample shapes from kimia and MPEG-7 datasets evaluated by our method.

After that, classes of horse and deer are linked due to the high similarity of shapes of a horse and a deer. In level three, class of dog is linked to the classes of horse and deer which they were linked before. These linkages are continued to high levels of the hierarchy such that the classes which are linked together in these levels are less similar to each other.

There are different hierarchical clustering methods that may be used. We apply two methods to cluster shapes. These hierarchical clustering methods are single linkage and complete linkage. We use them to cluster 500 randomly selected shapes of two standard silhouette datasets. These datasets are kimia and MPEG-7 datasets. Fig. 9 shows some of the evaluated shapes from the datasets and Fig. 10 shows results of applying the two clustering methods to the shapes. Fig. 10(a) and Fig. 10(b) show results of hierarchical clustering using single linkage and complete linkage methods, respectively. As seen in the figure, difference between the two methods is only in order of linkage of three classes. These classes are fish, bird and kangaroo. It seems that the single linkage performs more acceptable for clustering and linkage of these classes.

Using the clustering methods, we may categorize and classify shapes into different classes. This process may be considered as a learning phase. After this step our system learns different classes and categories of shapes and it is able to recognize that which shape belongs to which class. This is possible by looking in the classes for the

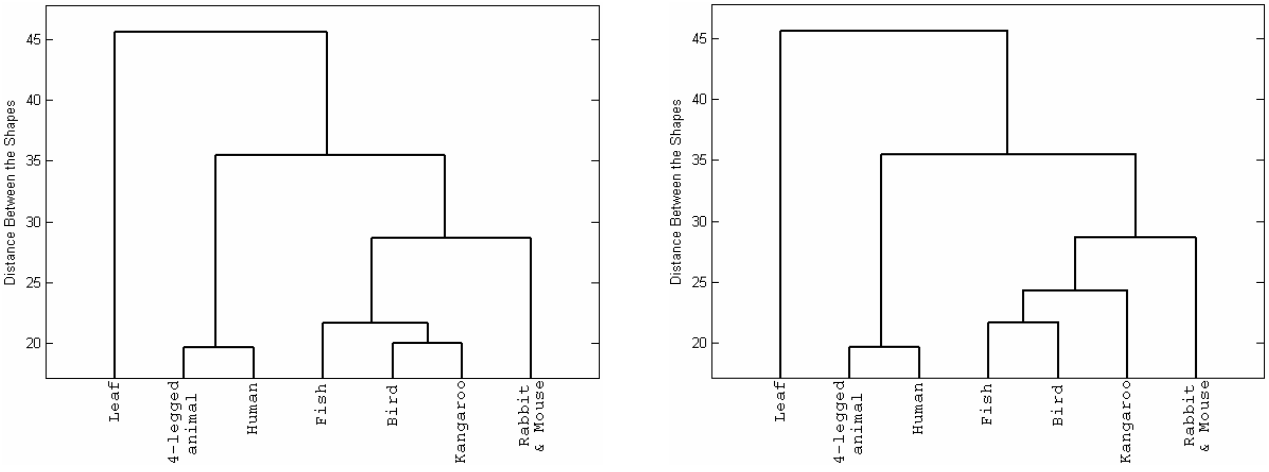



Figure 10. Clustering of shapes classes using two hierarchical clustering methods single and complete linkage in (a),(b), respectively.

given shape or a shape which is close to the given shape. The class which the retrieved shape belongs to, is the result and the target class.

Although up to this point using the proposed method, we have a recognition system which may learn and then may recognize any input shape, but with respect to the learning phase and high number of the shapes, any query shape should be matched with all of the previous input shapes. This causes a huge complexity and time overhead for the recognition process. In the next section, we propose a method for determining an indicator pattern for each class to deal with the high recognition overhead.

### *D. Median Skeletons as Classes Indicators*

After introducing a distance measure between shapes and classification of them using hierarchical clustering methods, it is straightforward to match shapes with different classes of shapes ordered by the clustering methods in a top-down manner. Any given shape is matched with classes of shapes from highest level of the resulted hierarchy to the lowest level. At the lowest level, the target class which the given shape belongs to, is found.

In this section, we propose a median skeleton and pattern for any shape class and we use it as an indicator of its relative class. This indicator helps to recognition process. Since the median skeleton has topological and geometric features of all shapes included in a class, it can be used for matching with the given shape. Using this indicator it is not necessary to match a given shape with all shapes in a class and it is sufficient to match it only with the indicator. This significantly reduces huge matching and recognition overhead.

For any shape class, we compute an indicator which has both topological and geometric features of a shape. In fact, a class indicator is a shape by itself which represents a medium form of all shapes in the class. We call this indicator as "median skeleton".

Indicators may be computed for each class in the learning phase. By assigning a new shape to a class the process of computing median skeleton is executed only once for indicator of the class and the new shape. Note that computing median skeleton for any two shapes is only performed after matching of them. In this way, we anticipate to have a median skeleton which represents a meaningful shape. Computing median skeleton for two unmatched shapes with huge difference and distance do not generate a meaningful skeleton and shape.

To compute a median skeleton between any two given shapes, we consider their skeletons. Assume skeletons $S_1$,$S_2$ such that their radius vectors have been computed before and we have both topological and geometric features of them. As we mentioned in sub-section A of section II, we may have two sequences $V_1$,$V_2$ for skeletons $S_1$,$S_2$ as their descriptors, respectively. In the matching phase using dynamic programming method, we may match each part of $V_1$ which represents a skeletal curve segment of skeleton $S_1$ with its corresponding part of $V_2$. As we stated earlier, each part represents a skeletal curve segment of the skeleton. We call these parts as vectors. Therefore, for computing a median skeleton it is

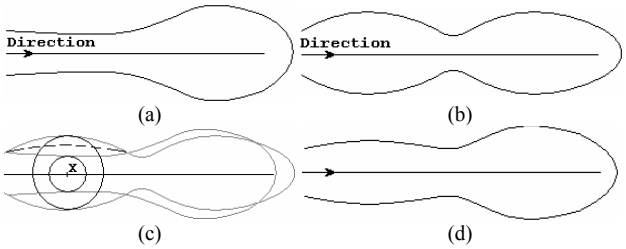



Figure 11. Computing median shape. (a),(b): two input shapes, (c): computing median values, (d): resulted median shapes for (a),(b).

sufficient to compute a median vector for any two matched (corresponding) vector of $V_1$,$V_2$.

In Fig. 11 an example of computing median vector for a part of a shape is shown. In order to compute median points, for point $X$ on skeleton $S_1$ and its corresponding point $X'$ on skeleton $S_2$ with radius values $r$ and $r'$, respectively, radius value for median point is calculated as an average value between $r$ and $r'$. This process is applied to each two corresponding values of sequences $V_1$,$V_2$. Also, the process of computing median vector is performed for each two corresponding vectors.

In this way, we have a median vector for any two corresponding vectors of $V_1$,$V_2$. These median vectors are computed by traversing of sequences $V_1$,$V_2$. By replacing computed median vectors successively, a new sequence is generated. This new sequence represents median shape for sequences $V_1$,$V_2$. Therefore, we may have a median sequence, a median skeleton and a median shape for any two input shapes.

Although we define the process of computing median skeleton for two given shapes, it may be defined for $n$ input shapes simply. In this case, calculating of average radius value is performed between $n$ radius values.

Fig. 12 shows shapes of a horse and a deer and their median shape computed using our method. Fig. 12(c) shows a median shape for the horse and deer in Fig 12(a,b). Although, resulted median shape may not be exist in the real world, but it represents a correct indicator for the two shapes.

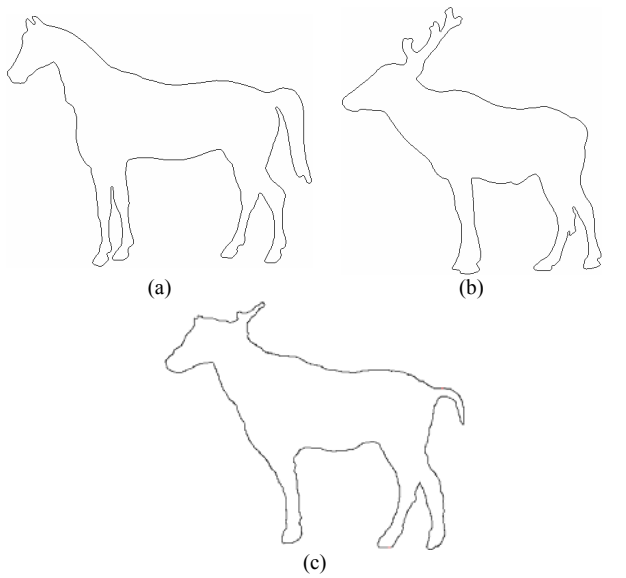



Figure 12. (a),(b): shapes of a horse and a deer, respectively. (c): median shape for the horse and the deer.

As seen in the figure, ears of horse and horns of deer have been lightened in the median shape. This process can be applied to all shapes in class of horse or class of deer. Also, this may be performed on a more general class e.g. class of four-legged animals depending on application. The proposed method for computing median shape is applied to classes of shapes which have been obtained by a hierarchical clustering method.

Up to this point, we have a hierarchy of shapes classes and each class has a median shape as its indicator. Thus, to recognize any input shape it is sufficient to match the input shape with indicator shapes starting from highest level of the hierarchy to the lowest level. At each level, the indicator shape which is closer to the input shape is chosen and the process is continued with its relative class and sub-tree.

### *E. Hierarchical Recognition*

As we have mentioned earlier, any given shape must be matched against the indicators in a top down manner in the hierarchy. This offers a hierarchical recognition process. Consider the hierarchy represented in Fig. 8. Suppose that shape of a cow is given as an input shape to be recognized using the obtained hierarchy. In the first level at point *A*, shape of a cow is more similar to the left sub-tree, thus the human class is rejected. Also, in the next level at point *B*, the class of elephant is rejected. This process continues till we reach to point *E* where the final decision is made. At this point the shape of cow is compared with indicators of classes of horse and deer, and class of deer is recognized as the most similar to the shape of cow.

It is north worthy that the recognition is performed only by 5 matching operation which has a few computation and recognition overhead. Note that there is no class of cow in the learned classes of the hierarchy. However, it is obvious that by increasing the number of learned classes, recognition of input shapes will be more accurate.

## IV. Experiments and Results

In this section, we present evaluations and results of using our method for the task of shape recognition. Note that the methods which are proposed for shape recognition and classification problem deal with binary images of shapes so-called silhouettes. In fact, these methods consider silhouettes as inputs of the shape recognition system which may be resulted as outputs of an image segmentation system [7]. As we stated before, our shape recognition system falls into this category and deals with binary images of shapes (silhouettes).

We perform our experiments on two standard binary image datasets namely kimia and MPEG-7. These datasets contain different classes of shapes and each class contains different shapes of real world objects. Moreover, we design a dataset of 100 deformed shapes for performing more evaluation of our system. Fig. 13 shows some example shapes of this dataset.

As the first experiment, we perform classification on a set of shapes in the datasets using our method.

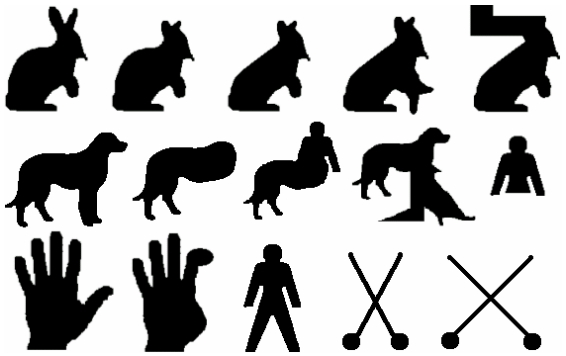


Figure 13. Example shapes of our designated shape dataset.

In addition to the evaluation, it may be concerned as a phase for learning of the system. We learn the system by evaluating a set of 590 randomly selected shapes from the datasets. First, we compute distances between any two shapes of the set using the distance measure presented in section III. Consequently, a distance table is generated which represents the distances between any two shapes of the set. Then we apply single linkage clustering method on the distance table and a hierarchy is obtained. In the resulted hierarchy, we evaluate number and rate of the correct classification of the shapes inside each class. Table 2 summarizes results of the evaluation.

There are 4 major classes in the hierarchy and we evaluate number of correct hits and false hits inside each class.

After learning phase, the system is ready to recognize any input shape. As a result, in the next experiment, we evaluate our method for recognition of new given shapes. In this experiment, we randomly select 440 new shapes from the datasets and evaluate and recognize them using the learned system. Any new shape is matched with indicators of the classes in the hierarchy and the closest indicator to the new shape is found. Then the new shape is assigned to the relative class and a new median shape is computed for the current indicator and the new shape. The new median shape is assigned to the class as its new indicator shape. Thus, in this process the system recognizes and learns new shapes at the same time.

We evaluate average recognition rate of the system in this process. Table 3 shows results of this evaluation.

Table 2. Results of classification of 590 shapes using single linkage clustering method.

| **Class** | Human | 4-legged animals | Leaves & Flowers | Birds |
|---|---|---|---|---|
| **Recognition Rate** | 92 % | 89 % | 81 % | 87 % |
| **Number of false hits / Number of shape inside the class** | 6/76 | 31/284 | 29/154 | 13/96 |

Table 3. Recognition rates of the system after the learning phase.

| **Class** | Human | 4-legged animals | Leaves & Flowers | Birds |
|---|---|---|---|---|
| **Recognition Rate** | 96.1 % | 94.5 % | 84.7 % | 93.2 % |

As seen in the table, after learning the system, recognition rates have been increased for all 4 classes, especially for classes of birds and 4-legged animals. Moreover, recognition of a new shape is performed by a few matching operation. This is a great result for recognition of shapes. Consider there are *n* detected classes in the system. The worst case occurs when the hierarchy is like a binary tree which always grows from its left child. In this case, recognition of a new given shape can be performed by *n-1* matching operation plus an operation for computing a new indicator.

By recognition of new shapes, recognition rate of the system increases gradually. Note that values of recognition rates in Table 1 are average recognition rates. We continue recognition and test the system by learning 300 new shapes from the datasets. Fig. 14 shows variations of recognition rate by growing number of learned shapes from 590 to 1330. As seen in the figure, before learning of 1000 shapes, recognition rate of the system grow up to 94.6 %. After 1000 shapes, recognition rate almost do not increase and remains about 95 % due to the fact that the system is saturated at this point.

Next experiment is a retrieval test which shows efficiency of our distance measure presented in section III. In this experiment we do not use clustering methods. A sample shape is selected and the datasets are looked for the same and most similar shapes based on the distance between them and the selected shape. Retrieved shapes are sorted in order of increasing of the distance. This process is performed for a set of shapes and number of correct and false hits for each shape is calculated. We perform retrieval test on 60 sample shapes from the datasets and for each sample shape, we retrieve 10 closest shapes from the datasets. Therefore, the experiment is performed on 600 shapes. Also we compare result of the shape retrieval with another skeleton-based method so-called shock graph [15] and a contour-based shape matching method so-called shape context [1]. Table 4 summarizes this comparison. In the table, recognition rates of our method and other methods for the first 10 closest shapes are shown. As seen in the table, our method outperforms the other methods and it has higher recognition rate for retrieving correct similar shapes. This illustrates high efficiency of our distance measure.

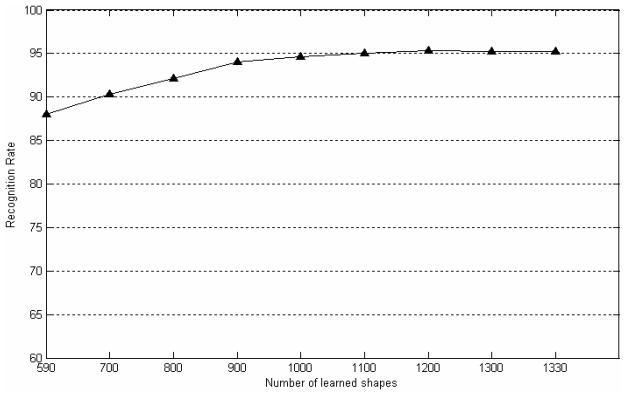



Figure 14. Improvement and increase of recognition rate by increase of learned shapes.

Table 4. Results of retrieving of 600 sample shapes from the datasets. For each shape, 10 closet matches are retrieved and recognition rates of the methods are computed.

| **Method** | **1st** | **2nd** | **3rd** | **4th** | **5th** |
|---|---|---|---|---|---|
| **Our method** | 100 | 100 | 99 | 99 | 98 |
| Shape Context[1] | 97 | 91 | 88 | 85 | 84 |
| Shock Graph[15] | 99 | 99 | 99 | 98 | 98 |
| **Method** | **6th** | **7th** | **8th** | **9th** | **10th** |
| **Our method** | 97 | 97 | 97 | 94 | 85 |
| Shape Context[1] | 77 | 75 | 66 | 56 | 37 |
| Shock Graph[15] | 97 | 96 | 95 | 93 | 82 |

## V. CONCLUSION

In this paper, we proposed a new skeleton-based method for shape recognition and classification problem. We performed these tasks by first representing a method for capturing geometric and topological features of the shape and its skeleton. This is performed by traversing skeleton of shape and transforming it into an enriched sequence. Then we used dynamic programming method to match the sequences and we introduced a distance measure for determining similarity or dissimilarity of shapes. The distance measure was used as a measure for classification of shapes by hierarchical clustering methods. Finally, recognition of shapes was performed by matching them with shapes classes in the resulted hierarchy.

The hierarchical structure improves efficiency of recognition and its computation overhead. This is useful for real world applications which need high speed recognition of input shapes. Moreover, learning phase before using the recognition system helps and improves its recognition rate significantly.

## REFERENCES

[1] Serge Belongie, Jitendra Malik, Jan Puzicha, "Shape Matching and Object Recognition using Shape Contexts",IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.24 , No. 24, pp. 509-522, April 2002.

[2] Dengsheng Zhang, Guojun Lu , "Review of shape representation and description techniques", Pattern Regocnition 37 , 2004, pp. 1-19.

[3] Kaleem Siddiqi, Ali Shokoufandehs,Sven J. Dickinsons, Steven W. Zucker, "Shock Graphs and Shape Matching" , International Journal of Computer Vision 35(1) , 1999, pp. 13-32.

[4] H. Zaboli, M. Rahmati, "An Improved Shock Graph Approach for Shape Recognition and Retrieval", IEEE International Conference on Modeling and Simulation (AMS07), 2007, pp. 428-433.

[5] E. Gose, R. Johnsonbaugh, S. Jost, Pattern Recognition and Image Analysis, Prentice Hall, 1996.

[6] T. Bernier, J.-A. Landry, "A New Method for Representing and Matching Shapes of Natural Objects", Pattern Recognition 36, 2003, pp. 1711-1723.

[7] Rafael C. Gonzalez, Richard E. Woods, Digital Image processing, Addison-Weseley, 1993.

[8] H. Blum, "A Transformation for extracting new descriptors of Shape", in: W. Whaten-Dunn(Ed.), MIT Press, Cambridge, 1967. pp. 362-380.

[9] Louisa Lam, Seong-Whan Lee, Ching Y. Suen, “Thining Methodologies – A Comprehensive Survey”, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 14,No. 9, 1992.

[10] R. L. Ogniewics, O. Kubler, "Hierarchic Voronoi Skeletons", Pattern Recognition 28, Vol. 3, 1995, pp. 343-359.

[11] M. van Eede, D. Marcini, A. Telea, C. Sminchisescu, S. Dickinson, "Canonical Skeletons for Shape Matching", 18th International Conference on Pattern Recognition, Vol. 2, 2006, pp. 64-69.

[12] L. Gorelick, M. Galun, E. Sharon, R. Basri, A. Brandt, "Shape Representation and Classification Using the Poisson Equation", IEEE Transaction on Pattern Recognition and Machine Intelligence, Vol. 28, No.12, 2006, pp. 1991-2005.

[13] Marcello Pelillo, Kaleem Siddiqi, Steven W. Zucker, "Matching Hierarchical Structures Using Association Graphs", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 21, No. 11, November 1999.

[14] Thomas B. Sebastian, Benjamin B. Kimia, “Curves vs Skeletons in Object Recognition”, ICPR, 2001, pp. 22-25.

[15] Thomas B. Sebastian, Philip N. Klein, Benjamin B. Kimia, “Recognition of Shapes by Editing Shock Graphs”, ICCV 2001, pp. 755-762.

[16] Sung Kwan Kang, Muhammad Bilal Ahmad, Jong Hun Chun, Pan Koo Kim and Jong An Park, Modified Radius-Vector Function for Shape Contour Description, Lecture Notes in Computer Science, Vol. 3046, 2004, pp. 940-947.