Annales UMCS Informatica AI 3 (2005) 35-43

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Partial shape matching using convex hull and Fourier descriptors

Krzysztof Kocjan*

Institute of Mathematics, Silesian University, Bankowa 14, 40-007 Katowice, Poland

Abstract

An application of Fourier descriptors and convex hull for shape analysis is presented. Convex hull is used for dividing a shape into small parts. Amplitude spectrum which is invariant to scaling, translation and choosing a starting point is obtained from the Fourier descriptors (see e.g [1-3]) and used for comparison. All calculations are performed with the author's software and some algorithms from literature [1,4]. For convex hull estimation the *Graham* algorithm is used.

1. Introduction

Nowadays, the shape analysis for of applications is a very modern discipline. From year to year more and more scientists try to explore information which is hidden in a picture. The Fourier descriptors (**FD`s**) are often used to describe shapes and are helpful for their comparison. It is known, that some first **FD`s** can approximate the investigated object very well. Comparison experiments give good results in the case of simple transformations and small changes of the object. However, if we cut out or change some part of the shape we obtain **FD`s** which are so different from the original, that consequently the comparison is impossible. If we change some part of the shape and divide that shape into segments we can observe then only a few segments different from the original, but the others are the same. This observation and [2,5] has motivated the presented paper.

Authors in [2] for many different angles α and β, $0 \le \alpha$, $\beta \le 2\pi$ built a database consisting of **FD`s** for the interval $[\alpha, \beta]$. Changing parameters from α to β describe a part of the shape. However, in [5] for partitioning a polynomial approximation is used. We have chosen convex hull, because of more stability with extracting segments than corners or polygonal approximation. Having points creating the convex hull we can choose only points that directly touch the boundary of the shape and with the help of them it is possible to divide the shape

* *E-mail addresss*: kocjank@ux2.math.us.edu.pl

into segments. The algorithm for dividing and comparing objects is discussed further in this article.

Until now there has not been a universal method to compare objects. Some solutions are better for one kind of objects, but fail with the others. A great variety of different methods is used for recognition of shapes. As follows from our experiments the local analysis is better than the global one. If we look for an example of a shape of a dog, we perceive legs, which are some features of the dog. If we look at different pictures of the dog, we can see the legs too. This indicates similarity of two dogs. Of course, the more shared features are, the better similarity is. Hence, we take into account locality to perceive the same features. In this article the presented approach is a trial of perceiving shared features and state about similarity of two objects.

In [2] there is used a method, whose main disadvantage is a large database for objects for different angles α and β. In our work we have only a few segments needed for comparison. A better approach is presented in [5], but choosing a parameter responsible for approximation is very difficult. Moreover we use a two-step comparison. The standard error between the segments is computed to find out if they are the same or not, but the essential comparison is made further. The proposed similarity measure depends among others on the number of segments in a database or the number of fitting segments.

Of course the presented approach is not universal and may be used for some objects, but for others it may fail. Some problems are discussed and some improvements are proposed.

2. Fourier descriptors

Suppose, we have a periodic piecewise continuous and differentiable function $f(x)$ defined on $[0,2\pi]$. We can approximate such a function by the Fourier series in the following way

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_k \cos kx + b_k \sin kx \right), \tag{1}$$

where the Fourier coefficients are

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos kx dx \quad \text{and} \quad \mathrm{b}_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin kx dx \tag{2}$$

We can also compute a sequence $\{A_k\}$ called the *amplitude spectrum* by the formula

$$A_k = \sqrt{a_k^2 + b_k^2} \tag{3}$$

In practice, we apply the discrete version of the Fourier transform, namely considering the value of the function $f(x)$ at N points $x_r = 2\pi r / N$. Then we can write

$$f(x_r) = a_0 + \sum_{n=1}^{[(N+1)/2]-1} \left(a_n \cos x_r n + b_n \sin x_r n\right) \tag{4}$$

$$\begin{aligned} a_0 &= \frac{1}{N}\sum_{r=0}^{N-1} f(x_r), \\ a_n &= \frac{2}{N}\sum_{r=0}^{N-1} f(x_r)\cos x_r n, \\ b_n &= \frac{2}{N}\sum_{r=0}^{N-1} f(x_r)\sin x_r n, \end{aligned} \tag{5}$$

where r = 0,1..,N-1 and n = 0,1,…[(N+1)/2] – 1.

To distinguish between shapes we use the numbers $\{A_k\}$ or after normalization the sequence $\{B_k\}=\{A_k/A_0\}$. After normalization $\{B_0\}$ is always *1*. In this work we use several first numbers $\{B_k\}$ to describe a contour of the shape.

3. Shape segmentation

All investigated objects are black and white bitmap after binarization. The first step to partitioning is to find contour of the object. With the help of the algorithm from [1], we can do this very well. If we have all contour points, we next search for convex hull by the Graham algorithm. Some examples of this operation are presented below.

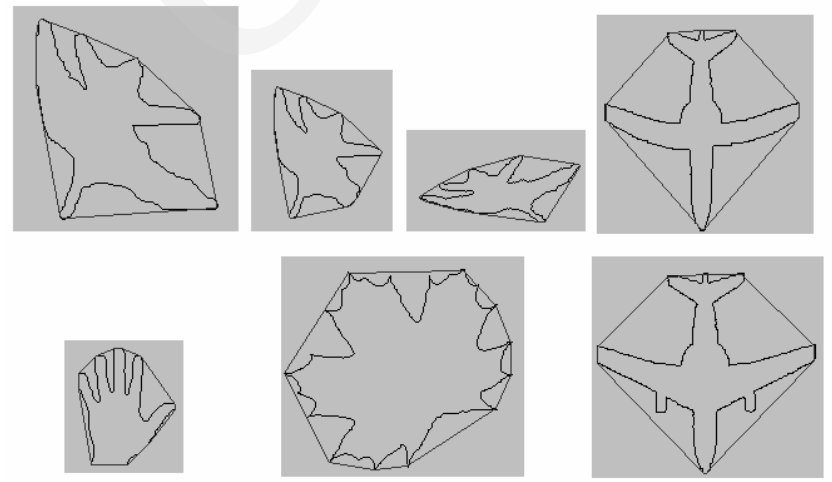

Fig. 1. Sample objects with convex hull

Additionally, we set the starting point on the contour and on the convex hull at the same point, which satisfies the following condition. We choose two directly following points *A, B* from the convex hull, which also touches the

shape. Next, we check the point lying on the boundary between *A* and *B* so that the distance *D* between the shape and the line *l* is larger than the previously fixed small threshold value D_{min}. If such a point exists, we choose *A* for the starting point, if not, then the object is convex or nearly convex. It depends on the threshold value D_{min}. This is shown in Fig. 2.

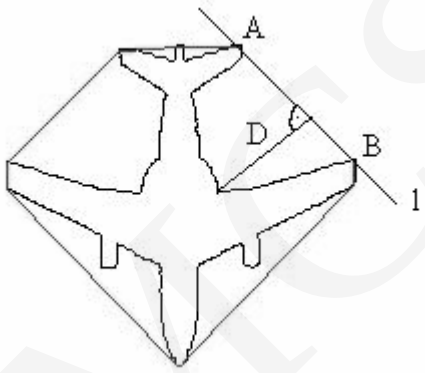

Fig. 2. Choosing the starting point

Suppose *l: ax+by+c=0*, then formally the condition is

$$D = \max\left\{ D_i : D_i = \frac{ax_i + by_i + c}{\sqrt{a^2 + b^2}},\ i = 1,...,n,\ A = (x_1, y_1),\ B = (x_n, y_n) \right\} > D_{\min}. \quad (6)$$

Starting from any point satisfying the above condition we build an array of boundary points from that point to the next. In the next array we hold points from the second to the third ones with the same condition. We stop when we build the array of points from the last to the first. Formally, suppose we have an object *X* and points $P_i = (x_i, y_i)$, $i = 0,1, \ldots, N-1$ lying on the contour. *N* denotes the number of points. Also, let the founded points from the convex hull touch the object C_i, $i=0,1, \ldots, k-1$, where *k* is the number of convex hull points asuming that these points are after setting the starting point. Of course we have $P_0 = C_0$. Let the first segment be S_0. This segment contains the following points

$$S_0 = C_0, P_1, P_2, \ldots, P_{i-1}, P_i = C_1.$$

If C_1 and C_2 satisfy condition *(6)* then

$$S_1 = C_1, P_{i+1}, P_{i+2}, \ldots, P_{j-1}, P_j = C_2,$$

else we store the points C_1, P_{i+1}, P_{i+2}, …, P_{j-1} in a new array $\overline{S}$. Up to some points C_k and C_{k+1} the condition *(6)* is not true, we add points from the boundary to the array $\overline{S}$. If the condition becomes true, we count the number of points in the array $\overline{S}$ and if this number is greater than the fixed value N_{min}, then $\overline{S}$ is the next segment. In another case, we reject points from $\overline{S}$. The results of segmentation on several tested shapes can be seen below.

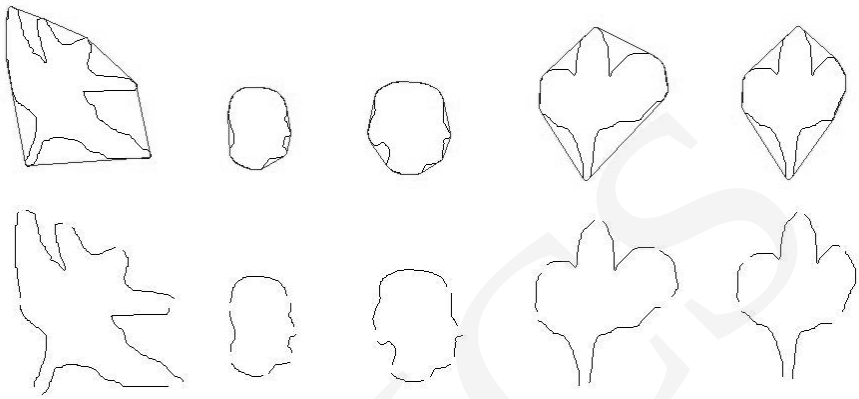

Fig. 3. a) contour b) segments obtained with the help of convex hull

For the segment approximation we use the Fourier descriptors, but at first for each segment we build the distance function to calculate **FD's**. Of course, we may use any other function, but the distance function is very simple and gives good results. To build the distance function we use the following procedure.

For the first $P=(P^x,P^y)$ and the last $Q=(Q^x,Q^y)$ point in a segment we calculate a center $S=(S^x, S^y)$ from the known formula

$$S_x = \frac{P^x + Q^x}{2} \qquad S_y = \frac{P^y + Q^y}{2} \tag{7}$$

and all the distances from S to each point from the segment.

$$f(i) = \sqrt{\left(S^x - P_i^x\right)^2 + \left(S^y - P_i^y\right)^2},\ i = 0,1,...,p-1, \tag{8}$$

where p denotes the number of points in the segment, P_i^x, P_i^y are respectively the x-coordinate and the y-coordinate in the contour point P_i.

In the case, when only one segment exists we set S to the center of gravity of the shape.

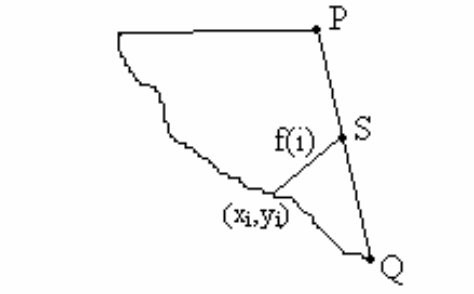

Fig. 4. The distance function for one sample segment

The next step is to calculate for each function the normalized **FD's** (normalized amplitude spectrum). Consequently, we obtain a matrix of values which is used for comparison

$$M = \left[m_{ij} \right]_{\substack{0 \le i \le k-1 \\ 0 \le j \le n}},$$

where *j*-th column contains i-th **FD's** for *j*-th segment. To build the database of objects we must calculate the matrices and store them.

4. Comparison algorithm

To compare a new object with those in the database we must calculate the matrix for that new object first. The algorithm proposed here for comparison of two objects contains two steps. We start to find the number of best fitted segments Seg_p using some threshold value *T*. Suppose, we have two matrices

$$M = \left[m_{ij} \right]_{\substack{0 \le i \le n \\ 0 \le j \le k_1}} \qquad \overline{M} = \bar{M} = \left[\bar{m}_{ij} \right]_{\substack{0 \le i \le n \\ 0 \le j \le k_2}} .$$

Matrix M is for the new object, however $\bar{M}$ is for the object from the database. Now we can calculate Seg_p as follows

$$Seg_p = \left| \left\{ A : A = \sqrt{\sum_{i=1}^{n} \left(m_{ij} - \bar{m}_{il} \right)^2},\ A < T,\ j = 0,1,...,k_1 - 1,\ l = 0,1,...k_2 - 1 \right\} \right|. \quad (9)$$

Let Seg_d and Seg_c denote the number of segments in the database and the one of segments for the new object. To distinguish between two objects *X, Y* we propose the following alternative two formulae

$$Sim1(X,Y) = \frac{1}{2}\left(\frac{Seg_p}{Seg_d} + \frac{Seg_p}{Seg_c} \right) \frac{\min\{Seg_d, Seg_c\}}{\max\{Seg_d, Seg_c\}} \quad (10)$$

The *Sim2* formula depends on error between segments.

$$Sim2(X,Y) = \begin{cases} \frac{1}{2}\left(\frac{Seg_p}{Seg_d} + \frac{Seg_p}{Seg_c} \right) \frac{\min\{Seg_d, Seg_c\}}{\max\{Seg_d, Seg_c\}} \sqrt{1-E} & for \ \ 0 \le E \le 1 \\ 0 & for \ \ E > 1 \end{cases} \quad (11)$$

where

$$E = \sum_{k=1}^{Seg_p} A_k$$

for A_k, *j*, *l* satisfying the conditions from formula (9). Of course for completely different shapes, that is $Seg_p=0$ and $E>0$, we have $Sim1(X,Y)=Sim2(X,Y)=0$ and for the same shape $Seg_p = Seg_d = Seg_c$ and $E=0$, we have

$$Sim1(X,Y) = \frac{1}{2}(1+1) \cdot 1 = 1,$$

$$Sim2(X,Y)=\frac{1}{2}(1+1)\cdot 1\cdot\sqrt{1-0}=1.$$

Additionally, if $Seg_p < Seg_d$ or $Seg_p<Seg_c$ and $E>1$ (very large error), then *Sim2* is set to *0*. For the other object $0 < Sim1(X,Y)<1$ and $0< Sim2(X,Y)<1$.

For comparison, we can use standard similarity measures that are often used in literature [6], but in this method the above measure is good and reasonable. This is because it expresses the number of shared (the same or different) features.

5. Experiments

Experiments with many objects have been performed and some results for the objects are presented in Fig. 5. For approximation twenty Fourier descriptors have been used. The threshold T has been set to *0.08* and the numbers $D_{min}=10+0.005\cdot N$, $E_{min}=4+0.005\cdot N$, where N denotes the number of contour points.

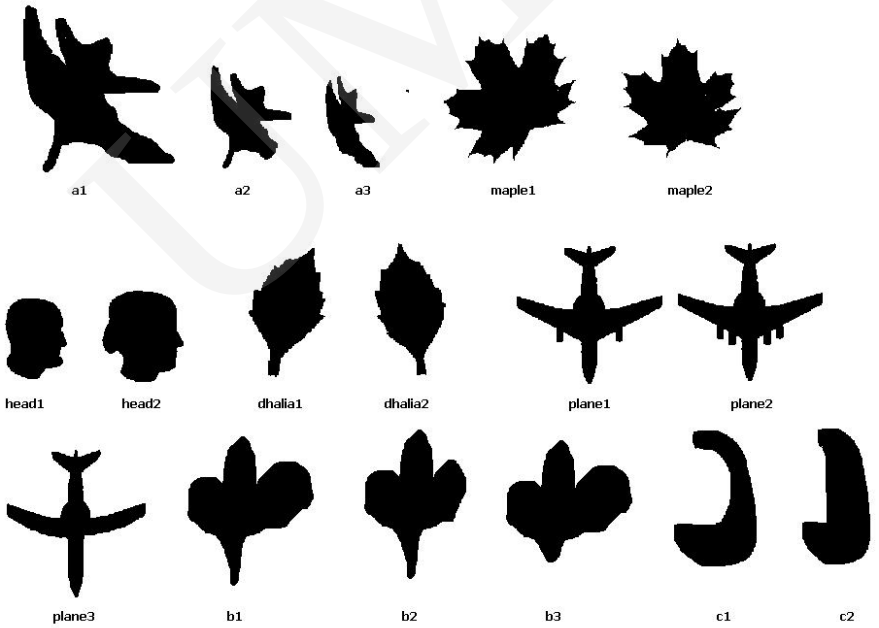

Fig. 5. Objects chosen for the analysis

We can see that the presented objects are subjected to different distortions. For example, the object **a2** is the same as **a1**, but some part from **a1** has been cut. The objects **maple1** and **maple2** are leaves entered to computer by a scanner. In Tab.1 we gathered the results for all objects from Fig. 5 using *Sim1*, but in Tab.2 using *Sim2*.

Table 1. Results from the analysis using *Sim1*

	a1	a2	a3	maple1	maple2	head1	head2	dahlia1	dahlia2	plane1	plane2	plane3	b1	b2	b3	c1	c2
a1	**1.000**	**0.714**	**0.714**	0.372	0.372	0.133	0.133	0.340	0.000	0.351	0.351	0.469	0.265	0.398	0.398	0.092	0.092
a2		**1.000**	**0.571**	0.223	0.297	0.133	0.000	0.170	0.100	0.234	0.234	0.351	0.133	0.265	0.265	0.184	0.091
a3			**1.000**	0.298	0.298	0.265	0.265	0.255	0.296	0.351	0.351	0.351	0.265	0.398	0.398	0.091	0.184
maple1				**1.000**	**0.454**	0.140	0.140	0.260	0.083	0.157	0.157	0.157	0.140	0.210	0.140	0.054	0.054
maple2					**1.000**	0.140	0.211	0.347	0.413	0.314	0.235	0.157	0.281	0.211	0.351	0.054	0.107
head1						**1.000**	**0.667**	0.240	0.278	0.219	0.219	0.109	0.167	0.167	0.333	0.111	0.111
head2							**1.000**	0.320	0.370	0.219	0.219	0.109	0.167	0.333	0.333	0.000	0.111
dahlia1								**1.000**	**0.665**	0.090	0.090	0.090	0.160	0.160	0.240	0.000	0.060
dahlia2									**1.000**	0.105	0.105	0.105	0.092	0.185	0.185	0.000	0.068
plane1										**1.000**	**0.750**	**0.625**	0.109	0.109	0.219	0.156	0.000
plane2											**1.000**	**0.625**	0.109	0.109	0.219	0.156	0.000
plane3												**1.000**	0.109	0.109	0.219	0.156	0.000
b1													**1.000**	**0.667**	**0.667**	0.000	0.111
b2														1.000	0.500	0.111	0.222
b3															**1.000**	0.111	0.000
c1																**1.000**	**0.500**
c2																	**1.000**

Table 2. Results from the analysis using *Sim2*

	a1	a2	a3	maple1	maple2	head1	head2	dahlia1	dahlia2	plane1	plane2	plane3	b1	b2	b3	c1	c2
a1	**1.000**	**0.670**	**0.431**	0.316	0.319	0.118	0.130	0.401	0.000	0.322	0.322	0.414	0.250	0.365	0.363	0.089	0.089
a2		**1.000**	**0.435**	0.203	0.260	0.120	0.000	0.221	0.113	0.217	0.217	0.323	0.128	0.251	0.252	0.174	0.090
a3			**1.000**	0.184	0.183	0.000	0.000	0.100	0.098	0.099	0.099	0.190	0.149	0.286	0.148	0.137	0.136
maple1				**1.000**	0.405	0.064	0.131	0.214	0.076	0.147	0.147	0.147	0.132	0.193	0.133	0.052	0.053
maple2					**1.000**	0.124	0.193	0.216	0.326	0.272	0.211	0.147	0.245	0.192	0.290	0.052	0.103
head1						**1.000**	**0.534**	0.197	0.284	0.188	0.188	0.189	0.147	0.147	0.287	0.137	0.135
head2							**1.000**	0.392	0.392	0.207	0.207	0.108	0.164	0.314	0.316	0.000	0.108
dahlia1								**1.000**	**0.740**	0.122	0.122	0.122	0.208	0.209	0.305	0.000	0.076
dahlia2									**1.000**	0.121	0.121	0.121	0.107	0.205	0.209	0.000	0.000
plane1										**1.000**	**0.750**	**0.593**	0.107	0.107	0.202	0.145	0.000
plane2											**1.000**	**0.596**	0.106	0.106	0.202	0.144	0.000
plane3												**1.000**	0.106	0.106	0.204	0.145	0.000
b1													**1.000**	**0.641**	**0.628**	0.000	0.106
b2														**1.000**	**0.471**	0.108	0.212
b3															**1.000**	0.107	0.000
c1																**1.000**	**0.491**
c2																	**1.000**

In both tables it is clearly seen that for the same objects, after some distortions the algorithm has been well matched, but in the investigations there were objects for which the presented approach fails. For similar objects the measure is larger than 0.6, but for dissimilar ones the measure is less than 0.35. The values between 0.35 and 0.6 suggest average similarity. We can use affined invariant **FD's** to improve this algorithm. See [7].

6. Conclusions

It is easy the notice, that the disadvantage of locality is also the fact that having some shape we can move some parts. Consequently, we obtain a completely different shape from the original, but parts are the same and the algorithm indicates similarity. Taking this into account it is necessary to think about modifying the similarity measure to obtain the number depending on the globality of a shape. It is seen that we cannot confine ourselves only to locality or only to globality, because we can always find objects that are completely different, but the algorithm recognizes them to be the same.

References

[1] Kindratenko V., *Development and Application of Image Analysis Techniques for Identification and Classification of Microscopic Particles*, Universiteit Antwerpen, Ph.D. Thesis, Antwerpen, (1997), (http://cgi.ncsa.uiuc.edu/People/kindr/phd/index.html).

[2] Alkhodre A., Belarbi M., Langs G., Tosovic S., Sicard N., *Shape Description via Fourier Analysis*, ERASMUS Intensive Program 2001 Pavia, May 7-18 (2001).

[3] Kocjan K., *Analysis of 2D shapes with the help of Fourier descriptors*, Systemy wspomagania decyzji, December (2003).

[4] Jankowski M., *Elements of computer graphics*, Wydawnictwa Naukowo-Techniczne, Warszawa, (1990) 65, in Polish.

[5] Gorman J.W., Mitchell O.R., Kuhl F.P., *Partial Shape Recognition Using Dynamic Programming*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 10(2) (1988) 257.

[6] Voltkamp R., *Shape matching: Similarity measures and algorithms*, International conference on Shape Modeling and Application (SMI'01).

[7] Arbter K., Snyder E., Burkhard H., Hirzinger G., *Application of affine-invariant Fourier Descriptors to recognition of 3D objects*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(7) (1990) 640.