Shape Categorization Using String Kernels

Mohammad Reza Daliri[1], Elisabetta Delponte[2],
Alessandro Verri[2], and Vincent Torre[1]

[1] SISSA, Via Beirut 2-4, 34014 Trieste, Italy
[2] DISI, Universita degli Studi di Genova, Via Dodecaneso 35, 16146 Genova, Italy

Abstract. In this paper, a novel algorithm for shape categorization is proposed. This method is based on the detection of perceptual landmarks, which are scale invariant. These landmarks and the parts between them are transformed into a symbolic representation. Shapes are mapped into symbol sequences and a database of shapes is mapped into a set of symbol sequences and therefore it is possible to use support vector machines for categorization. The method here proposed has been evaluated on silhouettes database and achieved the highest recognition result reported with a score of 97.85% for the MPEG-7 shape database.

1 Introduction

The final goal of computer vision is to make machines as capable as humans in terms of visual perception and understanding [23]. Object recognition and classification has been extensively studied and analyzed in recent years, but current techniques are far from needed. An even more difficult task for a machine is to determine the category to which the object belongs, rather than to find out whether or not that particular object has been seen before. There are several reasons that make this problem so difficult. The first reason is related to the uncertainty about the level of categorization in which recognition should be done. Based on the research made by cognitive scientists [9], there are several levels at which categorization is performed. Another reason is the natural variability within various classes. The generality of a class is directly proportional to the within-class variation. Moreover, the characterization should be invariant to rotation, scale, translation and to certain deformations. Objects have several properties that can be used for recognition, like shape, color, texture, brightness. Each of these cues can be used for classifying objects. Biederman [4] suggested that edge-based representations mediate real-time object recognition. In his view, surface characteristics such as color and texture can be used for defining edges and can provide cues for visual search, but they play only a secondary role in the real-time recognition. There are two major approaches for shape-based object recognition: 1) boundary-based, that uses contour information [5], [20], [16], [3], [1], and 2) holistic-based representation, requiring more general information about the shape [18], [17]. In this paper, a new representation for categorization based on the extraction of the perceptually relevant landmarks is proposed. Each shape is transformed into a symbolic representation, where each shape is mapped in a string of symbols. The present manuscript is organized as follows: Localization and extraction of landmarks are investigated in Section 2. The symbolic representation is presented in Section 3. Section 4 describes the feature space composed by string kernels. In Section 5 geometrical invariants features are described. The results are presented in Section 6.

2 Extraction and Localization of Landmarks

A database of black shapes over a white background (Silhouettes) was used [21] (Figure 1). In this case the extraction of the contour is straightforward and it is represented by the edge chain (x(j),y(j)) j=1,...,N where N is the chain or contour length. The next step is finding the gradient of the contour at the optimal scale. As suggested by Lindeberg [13], the local scale can be estimated considering the normalized derivatives: $G_\lambda = t^{\lambda/2}\sqrt{L_x^2 + L_y^2}$.

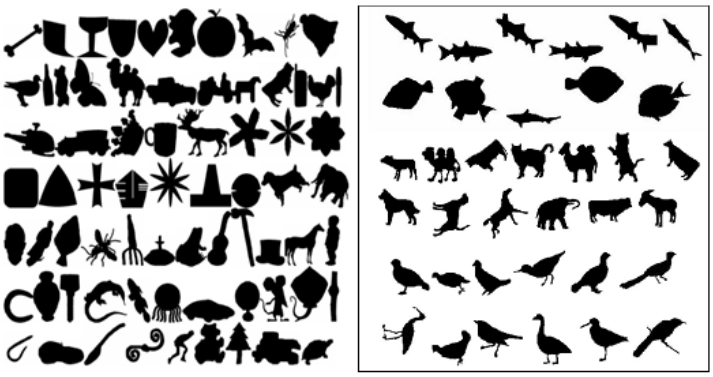


Fig. 1. Some sample shapes from MPEG7 database and Kimia database used in our experiments

Where L_x and L_y are the x and y derivatives of the original image convolved with the Gaussian filter $exp(-(x^2+y^2)/2t)$ with $t = \sigma^2$. These normalized derivatives $G_\lambda(t)$ depend on the value of the parameter λ. As discussed by Lindeberg [13] and Majer [15], the most convenient choice for the Gaussian step edges is $\lambda = 1/2$. The best scale was extracted with the Lindeberg formula and the gradient at this scale was computed by a simple 2-D gaussian filtering in X and Y direction in the image plane, $\overrightarrow{G} = (G_x, G_y)$. As the tangent vector is orthogonal to the gradient vector we obtain at the best scale, $\overrightarrow{T} = (T_x, T_y) = (G_y, -G_x)$. The curvature κ of a planar curve at a point P on the curve is defined as the instantaneous rate of change of the tangent's slope angle at point P with respect to arc length s: $\kappa = \frac{\partial \overrightarrow{T}}{\partial s}$. In order to calculate the derivative of each component of the tangent vector we convolve them by the first derivative of one dimensional Gaussian function:

$$\frac{\partial T_x}{\partial s} = \frac{\partial [T_x \otimes g(s,\sigma)]}{\partial s} = T_x \otimes [\frac{\partial g(s,\sigma)}{\partial s}] \tag{1}$$

so that: $g(s,\sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp{(-\frac{s^2}{2\sigma^2})}$.

This will be done for the Y component of the tangent vector. Because different shapes in our database have different length, it would be better to select a sigma related to the length of a shape contour, to formulate this statement we choose our sigma to be: $\sigma_1 = \sigma_0 \frac{l}{l_0}$, where l is the length of contour shape and, based on our experiment, we select $l_0 = 200$ and $\sigma_0 = 3$. Now, the curvature value will be calculated as follows: $\|\kappa\| = \sqrt{(\frac{\partial T_x}{\partial s})^2 + (\frac{\partial T_y}{\partial s})^2}$.

For having the complete calculation of the curvature we need to attribute a sign to it. Direction of tangent vector is a good representation for calculating the sign of curvature but it must be smoothed. We applied convolution to each component of the tangent vector with a one dimensional Gaussian function with $\sigma = 3$ for the small smoothing of the tangent vector to remove the noise. Now, with the smoothed tangent vector we can calculate the sign of curvature as follows: $Sign(\kappa) = sign[(T_{x,sm}(s), T_{y,sm}(s), 0) \times (T_{x,sm}(s-1), T_{y,sm}(s-1), 0)]$.

The complete definition of our curvature will be obtained by multiplying the value of curvature with its sign. The obtained curvature is noisy and in order to reduce it a non-linear filtering was used. The aim of the non-linear filtering was to smooth regions of low curvature and to leave unaltered regions of high curvature. We first compute the local square curvature as:

$$\overline{\kappa^2(n)} = \frac{1}{2\sigma_1 + 1} \sum_{i=-\sigma_1}^{\sigma_1} \kappa^2(n+i) \tag{2}$$

Non-linear filtering is performed by convolving the curvature with a one-dimensional Gaussian function, where the scale of filter is:

$$\sigma_2(n) = \sigma_{min} + \frac{\hat{\kappa}}{\overline{\kappa^2(n)}} \tag{3}$$

In our experiment good results were obtained by using the values of $\sigma_{min} = 0$ and $\hat{\kappa} = 0.02$. In this way a robust and perceptually relevant representation for the curvature of the shapes was obtained. Now, the local maxima (negative and positive peaks) of the curvature are detected and identified as landmarks in the original 2-D contours(Figure 2).

3 Symbolic Representation

In this section we will transform each shape into symbolic representation to be used for categorization. Firstly angles close to 180 degrees are removed. In what follows a "dictionary" is presented, allowing the transformation of the curvature representation into a string of symbols.

3.1 Labeling of Angles

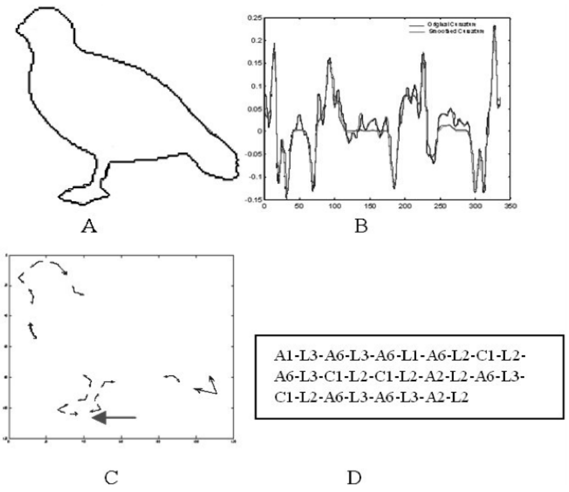
Features detected as corners are quantized so that angles have either 45 or 90 or 135 degrees. These angles can have either a positive or a negative value of curvature. A total of 6 different corners are obtained which can be labeled as A1, A2 , ... up to A6.

3.2 Labeling of Curves

Curve parts have the average curvature between straight lines and sharp angles that with setting a threshold can be found. Curves are labeled either as concave (C1) or convex (C2), according to the sign of their average curvature.

3.3 Labeling Links Between Angles (and Curves)

Pieces of the contour of silhouettes linking two corners (or curves) are labeled in three ways: L1 if it is a straight line, L2 if it is not a straight line (and it is not a curve) but has an average positive curvature and L3 if, on average, has a negative curvature.



A1-L3-A6-L3-A6-L1-A6-L2-C1-L2-
A6-L3-C1-L2-C1-L2-A2-L2-A6-L3-
C1-L2-A6-L3-A6-L3-A2-L2

Fig. 2. A) A contour of shape from our database with associated numbers based on an arbitrary starting-point in the contour. B) Curvature profile and smoothed one in order to have perceptually relevant peaks as described in the text. C) Angle representation based on the maxima and peaks information of the curvature representation. D) Symbolic representation for the bird shape.

4 Creating the Feature Space

In our approach, shape categorization becomes similar to text categorization, where each string of symbols can be either a sentence or a separate document. A standard approach [11] to text categorization uses the classical text representation [19], which maps each document into a high-dimensional feature vector, where each entry of the vector represents the presence or the absence of a feature. Our approach makes use of specific kernels [14]. This specific kernel named string kernel, maps strings, i.e. the symbolic representation of the contour obtained in the previous section, into a feature space. In this high dimensional feature space all shapes have the same size. This transformation provides the desired rotational invariance and therefore the categorization system is also invariant to the initial symbol of the string describing the shape. The feature space in this case is composed by the set of all substrings of maximum length L of k-symbols. In agreement with a procedure used for text classification [14], the distance and therefore the similarity between two shapes is obtained by computing the inner product between their representations in the feature space. Their inner product is computed by making use of kernel functions [14], which compute the inner product by implicitly mapping shapes to the feature space. In essence, this inner product measures the common substrings of the symbolic representations of the two shapes: if their inner product is high the two shapes are similar. Substrings do not need to be contiguous, and the degree of contiguity of one substring determines its weight in the inner product. Each substring is weighted according to its frequency of appearance and on its degree of compactness, measured by a decay factor, λ, between (0,1) [14]. To create the feature space we need to search all possible substrings starting from each single-symbol to strings of length L composed by k symbols, which in our case are the 11 symbols introduced in Section 3. For each substring there is a weight in the feature space given by the sum of all occurrences of that sub-string considering the decay factor for non-contiguity. After creating the invariant feature space, we need to use a classifier to find the best hyper-planes between the different classes. Support Vector Machines (SVM) are a very successful class of statistical learning theory in high dimensional space [24]. For classification, SVMs operate by finding a hyper-surface in the space of possible inputs. In their simplest version they learn a separation hyper plane between two sets of points, the positive examples from the negative examples, in order to maximize the margin -distance between plane and closest point. Intuitively, this makes the classification correct for testing data that is near, but not identical to the training data. Further information can be found anywhere such as [6], [8].

5 Geometric Invariant Features

Beside the high dimensional feature space described in the previous section, a set of geometrical properties for each shape were measured. They consist of 16 different numbers that are normalized so to be invariant for rotation and size

Table 1. Some of the Geometric Invariant Features

Geometric Feature	Definition
Roughness	Perimeter/Convex Perimeter
Compactness or Circularity	$(Perimeter^2)/(4 * \pi *$Area of the shape$)$
Solidity	Number of pixels in the convex hull/the number of shape points
Rectangularity	Number of pixels in the bounding box/ the number of shape pixels
Normalized Major Axis Length	The length of the major axis of the ellipse that has the same second-moments as the shape
Normalized Minor Axis Length	The length of the minor axis of the ellipse that has the same second-moments as the shape
Elongation	Major Axis Length/Minor Axis Length
Normalized Equivalent Diameter	The diameter of a circle with the same area as the shape
Eccentricity	The ratio of the distance between the foci of the ellipse and its major axis length

transformation. Table 1 illustrates some of these geometrical features. For further information we refer readers to [7].

6 Experimental Results

In this section, some experiment results aiming at evaluating and comparing the proposed algorithm for shape classification will be presented. Firstly, a database extracted from Kimia's silhouette database [21] was used. Three different categories were considered composed by the category of birds consisting of 51 shapes, the category of mammals consisting of 178 shapes and the category of fish consisting of 79 shapes. Some shapes of the database were rotated and resized. We used LIBSVM [10] tools that support multi-class classification. To test the success of our classification the cross-validation leave-one-out method was used. In the first experiment the feature vector is created without inserting any information about the distance from the 2-D image of each shape. Different kernel functions with different parameters have been tested to reach the best result, but a simple linear kernel was the best. As discussed in section 4 it is possible to consider feature vectors with different maximum length of symbols and therefore we compared results obtained with categorization based on substrings with a maximum length of 3 and 4 symbols. As shown in Table 2 successful categorization

Table 2. Comparison of different maximum lengths for searching substring based on the classification rate($\lambda = 0.5$)

	Bird	Mammal	Fish
Substring with maximum length of 3	64.7%	87%	86%
Substring with maximum length of 4	66%	88.7%	84.8%

Table 3. Classification rate for the best parameter of $\lambda(0.3)$ after inserting the distance information between landmarks

Bird	Mammal	Fish
93.61%	92.69%	86.07%

Table 4. Classification rate for combined-features

Bird	Mammal	Fish
96.8%	97.75%	96.2%

increases with longer substring, but not so much to justify a significantly heavier computational load. As shown in Table 2, successful categorization for birds is worse than for mammals and fish, as there was a higher inter-class variation for birds with less number of samples. The database was enriched by creating new shapes by re-scaling (up to 1.5) and rotating, flipping and mirroring some of the bird shapes. After inserting new bird shapes, this category was consisting of 94 different bird images. The result was improved as shown in Table 3. In the second set of experiments we introduced also information on the distance between landmarks and different decaying factor (λ) similar to that used for text categorization [13] was tested. The best value for the parameter λ was equal to 0.3 and we set it to this value for further experiments (Table 3). Features obtained from the curvature do not catch important geometrical features of the shape to be categorized and therefore categorization based on mixed features was considered. Table 1 illustrates 17 different geometrical features which were computed for every shape and were added to the vector feature. These geometrical features consist of 17 different features such as roughness, elongation, compactness, rectangularity, convex area,..., that has been normalized so to have features invariant for size and rotation transformations. Table 4 illustrates results from cross-validation leave-one-out method combining geometric invariant and feature vector derived by string kernel. Finally the proposed categorization method was tested also on large shape database MPEG-7 CE-Shape-1 [12] consisting of 70 types of objects each having 20 different shapes. Geometrical invariant features listed in Table 1 were combined with feature vectors derived by string kernels. For the experiment one-against-one strategy, and cross-validation leave-one-out method (for each two different categories) was used. Table 5 reproduces a comparison of successful classification between the proposed methods and those available in the literature. As shown in Table 5 the combination of geometrical features (see Table 1) and landmarks extracted from the contour makes the proposed categorization rather successful and better than all previously proposed methods [22] and [2]. Some authors report retrieval accuracy over MPEG7 shape database, but as our method rely on learning module (SVM), it is useful for recognition and categorization not retrieval, so we can not report that accuracy here.

Table 5. Classification accuracy for different methods for the MPEG7 shape database

Method	Classification Accuracy
Chance probabilities [22]	97.1%
Normalized square distances [22]	96.9%
Racer [22]	96.8%
Polygonal representation and elastic matching [2]	97.79%
Proposed method in this paper	97.85%

7 Conclusion

In this paper an algorithm for object categorization based on shape information is proposed. In this model, landmarks from the shape contours are first extracted and then are transformed into a sequence of symbols. By using tools used for text categorization [14] and combining the information extracted from the contour with additional geometrical features a rather good categorization is achieved (see Table 5). The feature space representation makes our system completely invariant to affine transforms. The proposed method is expected to be robust for the partial occlusions, because it is based on the similarity of substrings, i.e. to local property of shapes.

References

1. K. Arbter, W.E. Snyder, H. Burkhardt, and G. Hirzinger. Application of affine-invariant fourier descriptors to recognition of 3-d objects. *IEEE PAMI*, 12(7):640–647, 1990.
2. E. Attalla and P. Siy. Robust shape similarity retrieval based on contour segmentation polygonal multiresolution and elastic matching. *Pattern Recognition*, 38(12):2229–2241, 2005.
3. S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE PAMI*, 24:509–522, 2002.
4. I. Biederman and G. Ju. Surface versus edge-based determinants of visual recognitions. *Cognit. Psych.*, 20:38–64, 1988.
5. H. Blum. A transformation for extracting new descriptors of shape. In Weiant Wathen-Dunn, editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge, 1967.
6. C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.
7. L.D.F. Costa and R.M.C. Junior. *Shape Analysis and Classification: Theory and Practice.* CRC Press, 2000.
8. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines.* Cambridge University Press, 2000.
9. S. Edelman. *Representation and Recognition in Vision.* MIT Press, 1999.
10. R.E. Fan, P.H. Chen, and Lin C.J. Working set selection using the second order information for training svm. Technical report, Department of Computer Science, National Taiwan University, 2005.

11. T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398 in Lecture Notes in Computer Science, pages 137–142. Springer Verlag, Heidelberg, DE, 1998.
12. L.J. Latecki, R. Lakämper, and U. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *IEEE Conf. on CVPR*, pages 424–429, 2000.
13. T. Lindeberg. Edge detection and ridge detection with automatic scale selection. *Int. J. of Comput. Vis.*, 30(2):77–116, 1998.
14. H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C.J.C.H. Watkins. Text classification using string kernels. *Journal of Mach. Learn. Res.*, 2:419–444, 2002.
15. P. Majer. The influence of the gamma-parameter on feature detection with automatic scale selection. In *Scale-Space '01: Proceedings of the Third International Conference on Scale-Space and Morphology in Computer Vision*, pages 245–254. Springer-Verlag, 2001.
16. F. Mokhtarian and A. Mackworth. Scale based description and recognition of planar curves and two-dimensional shapes. *IEEE PAMI*, 8(1):34–43, 1986.
17. H. Murase and S.K. Nayar. Visual learning and recognition of 3-d objects from appearance. *Int. J. of Comput. Vis.*, 14(1):5–24, 1995.
18. E. Rivlin and I. Weiss. Local invariants for recognition. *IEEE PAMI*, 17(3):226–238, 1995.
19. G. Salton, A. Wong, and C.S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.
20. T. Sebastian, P.N. Klein, and B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE PAMI*, 26(5):550–571, 2004.
21. D. Sharvit, J. Chan, H. Tek, and B.B. Kimia. Symmetry-based indexing of image databases. *J. of Visual Communication and Image Representation*, 9(4):366–380, 1998.
22. B.J. Super. Learning chance probability functions for shape retrieval or classification. In *IEEE Workshop on Learning in Computer Vision and Pattern Recognition at CVPR*, volume 6, page 93, 2004.
23. S. Ullman. *High-level Vision.* MIT Press, 1996.
24. V. Vapnik. *Statistical Learning Theory.* Wiley-Interscience, 1998.