Integrating Contour and Skeleton for Shape Classification

Xiang Bai Wenyu Liu
Dept. of Electronics and Information Engineering
Huazhong University of Science and Technology, China
xiang.bai@gmail.com, liuwy@hust.edu.cn

Zhuowen Tu
Lab of Neuro Imaging
University of California, Los Angeles
ztu@loni.ucla.edu

Abstract

Shape analysis has been a long standing problem in the literature. In this paper, we address the shape classification problem and make the following contributions: (1) We combine both contour and skeleton (also local and global) information for shape analysis, and we derive an effective classifier. (2) We collect a challenging shape database in which there are 20 categories of animals, with each having 100 shapes. All these shapes are obtained from real images with a large variation in pose, viewing angle, articulation, and self-occlusion. (3) We emphasize the importance of having good representation for shape classification to address the unique characteristics of shape. A thorough experimental study is conducted showing significant improvement by the proposed algorithm over many of the state-of-the-art shape matching and classification algorithms, on both our dataset and the well-known MPEG-7 dataset [19]. In addition, we applied our algorithm for recognizing and classifying objects from natural images and obtained very encouraging results.

1. Introduction

One major task in shape analysis is to study the underlying statistics of shape population and use the information to extract, recognize, and understand physical structures and biological objects. Though being elegant in theory, the general shape statistics [16, 30] learned are yet to be verified to produce compelling results on modern shape database such as the MPEG-7 [19]. In this paper, we focus on 2D shapes of closed contour, which can be represented by continuous points or parameterized by arc length.

One intrinsic difficulty in analyzing shape, unlike image patch, is its lack of a common space. For example, one can define the origin anywhere on the contour, and a certain part may appear or not appear on a particular instance of a shape. One solution is to roughly register all the shapes to the same template and represent them using the same number of aligned points [10]. However, this approach will only work on very focused shapes with small variation. This is due to two reasons: (1) any registration process will introduce artifacts, (2) it is extremely hard to define a common shape to which all shapes can map.

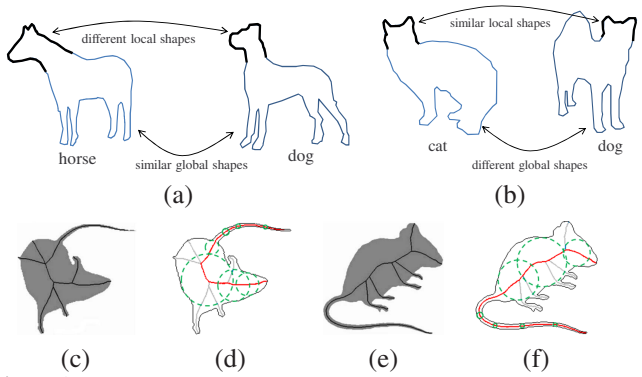


Figure 1. Illustration of the complementariness of using **local** v.s. **global** shape, and **contour** v.s. **skeleton**. The first row ((a) and (b)) shows that different objects may have different local/global contour segments. The second row ((c), (d), (e), and (f))displays two non-rigid shapes which have the same radius sequence on the skeleton paths.

The area of shape analysis, on one hand, has been recently driven by designing smart features for shape matching, such as shape context [4]. This line of the work has produced increasingly encouraging results on matching [9, 1, 2, 4, 14, 19, 21]. On the other hand, these matching-based algorithms have their own limitations. One major task in shape analysis is to recognize a given shape, and tell which type of object it is. This is a classification problem. Matching-based algorithms perform classification essentially through exemplar-based or nearest neighborhood approach by matching the query shape against all those in the database. On few training samples, these algorithms have difficulty capturing the large intra-class variation. On large training samples, it is extremely time consuming to perform shape matching one-by-one.

It is not yet clear how the biological vision systems perform shape understanding. Nevertheless, two sets of concepts have been popular in the shape domain, **contour** v.s. **skeleton** (medial axis) and **local** v.s. **global**. Contour-based approaches [4, 14, 19, 21] are often good at representing detailed shape information and somewhat robust against occlusion, but they are sensitive to articulation and non-rigid

deformation [8]. Skeleton-based approaches [1, 2, 26, 29], on the other hand, can cope well with non-rigid deformations, but only carry rough structural information. The skeleton forms the centers of the maximal disks inside the contour boundary, and the radii of these maximal disks can be used to represent the thickness of an object. Similar shapes sometimes have similar skeleton structure and path. Fig. (1.b) illustrates an example where two shape instances of the same object show similar radius sequences. In the past, these two streams of work have been studied mostly in isolation. Indeed, contour and skeleton provide complementary information (shown in our experimental study). Sometimes, for a certain type of shape, a piece of contour segment (part) might be very informative to tell it apart from the others, e.g. the ears of a cat, even though it might be very small. In other occasions, one really needs to use the global information. Fig. (1.a) shows two examples where both local parts and global shape information are used in shape understanding. It is worth mentioning that the contour information is somewhat implicitly linked with the skeleton, through the use of disks, in some existing works [26, 33, 17]. Neverthless, very few studies have tried to explicitly address the issue of combining contour and skeleton in a shape representation.

Inspired by the above observations, we present a method for shape classification with the following features: (1) it combines both local and global features extracted from contour and skeleton; (2) it can effectively classify shapes of large intra-class variation, inter-class similarity, and occlusion, without using too many training samples; (3) it is easy to implement and is faster than matching-based algorithms. On the MPEG-7 dataset [19] of 70 classes, we obtain 96.6% classification rate using 10 samples (per-class) for training and the remaining 10 for testing. Though widely used, it is evident that the MPEG-7 dataset does not capture the large variation of object shape with 20 instances per class. Therefore, we collected 20 animal classes each having 100 shapes of very large change in pose and deformation. We performed a thorough experimental study and showed improvement over many of the state-of-the-art algorithms. In addition, we combined the algorithm with a segmentation algorithm and obtained encouraging results for recognizing objects in natural images [6, 20].

2. Related work

A key issue in shape analysis is finding a good representation. Due to the lack of a common dimension for shape, traditional classification algorithms are hard to directly apply. Most of the existing methods for shape classification are essentially matching-based, on either contour or skeleton. Skeleton-based approaches [29, 2, 1, 27] have shown promising results. Some representative contour based approaches include: Belongie et al. [4] which matches sampled points using context features; Latecki and Lakämper [19] which uses simplified polygons by Discrete Curve Evolution (DCE); and Ling and Jacobs [21] which builds inner-distance on top of the shape context.

On the one hand, a shape class may have large variations due to pose, deformation, and self-occlusion. It may require a lot of training samples to faithfully represent its distribution. On the other hand, different shape instances may share similar parts and the variation can be covered by using different configurations of these parts. The idea of using local and global parts/features has been used in several matching based algorithms [14, 24].

One related work to ours is Sun and Super [31] where shape classification is performed based on contour segments (parts) only. Our algorithm, however, is more general and we give theoratical justification to our classifier. We show that other features, such as inner-distance [21] and shape context [4], can also be incorporated. In the experiments, we demonstrate a large improvement over the method in [31]. Our algorithm is built on top of several existing methods such as DCE [19] and [3] (for extracting skeleton), [2] (for computing skeleton paths), [31] (for computing contour segments).

3. Shape classification

Given a number of classes with each class having a set of training shapes, we can use various techniques to perform classification. Here, one thing we want to emphasize is that *representation* is key for shape. A successful shape classifier has to address the unique property of shape.

Traditional classifiers such as Boosting [15] and SVM [32] have been shown to be very effective in performing feature selection and fusion. One requirement for these algorithms to work, however, is that the extracted features have to be ordered, which is very difficult in shape classification. Using boosting gives poor classification results on our dataset.

3.1. A generative model

For a shape class l, ideally, one would faithfully study its manifold to understand its instance $S^{(l)}$. However, studying the manifold of a general shape class with large deformation and variation in configuration is an extremely difficult task. Alternatively, one can define its likelihood by

$$p_l(S) = \int\int p(S|S^{(l)}, T)p(T)p(S^{(l)})dTdS^{(l)}, \qquad (1)$$

where S is a query shape, T is the underlying transformation, and $p(S|S^{(l)}, T)$ is the likelihood. To simplify the notation, we assume equal prior on the Ts and $S^{(l)}$s.

$$p_l(S) = \int p(S|S^{(l)}, T)dTdS^{(l)} = \frac{1}{Z_l}\int \exp\{-D(S, T(S^{(l)}))\}dTdS^{(l)}, \qquad (2)$$

where $D(S, T(S_l))$ is a similarity measure between S and shape $S^{(l)}$ deformed by T, and Z_l is the normalization function making $p_l(S)$ directly comparable on different ls. There are three major obstacles in computing eqn. (2): (1) measuring similarity D often requires performing explicit shape matching between S and $T(S^{(l)})$, which is difficult and computationally expensive to do; (2) all the instances in the space of l with all the corresponding transformations need to be checked; (3) it is very hard to compute the Z_l.

While a shape looks globally different after being deformed, their local parts often observe a certain degree of invariance. Now, we assume that some invariant shape features can be extracted for $S = (s_1, ..., s_M)$, where M is the number of features. These features are of the same type and directly comparable, e.g. different contour segments. We will give more detailed description later. Likewise, $S^{(l)} = (s_1^{(l)}, ..., s_{M(S^{(l)})}^{(l)})$. Note that different shapes often have different number of features. Let $SS^{(l)} = \{S_k^{(l)}, k = 1..K\}$ be a training set of K shapes. We collect all the features for each shape $S_k^{(l)}$ and put them together into a new set $\{s_j^{(l)}, j = 1..N\}$. If we assume the invariance of extracted features w.r.t. T and the independence among them, eqn. (2) can be simplified to

$$p_l(S) \approx \prod_{i=1}^{M} \frac{1}{N} \sum_j G(D(s_i, s_j^{(l)}); \sigma), \tag{3}$$

where $D(s_i, s_j^{(l)})$ measures the similarity between s_i and $s_j^{(l)}$, G is a truncated normal distribution since D is always positive. This approximation using non-parametric presentation allows us to avoid computational difficulty in eqn. (2): (1) every feature of query shape is measured against the features in the training set, which is much more efficient than performing shape matching for all shapes (one can perform clustering to reduce the number features); (2) explicit transformation is not computed; (3) normalization is guaranteed and partition function Z_l is no longer needed.

3.2. Duality of contour and skeleton

Given a 2D contour, one can compute its skeleton; likewise, we can obtain the contour from the skeleton with the information about its corresponding maximal disks. Therefore, contour and skeleton observe duality of a 2D shape, and we can always derive one from the other. If we compute features from both contour and skeleton, they are seemingly redundant. However, how accurately eqn. (3) approximates eqn. (2) is decided by how informative and invariant the extracted features are. In this regard, features extracted from contour and skeleton are quite complementary to each other: skeleton features are often robust against articulation and non-rigid transformation; contour features are stable and informative w.r.t. global and affine transformation. This observation motivates us to integrate contour and skeleton information together. So long as the integration is well justified (we use classification error as the objective function in this paper), we can argue that it is a principled approach to combine two types of information.

4. Shape model

In this section, we give the details of our shape model. Let S be a shape and we use $\mathcal{A}(S)$ to denote two sets of typical features computed from S as

$$\mathcal{A}(S) = \{\mathcal{C}(S), \mathcal{H}(S)\}, \tag{4}$$

where $\mathcal{C}(S)$ denotes the set of **contour segments** on S, and $\mathcal{H}(S)$ represents the set of **skeleton paths** (note that the feature set is not limited by these two types only).

4.1. Contour segments

Contour segments were used in [31], but here we extract them based on Discrete Curve Evolution (DCE) [19] instead of curvatures [31]. Below, we briefly discuss how to extract them. Let $S(t) = (x(t), y(t))$ be the contour of S parameterized by $t \in [0, 1]$. We first apply the DCE method to obtain a simplified polygon on S with the vertices denoted as

$$\vec{u} = (u_1, ..., u_N),$$

where N denotes the number of vertices, which is not known *a prior* and is automatically computed. $\vec{u}$ are the salient points on S, also called **critical points**. Fig. (2.b) shows a simplified polygon computed for an input shape S. The critical points are then mapped back to the original shape and are displayed in Fig. (2.c).

Contour segments $\mathcal{C}(S)$ are extracted for all pairs of critical points (u_i, u_j) as

$$\mathcal{C}(S) = \{c_{i,j} = (u_i, u_j), i \neq j, i, j \in 1..N\}.$$

It is noted that u_i and u_j do not have to be adjacent to each other. Also,

$$S = c_{i,j} \cup c_{j,i},$$

since one represents a segment and the other is its counterpart. In our implementation, we remove straight lines from the set of contour segments, $c(S)$, since they are not so informative. Two thick lines in Fig. (2.c) show two contour segment examples.

4.2. Skeleton paths

In this section, we discuss how to extract skeleton (medial axis), and thus the skeleton paths. One critical issue about skeleton extraction is that the procedure is often sensitive to deformation and noise on the boundary. Therefore, pruning the redundant skeleton branches becomes essential [27, 18]. Given a 2D shape, we apply the pruning algorithm

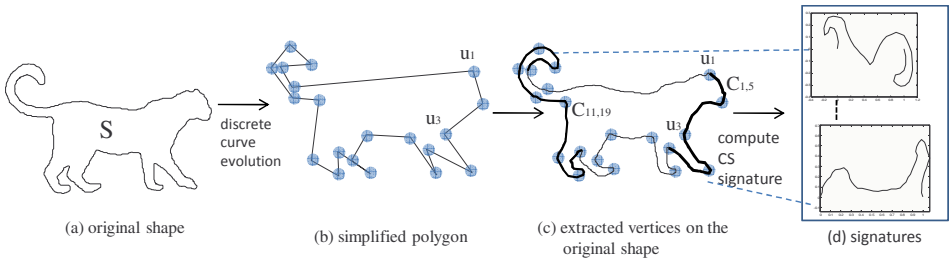


Figure 2. (a) is a contour of a leopard. (b) is the simplified polygon of (a) by DCE [19]. (c) shows the critical points on the contour, which are the vertices of the polygon in (b). (d) displays two signatures for the contour segments (shown in thick lines).

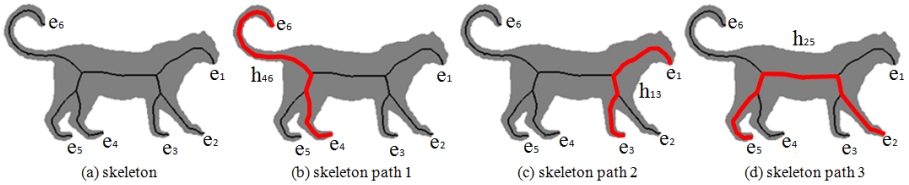


Figure 3. (a) is the skeleton of a leopard extracted by algorithm [3], on which $e_1, e_2, ..., e_6$ are the skeleton endpoints. (b), (c), (d) show the three skeleton paths (in red) between three pairs of endpoints $(e_4, e_6), (e_1, e_3), (e_2, e_5)$ separately.

[3] to extract the skeleton (we use their publicly available code). Fig. (3.a) shows an example of an extracted skeleton.

Skeleton-based shape matching and recognition algorithms have been heavily studied and some representative works are [29, 17, 33]. However, even for the same object, the topology of its 2D skeletons might vary a lot. In shock graph [29], for example, a big effort was devoted to define the topology change for the skeleton points, even though the 2D shapes might not appear so drastically different. Instead, we focus on the paths of extracted skeleton, which are more direct and robust to compute.

For denotational clarity, we briefly discuss some general definitions first: a skeleton point having only one adjacent point is an **endpoint** (the skeleton endpoint); a skeleton point having three or more adjacent points is a **junction point**; if a skeleton point is not an endpoint nor a junction point, it is called a **connection point**.

Let $K(S)$ be the skeleton of S, and $e(S) = \{e_i, i = 1..M\}$ be all the endpoints of $K(S)$. M is the total number of endpoints. Fig. (3.a) shows a skeleton with 6 endpoints. A **skeleton path** $h_{i,j} = (e_i, e_j)$ is the shortest path between the two end points e_i and e_j. Fig. (3.(b-d)) show three skeleton paths. Note that in this paper, $h_{i,j} = (e_i, e_j)$ and $h_{j,i} = (e_j, e_i)$ are considered two different skeleton paths since they have different directions. Thus,

$$\mathcal{H}(S) = \{h_{i,j} = (e_i, e_j), i \neq j, i, j \in 1..M\}. \quad (5)$$

Skeleton paths can often be very efficiently computed and they are very informative. This is due to two reasons: 1) the instability of the junction points of skeleton is avoided in skeleton path; 2) different shape instances for the same object usually have very similar sequences of radii (the radius of maximal disks with the centers on the skeleton path) even though the object might have big deformation due to articulation and non-rigid transformation. Fig. (1.b) shows an illustration, and skeleton paths have been successfully used for shape matching in [2].

4.3. Normalization

The previous sections describe how contour segments and skeleton paths are computed. One thing we have not talked about is how to normalize them. Some factors, e.g. global scale and rotation, are arguably not intrinsic to shape understanding.

To perform contour segment normalization, we compute the shape signature [22] to achieve invariance to planar similarity transformations (2-D translation, rotation, and uniform scaling). Each contour segment c is sampled with n equal distance points $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$. We use $n = 50$ for all the results in this paper. Then, c is transformed to $\vec{v}$, which we call signature. The transformation is done by mapping $\mathbf{x}_1$ to $\mathbf{x}'_1 = (0, 0)$ and $\mathbf{x}_n$ to $\mathbf{x}'_n = (1, 0)$, which allows us to compute the planar similarity transform mapping the remaining points to $\mathbf{x}'_2, ..., \mathbf{x}'_{n-1}$ in the normalized reference frame. In Fig. (3.d), the signatures for the two contour segments are computed.

Each signature $\vec{v}$ is then represented by a vector $\vec{v} = (x_1, y_1, ..., x_n, , y_n)^T$. Finally, we define the set of all signatures as

$$\mathcal{V}(S) = \{\vec{v}_1, ..., \vec{v}_{|\mathcal{C}(S)|}\},$$

where $|\mathcal{C}(S)|$ is the total number of contour signatures.

Next, we discuss how to normalize skeleton paths. For each skeleton path h, we sample n number of equal distance points $\{\mathbf{y}_1, ..., \mathbf{y}_n\}$. Let $r(\mathbf{y}_i)$ denote the radius of the maximal disk at the skeleton point i. A vector of the radii of the maximal disks is denoted as $\vec{r} = (r(\mathbf{y}_1), r(\mathbf{y}_2), ..., r(\mathbf{y}_n))$.

In this paper, the radius $r(\mathbf{y}_i)$ is approximated by the value of distance transform $DT(\mathbf{y}_i)$ at each sample skeleton point $\mathbf{y}_i$. Suppose there are N_0 skeleton points $k_j, j = 1..N_0$ of $K(S)$. To make each r invariant to the scale, we

normalize it by

$$\vec{r} = \frac{DT(\mathbf{y}_i)}{\frac{1}{N_0}\sum_{j=1}^{N_0} k_j}. \quad (6)$$

Let

$$\mathcal{R}(S) = \{\vec{r}_1, ..., \vec{r}_{|\mathcal{H}(S)|}\}$$

denote the set of radius vectors $\vec{r}_i$ corresponding to all the skeleton paths in K.

4.4. Models

The feature set for a shape S is $\{\mathcal{V}(S), \mathcal{R}(S)\}$. For a set of training shapes of the same class SS, we collect two sets

$$\mathcal{V}(SS) = \bigcup_{1}^{|SS|} \mathcal{V}(S) \ \ and \ \ \mathcal{R}(SS) = \bigcup_{1}^{|SS|} \mathcal{R}(S),$$

where $|SS|$ is the number of training shapes. As discussed in section (3.1), we use non-parametric representation

$$p(\vec{v}) = \frac{1}{|\mathcal{V}(SS)|} \sum_{j=1}^{|\mathcal{V}(SS)|} G(D(\vec{v}, \bar{\vec{v}}_j), \sigma). \quad (7)$$

For the skeleton paths, we use

$$p(\vec{r}) = \frac{1}{|\mathcal{R}(SS)|} \sum_{j=1}^{|\mathcal{R}(SS)|} G(D(\vec{r}, \bar{\vec{r}}_j), \sigma), \quad (8)$$

where the distance measure can be of many forms, e.g.

$$D(\vec{r}, \vec{r}_j) = \sum_{t=1}^{n} \frac{(\vec{r}(t) - \vec{r}_j(t))^2}{\vec{r}(t) + \vec{r}_j(t)}. \quad (9)$$

In the implementation, KD-tree [5] can be adopted for efficient computation. Each time, only K nearest neighborhoods are retrieved to approximate eqn. (7) and (8). The contour and skeleton models are learned sepertately.

4.5. Combining contour and skeleton information

One choice is to directly take the average of the eqn. 7 and eqn. 8, which is essentially a voting scheme of nearest neighborhood classifier on each individual feature. As stated before, given a shape S, we can extract its feature set

$$\{\mathcal{V}(S) = \{\vec{v}_i\}, \mathcal{R}(S) = \{\vec{r}_i\}\}.$$

Our classifier is simply

$$l^* = \arg min_l - \{\alpha \sum_i \log p_l(\vec{v}_i) + (1-\alpha) \sum_i \log p_l(\vec{r}_i)\}, \quad (10)$$

where l denotes shape class, α, which is learned, balances the importance between contour segments and skeleton paths. $p_l(\vec{v})$ and $p_l(\vec{v})$ are the models described in eqn. (7) and eqn. (8) respectively. Eqn. (10) is a realization of eqn. (3), but with two types of features. The balancing of the two is done by minimizing the classification error in a coherent objective function. Therefore, the integration of contour and skeleton information is justified. Our approach has some interesting properties: (1) It explicitly combines both the contour and skeleton information. (2) Shape parts across different scales are automatically integrated in both training and testing, since features from all pairs of critical points are used. Also, we do not have to search for the scale of the parts. (3) It is invariant to translation, rotation, and global scale change. It is also robust, to a certain degree, against articulation and non-rigid transformation. This is due to the use of skeleton paths, which are insensitive to non-rigid transformation. (4) No procedure for matching and alignment is needed. (5) It is robust against both self-occlusion and missing parts. The basic theoretical justification for our model is given in sect. (3.1). The resulting final classifier is a simple formulation. It is also somewhat related to the recent ensemble learning [7] where classification is done by voting an ensemble of features.

5. Outline of our algorithm

We give the outline of our shape classification below.

Training: (1) Collect a set of training shapes of several classes. (2) For each training shape, discrete contour evolution (DCE) method [19] is applied to extract simplified polygons with its corresponding vertices. (3) Contour segments are then extracted from all pairs of vertices on the input shape, followed by the computation of signatures. (4) Collect all the signatures from all the shapes in the class. (5) The skeleton is computed for each shape using [3]. (6) Collect all the skeleton paths and perform normalization. (7) Learn classifier eqn. 10.

Testing: (1) Given an input shape, compute its signatures and radius sequences using the same procedures as above. (2) Compute the value using eqn. (10) and assign the shape with the class label that achieves the smallest value. (If the discriminative classifier is used, then the features are sent to the classifier, e.g. AdaBoost, to perform classification.)

6. Shape database

Many of the recently established datasets in computer vision have greatly inspired the development of working systems for object detection, segmentation and recognition. Some typical ones include the Berkeley [23], PASCAL [12], MSRC [28], Caltech-101 [13], Weizmann [6], and LabelMe [25] datasets. Despite some ongoing debates about how representative these datasets are, they nevertheless have provided objective benchmarks for the algorithm evaluations. In shape analysis, several datasets have been widely used such as the Kimia [26], MPEG-7 [19], Aslan and Tari [1] datasets. In particular, many algorithms [4, 21, 14] have

reported matching results on the MPEG-7 dataset, in which there are 70 categories of shape with each having 20 shapes. Inspired by the efforts to establish benchmark datasets, we collected a dataset of 20 classes $\times$ 100 shapes. We are motivated by the following observations: (1) the common image datasets for recognition and detection are not focused on shape; (2) datasets obtained by web-based interactive tool, such as LabelMe, rely on random users to provide segmentation, and the quality is often not satisfactory; (3) the Berkeley dataset [23] is focused on regions instead of objects; (4) the MPEG-7 dataset [19] contains too few shapes in each category, and it is evident that the large variation of object shape is not represented. We define 20 classes of animals, namely: Bird, Butterfly, Cat, Cow, Deer, Dolphin, Duck, Elephant, Crocodile, Fish, Flying Bird, Chicken, Horse, Leopard, Monkey, Mouse, Spider, Tortoise, Rabbit, and Dog. All of these shapes are obtained for objects from real images.

7. Experimental results

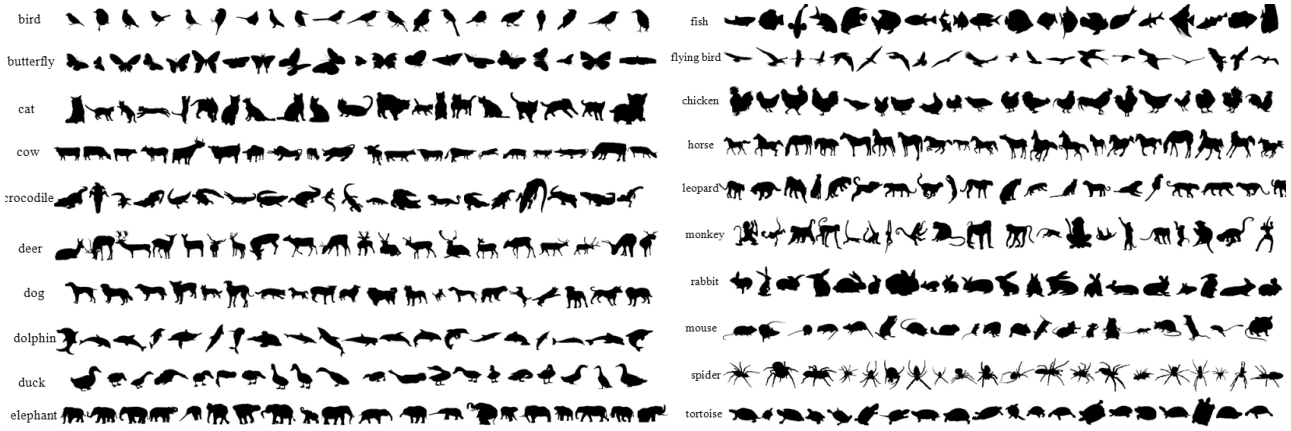
First, we illustrate our proposed algorithm on the MPEG-7 dataset, which has $70 \times 20 = 1,400$ shapes. Since the classifier does not depend on the specific choices of features, we show the classification results using contour segments (CS), skeleton paths (SP), and both. This experiment is to demonstrate the effectiveness of using the combination of contour and skeleton cues. We randomly sample 10 shapes from each class for training and 10 for testing. The results are obtained by averaging a couple of trials. Table (1) shows the classification results using the three choices. The method in [31] gives rise to a score of 90.9%. Our classification algorithm based on contour segments achieves 91.1%, and the final one combining contour and skeleton improves to 96.6%. It is apparent that using both the contour and skeleton cues greatly improve the results over using either one of them alone. Our method voids matching and is 10-fold faster than matching based algorithm, such as shape context [4]. It usually takes about 8 seconds on a modern PC for our shape classification algorithm whereas it takes about $1 \sim 2$ minutes for a matching based algorithm. The speed of our algorithm can be further improved.

Our algorithm achieves 96.6% classification rate on the MPEG-7 dataset of 70 classes. This suggests that the MPEG-7 dataset might not be sufficient to judge the quality of a shape classification algorithm. Therefore, we conducted another set of evaluation on our dataset of 20 animals of 100 shapes each. These shapes have much larger intra-class variations and inter-class similarities than the MPEG-7 dataset. It is noted that there is severe self-occlusion of the shapes in this database. We performed a comparison to the state-of-the-art algorithms. Like in the previous experiment, we randomly split the shapes into half for training and half for testing. One direct comparison is to see how matching-based algorithm performs on this task. This is done by computing the distance between a query shape to all the shapes in the training set, and the query shape is given the label of the closest matched shape. We use inner-distance (IDSC) [21], which improves over the well-known shape context algorithm [4]. The overall classification rate is 73.6% which is worse than 78.4% and 78.7% by our final system. So a state-of-the-art matching-based algorithm not only gives worse result for classification, but is also slower (we downloaded the source code provided by the authors and obtained similar results for matching on the MPEG-7 dataset as reported in [21]; this suggests that it is less likely that we are using a different implementation of the algorithm). Since the features for our classifier is not limited by what is described in the paper, we could directly use inner-distance as features which we call IDSC-F. The features in IDSC-F are those of inner-distance shape context (IDSC) on the nodes by DCE. Using IDSC-F yields a classification rate of 55.7%. Table (2) summarizes the choices of using different feature sets and the ones using combination of contour segment, skeleton path, and inner-distance yields the best score. However, it is only slightly better than that using CS and SP. We also tried the algorithm in [30] and the results achieved are not comparable to those in table (2).

To further show how the contour and skeleton cues are complementary to each other, we show the numbers of each individual class of using contour, skeleton, and both in Table (3). Using both the cues gives rise to the best result in all the cases. Two cases are interesting to be pointed out: For the fish class, contour cues are much better than the skeleton cues. This is understandable since fish observe elongated and symmetric structures. For the mouse class, skeleton cue is much more informative than the contour cue. This is because mouse shape observes large variation in deformation which can be easily captured by skeleton, but is difficult for the contour segments.

All the above experiments are carried on manually extracted shapes. A question one might ask is how the proposed algorithm can be used to detect and recognize objects in real-world images. To validate this idea, we combine the Wizemann horse (328) dataset [6] and the ETHZ cow (112) dataset [20] and split them into half for training and half for testing. We use a learning-based algorithm [11] to extract the foreground in each image. The first two columns in Fig. (5) show some typical results and the other columns show difficult images with severe segmentation errors. Sometimes, only parts of the objects are successfully segmented. We perform binarization on the probability maps reported by the learning algorithm. The shape classification is then a two-class classification problem. Table (4) shows the classification results on automatically extracted shapes, which are 96.4% and 90.6% for the cow and horse respectively. We obtain the same conclusion as in the manually extracted

(a) some real images from which the object shapes are extracted



(b) some shape examples for the 20 animal classes in our dataset.

Figure 4. Our shape dataset.

Features	The method [31]	Contour Segments (CS)	Skeleton Path (SP)	CS & SP
Classification Rate	90.9%	91.1%	86.7%	**96.6%**

Table 1. Classification rates on the MPEG-7 dataset using [31], contour segment features, skeleton path features, and both. Using both types of features yields the best performance.

Features	CS [31]	IDSC [21]	IDSC-F	CS	SP	CS & IDSC-F	SP & IDSC-F	CS & SP	CS & SP & IDSC-F
Clas. Rate	69.7%	73.6%	55.7%	71.7%	67.9%	72.5%	73.1%	**78.4%**	**78.7%**

Table 2. Comparison of different classification algorithms using different types of features. The first result by [31]. IDSC is using the matching based inner-distance algorithm [21]. $IDSC - F$ is the maximum likelihood classifier based on the IDSC features. The rest are the rate by using various combinations of feature types.

	Bird	Butterfly	Cat	Cow	Deer	Dolphin	Duck	Elephant	Crocodile	Fish
CS	76%	89%	39%	70%	69%	87%	83%	95%	54%	70%
SP	55%	89%	37%	80%	65%	64%	79%	90%	60%	51%
CS & SP	**76%**	**93%**	**48%**	**80%**	**79%**	**89%**	**89%**	**97%**	**66%**	**74%**
	Flying Bird	Chicken	Horse	Leopard	Monkey	Mouse	Spider	Tortoise	Rabbit	Dog
CS	57%	89%	96%	56%	21%	52%	98%	83%	81%	69%
SP	35%	86%	77%	64%	33%	82%	94%	81%	72%	62%
CS & SP	**65%**	**94%**	**97%**	**65%**	**33%**	**84%**	**100%**	**90%**	**87%**	**75%**

Table 3. Detailed classification rate for each individual class. The contour and skeleton cues are quite complementary to each other. The fish and mouse classes are especially interesting.

shapes: our representation is effective, and combining both the contour and skeleton yields good performance. The algorithm is robust even though there are severe segmentation errors and it is not sensitive to the local noisy boundaries.

8. Conclusion and discussion

In this paper, we have introduced a new shape classification algorithm, and demonstrated significantly improved results over the state-of-the-art algorithms. There has been much less attempts for direct shape classification than matching in the past. This is due to a property of shape lacking good alignment. Finding the right representation and feature for shape analysis is very important. The strength of this paper lies in that: (1) it emphasizes the importance of combining contour with skeleton cues for shape analysis; (2) it explores the means of using shape features and shows that good classification results can be obtained; (3) it produces encouraging results on challenging shape datasets as well as shapes automatically extracted from natural images; (4) a practical shape dataset of challenging shapes is collected and reported. and (5) it can be combined with detection and segmentation to perform object recognition. One disadvantage of our approach is that we need a connected region for each shape, which might not always be available for some challenging cases.

Acknowledgements

This work is supported by Office of Naval Research Award, No. N000140910099. Any findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research. This work is also supported by EMDRFC 20070487028 and NSFC 60873127. We would like to thank Quannan Li, Chengqian Wu, Xiaojun Yang, Yifan Li, Jian Zhou, Jiaming

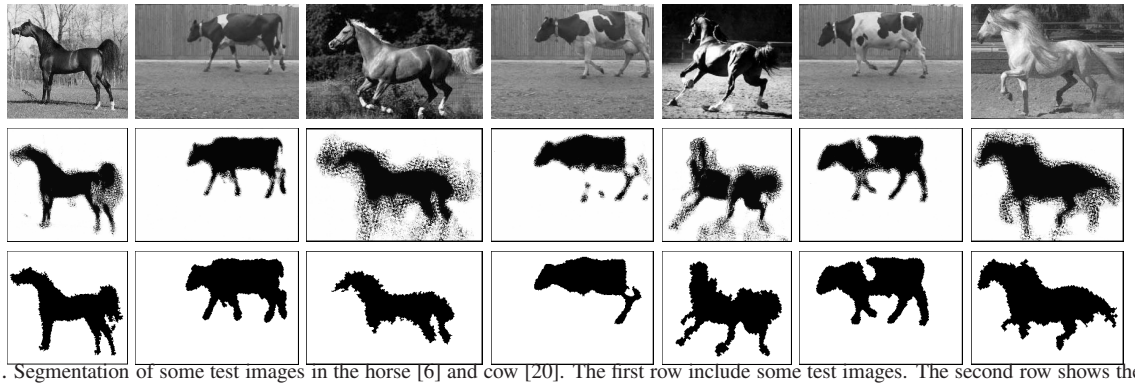


Figure 5. Segmentation of some test images in the horse [6] and cow [20]. The first row include some test images. The second row shows the probability maps reported by a learning based segmentation algorithm. The third row displays the binarization from the probability maps. The first two column are some typical results and the other columns show some unsatisfactory results of segmentation.

Features	Contour Segments (CS)	Skeleton Paths (SP)	CS & SP
Cow	89.1%	85.5%	**96.4%**
Horse	70.6%	87.1%	**90.6%**

Table 4. Classification results for horse and cow segmentation and recognition from real-world images.

Qiu, Zuodong Wu, Wei Xia, Chen Shen, Yanbo Xu, for their collecting and labeling the images of the animals dataset.

References

[1] C. Aslan, A. Erdem, E. Erdem, and S. Tari. Disconnected skeleton: shape at its absolute scale. *IEEE Trans. PAMI*, 2008.

[2] X. Bai and L. Latecki. Path similarity skeleton graph matching. *IEEE Trans. PAMI*, 30(7):1282–1292.

[3] X. Bai, L. Latecki, and W.-Y. Liu. Skeleton pruning by contour partitioning with discrete curve evolution. *IEEE Trans. PAMI*, 29(3):449–462, 2007.

[4] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. PAMI*, 24(4):509–522, 2002.

[5] J. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18:509–517, 1975.

[6] E. Borenstein, E. Sharon, and S. Ullman. Combining top-down and bottom-up segmentation. In *IEEE workshop on POCV*, volume 4, pages 46–53, June 2004.

[7] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[8] A. M. Bronstein, M. M. Bronstein, A. M. Bruckstein, and R. Kimmel. Analysis of two-dimensional non-rigid shapes. *IJCV*, 78:67–88, 2008.

[9] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *CVIU*, 89:114–141, 2003.

[10] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models–their training and application. *CVIU*, 61(1):38–59, 1995.

[11] P. Dollar, Z. Tu, and S. Belongie. Supervised learning of edges and object boundaries. In *CVPR*, 2006.

[12] M. Everingham, A. Zisserman, C. Williams, and L. V. Gool. The PASCAL Visual Object Classes Challenge. In *http://www.pascal-network.org/challenges/VOC/voc2006/*, 2006.

[13] L. Fei-Fei, R. Fergus, and R. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *CVIU*, 106(1):59–70, 2007.

[14] P. Felzenszwalb and J. Schwartz. Hierarchical matching of deformable shapes. In *CVPR*, Minneapolis, June 2007.

[15] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. of Comp. and Sys. Sci.*, 55(1):119–139, 1997.

[16] U. Grenander. *General Pattern Theory: A Mathematical Study of Regular Structures*. Oxford, 1994.

[17] R. Katz and S. Pizer. Untangling the blum medial axis transform. *Intl' J. of Com. Vis.*, 55(2).

[18] R. Kimmel, D. Shaked, N. Kiryati, and A. M. Bruckstein. Skeletonization via distance maps and level sets. *CVIU*, 62:382–391, 1995.

[19] L. Latecki and R. Lakämper. Shape similarity measure based on correspondence of visual parts. *IEEE Trans. PAMI*, 22(10):1185–1190, 2000.

[20] B. Leibe and B. Schiele. Analyzing appearance and contour based methods for object categorization. In *CVPR*, June 2003.

[21] H. Ling and D. Jacobs. Shape classification using the inner-distance. *IEEE Trans. PAMI*, 29(2):286–299, 2007.

[22] S. Loncaric. A survey of shape analysis techniques. *Pattern Reco.*, 31(8):983–1001, 1998.

[23] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *IEEE Trans. PAMI*, 26(5):530–549, 2004.

[24] G. McNeill and S. Vijayakumar. Hierarchical procrustes matching for shape retrieval. In *CVPR*, June 2006.

[25] B. Russell, A. Torralba, K. Murphy, and W. Freeman. Labelme: a database and web-based tool for image annotation. *Intl' J. of Com. Vis.*, 2007.

[26] T. Sebastian, P. Klein, and B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Trans. PAMI*, 26(5):550–571, 2004.

[27] D. Shaked and A. Bruckstein. Pruning medial axes. *CVIU*, 69(2):156–169, 1998.

[28] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint app., shape and context modeling for multiclass object recognition and segmentation. In *ECCV*, 2006.

[29] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shock graphs and shape matching. *Intl' J. of Com. Vis.*, 35(1):13–32, 1999.

[30] A. Srivastava, S. Joshi, W. Mio, and X. Liu. Statistical shape analysis: clustering, learning, and testing. *IEEE Trans. PAMI*, 27(4):590–602, 2005.

[31] K. Sun and B. Super. Classification of contour shapes using class segment sets. In *IEEE CVPR*, June 2005.

[32] V. Vapnik. *Estimation of dependences based on empirical data*. Springer-Verlag.

[33] S. C. Zhu and A. L. Yuille. Forms: A flexible object recognition and modeling system. *Intl' J. of Com. Vis.*, 20(3):1573–1405, 1996.