Efficient Partial Shape Matching Using Smith-Waterman Algorithm

Longbin Chen
Computer Sciences
UC, Santa Barbara
lbchen@cs.ucsb.edu

Rogerio Feris
Exploratory Computer Vision
IBM T.J. Watson Research
rsferis@us.ibm.com

Matthew Turk
Computer Sciences
UC, Santa Barbara
mturk@cs.ucsb.edu

Abstract

This paper presents an efficient partial shape matching method based on the Smith-Waterman algorithm. For two contours of m and n points respectively, the complexity of our method to find similar parts is only O(mn). In addition to this improvement in efficiency, we also obtain comparable accurate matching with fewer shape descriptors. Also, in contrast to arbitrary distance functions that are used by previous methods, we use a probabilistic similarity measurement, p-value, to evaluate the similarity of two shapes. Our experiments on several public shape databases indicate that our method outperforms state-of-the-art global and partial shape matching algorithms in various scenarios.

1. Introduction

Shape matching is to find point correspondences between the points of two shapes. It has applications in many different areas, including object recognition, stereo matching, and content based image retrieval. The shape of an object can be represented either by a group of independent points, by its contour, or by its skeleton. In this paper, we use an object's contour for shape matching, which can usually be defined by a circular sequence of feature points. Each feature point is described by its coordinates and shape descriptors. The problem of contour-based shape matching is

Sequence matching algorithms have been successfully applied for contour matching [2]. In this approach, both the *matching shape* and the *template shape* are represented as a sequence of feature points. Most previous sequential matching algorithms use *global matching* algorithms to find the distance between two shapes [2]. In these, a certain distance function, for instance, the edit-distance, is defined between two *whole* sequences and an optimization algorithm (e.g., dynamic programming), is designed to find the distance. Global matching methods work well when the shapes are globally similar and there is no large occlusion, articulation and other shape variance.

However, when one shape or both shapes have occlusions or large local shape deformations, i.e., when two shapes are only similar in parts, global shape matching algorithms are not likely to work well. In Figure 1, for instance, if a global matching algorithm is used, the occluded part of the hand on the right would be included in the distance measure between two shapes, resulting in a large bias.

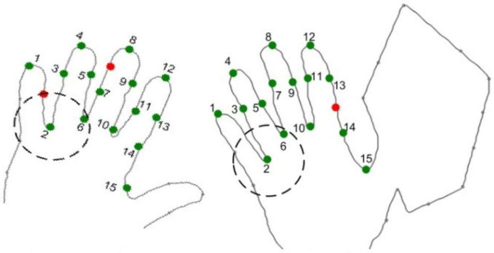


Figure 1. This figure illustrates the paper's main idea. Similar shape parts (point 1 through point 15) are efficiently found by using a local alignment algorithm and local shape descriptors without exhaustive search; green points are matched point pairs (labeled by number) while red points are unmatched points; other black points are not in similar segments. The original images are from Kimia's dataset [4].

This paper presents an efficient method to find the similar parts of two shapes and measure the similarity of the shapes using only similar parts. Our main contribution is to propose a local shape matching method based on the Smith-Waterman algorithm [1] to efficiently find similar parts of two shapes without requiring exhaustive search for all possible parts. For two contours of m and n points respectively, the complexity of our method to find similar parts is only *O(mn)*, compared with the exhaustive search of $O(2^{m+n})$. In addition, a large shape database can be easily indexed for fast retrieval [7][8] using our shape representation and similarity metrics.

Our experimental results show that our approach outperforms existing global methods in several public databases even when there is no occlusion, while our method is insensitive to global shape transformations such as translation, rotation, articulation, and scaling.

2. Shape Representation

2.1 Shape descriptors

Generally there are two kinds of shape descriptors: global and local. A global shape descriptor at a feature point takes into account the position of all other points relative to this point, for instance, shape context [3][3], inner-distance shape context[2], and curvature scale space [18], and discrete curve evolution [19] . In contrast to global shape descriptors, a local shape descriptor of a feature point depends only on its neighboring points and will not change as long as its neighboring feature points do not change. Local shape descriptors are robust to occlusions and usually insensitive to global shape transformations, such as rotation, articulation, or view changes. However, they lack the information about the whole shape and are generally more sensitive to local noises. Examples of local shape descriptors are tangent vector [12], primitives descriptors [14], the turning angle [16], and more can be found in [13] [17].

2.2. Turning Angle and DAS

In this paper, we use Turning Angle and Distance across the Shape feature as our shape descriptor. For each feature point along the object contour (see Section 2.3 for how these points are selected), shape descriptors are extracted. All feature points are initially sorted in counter-clockwise order. For a feature point *p*, its immediate predecessor *q*, and its immediate successor *r*, *qpr* forms a *turning angle* (or TA). The interior bisector of angle *qpr* intersects the contour at point *p'*. The length of *pp'* is the *distance across the shape* (or DAS) at point *p*. If the bisector intersects with the shape multiple times, the distance to the closest intersection is used.

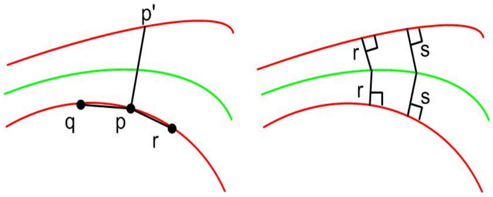


Figure 2. DAS shares some intuitive with the shock graph feature

(a) Turning Angle $\hat{qpr}$ and DAS $|pp'|$ (b) shock graph [4]

The DAS descriptor shares some intuition with the shock graph [4] and the distance transforms [6], but it is more straightforward and efficient to compute. The TA descriptor is translation, rotation, and scale-invariant. DAS is also translation and rotation-invariant. After DAS is normalized with respect to the shape's size, DAS is scale-invariant as well. TA and DAS are both sensitive to key point location and local contour noises. By applying our two heuristic rules for selecting feature points (see Section 2.3), we can reduce the variance of DAS and make matching practical. DAS is not completely local because sometimes it can be affected by contour points far away. We use a linear combination of TA and DAS similarities, while setting small weights for DAS when the distance across the shape is large. This ensures that that our similarity function is approximately local.

2.3. Feature point selection

Given an image, we use contour detection algorithms [2][3] to find the objects' contours. Our algorithm then finds the object's contour and represents it by the pixels on the contour, in a clockwise order. The number of edge pixels in a single image usually ranges from several hundreds to thousands. Some previous shape matching algorithms (e.g., [2][3]) use a smaller number of points (100 or 200) that are uniformly sampled on all contour pixels. Other matching methods use polygon approximation methods to approximate the contour with polygons [8][9]. Rather than uniform sampling, our method selects points according to two heuristics. The first is that high curvature points are more likely to be selected. The second is that selected key points should be kept from other selected points as far as possible. The distance between two points is defined as the number of pixels on the contour between these two points. By applying these two heuristics, we somehow address the issue of noises and robustness.

3. Efficient Partial Shape Matching

3.1 Global and local alignment

Picking points a_1 and b_1 from two contours respectively as the starting points, two shape contours are represented as:

$$A = [a_1,a_2, \ldots,a_m]$$
$$B = [b_1,b_2, \ldots,b_n]$$

where a_i and b_j are feature points, and *m, n* are the numbers of feature points. These feature points are described with local feature descriptors.

One global distance that is frequently used is the *edit distance*. The *edit distance* between two sequences *A* and *B* is defined as the minimal number of standard operations to transform sequence *A* to sequence *B*. Standard operations might be different with respect to applications. Three kinds of standard operation, however, are frequently used to compare two strings: *substitution*(a_i, b_j), *insert*(a_i) and *delete*(b_j). The edit-distance can be calculated using Dynamic Programming and has been applied for shape matching before [9][10][11][20]. Several algorithms use domain-specific operations as standard operations in calculating edit-distance, for example, deleting a node in a Shock Graph [4].

The *Smith-Waterman* algorithm (or SW algorithm) was proposed by Temple Smith and Michael Waterman for identification of common molecular subsequences [1]. It is a local alignment algorithm that matches two sequences using dynamic programming. SW algorithm differs from other matching algorithms in that it finds only similar subsequences without exhaustive search. It has been recently applied for some computer vision problems, such as object classification [23] and activity recognition [24].

3.2. Efficient partial shape matching

A *similarity function $f(a_i, b_j)$* is defined on two feature points, a_i and b_j. $f(a_i, b_j)$ is positive if a_i and b_j are similar and negative if a_i and b_j are dissimilar. Also, a *gap function $d(a_i)<0$, $d(b_j)<0$* is defined as the cost for deleting or inserting a feature point.

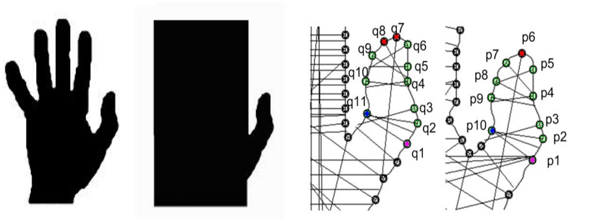
In order to find the similar parts of two contours, we use a matrix H to save all the matching costs. Preliminary values of H have the interpretation that $H_{c,r}$ is the maximum similarity score of any two segments ending in a_c and b_r, respectively. These values are obtained using dynamic programming [1]:

$$H_{c,r} = \max\begin{cases} H_{c-1,r-1} + f(a_c, b_r), & f(a_c, b_c) \in R \\ H_{c-1,r} + d(a_c), & d(a_c) < 0 \\ H_{c,r-1} + d(b_r), & d(b_r) < 0 \\ 0 \end{cases} \tag{1}$$

The last line in (1) is important because in this way we can make the only similar part visible in matrix H and dissimilar parts are set to zeros. The pair of segments with maximum similarity is found by first locating the maximum element of H. The other matrix elements leading to this maximum value are then sequentially determined with a trace-back procedure ending with an element of H equal to zero. The pair of segments with the next best similarity is found by applying the trace-back procedure to the second largest element of H not associated with the first trace-back.

Figure 3 shows a small part of the matrix H and the most similar pair of segments, which is also called a high score segment pair (see Section 3.3). In Figure 3.b, points q_1-q_{11} match points p_1-p_{10}. In Figure 3.c, the circled entries are the corresponding similarity measurement of matched points. The correspondence of entries and points are labeled by colors. Back tracking starts from the maximal entry (blue circled number, 0.312), ends when the entry is zero (the last non-zero entry is the circled purple number, 0.059). In Figure 3.b, points q_8, q_7 and p_6 are *insert* or *deleted* and don't have matching points. All other points are matched. In Figure 3.c, entries corresponding to dissimilar point pairs are forced to be zero to make only similar parts visible. The similar part we find is:

$$\left|\begin{array}{c|c|c|c|c|c|c|c|c|c|c|c} q_1 & q_2 & q_3 & q_4 & q_5 & - & q_6 & q_7 & q_8 & q_9 & q_{10} & q_{11} \\ p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & - & - & p_8 & p_9 & p_{10} \end{array}\right|$$



(a) (b)

		p1	p2	p3	p4	p5	p6	p7	p8	p9	p10		
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.006	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.000	0.040	0.057	0.027	0.000	0.000	0.000	0.000	0.000	0.000	0.030	0.029	0.000
q1	0.000	0.059	0.072	0.048	0.018	0.000	0.000	0.000	0.000	0.008	0.056	0.084	0.054
q2	0.000	0.030	0.118	0.091	0.061	0.025	0.000	0.000	0.000	0.000	0.035	0.081	0.051
q3	0.000	0.000	0.088	0.166	0.136	0.100	0.058	0.010	0.000	0.000	0.005	0.051	0.015
q4	0.000	0.000	0.052	0.136	0.226	0.196	0.160	0.118	0.070	0.034	0.004	0.015	0.000
q5	0.000	0.000	0.010	0.100	0.196	0.269	0.239	0.217	0.187	0.151	0.109	0.061	0.007
q6	0.000	0.013	0.000	0.058	0.160	0.239	0.203	0.222	0.236	0.243	0.213	0.177	0.135
q7	0.000	0.000	0.000	0.010	0.118	0.203	0.177	0.192	0.206	0.213	0.177	0.135	0.208
q8	0.000	0.028	0.000	0.000	0.070	0.161	0.147	0.156	0.196	0.247	0.244	0.214	0.178
q9	0.000	0.000	0.000	0.000	0.038	0.124	0.111	0.181	0.206	0.217	0.214	0.212	0.182
q10	0.000	0.000	0.000	0.000	0.046	0.094	0.069	0.151	0.216	0.254	0.224	0.217	0.187
q11	0.000	0.054	0.026	0.000	0.016	0.058	0.021	0.115	0.186	0.230	0.312	0.284	0.254
	0.000	0.024	0.000	0.000	0.000	0.016	0.100	0.073	0.150	0.200	0.282	0.254	0.218
	0.000	0.000	0.000	0.000	0.000	0.000	0.070	0.034	0.108	0.164	0.246	0.218	0.176
	0.000	0.000	0.000	0.000	0.000	0.000	0.034	0.000	0.060	0.122	0.204	0.176	0.128
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.006	0.074	0.156	0.128	0.074
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.020	0.102	0.074	0.014

(c)

Figure 3. An example of SW algorithm and High Scoring Segment Pair (a) shows a complete hand shape and an occluded hand shape;(b) shows that only similar parts are matched; (c) shows a small part of matrix H related to parts in (b).

3.3. High scoring segment pairs

A high-scoring segment pair, or HSP, is a pair of segments whose score cannot be improved by trimming or extension, given a similarity function and a gap function [7]. In Figure 1, green points and red points form the most similar segment pair, which is also a high score segment pair.

When matching two similar shapes, usually we can find one or more HSPs. When two shapes are dissimilar, we can still find HSPs from the similarity matrix, because feature points or segments might be similar by chance.

The *e*-value and *p*-values are two measurements usually used to evaluate the statistic significance of a HSP score when no gap is allowed in matching.

The *e*-value of a score S is the expected number of HSP with values at least S when matching two sequences without gap. Intuitively, when the numbers of features points of two shape increases, the number of HSPs will increase too. If the score of each HSP is random, the max score of HSPs will increase. The maximal value of a large number of *i.i.d.* random variables tends to be an extreme value distribution [22].

Therefore, *e*-value can be calculated by [7]:

$$E = K \cdot m \cdot n \cdot e^{-\lambda S} \tag{2}$$

where K and λ are two constants, and m, n are the length of two sequences. Taking the logarithm of both sides of Eq.2, we have

$$\ln E = \ln(K \cdot m \cdot n) - \lambda S \qquad (3)$$

So $\ln E$ is linear with respect to S.

In practice, we can use *p*-value to estimate the significance of a HSP. *p*-value is the probability of a HSP that results from random sequences. The number of HSP with score $S > S_t$ is given by the Poisson distribution [22], that is, the probability of finding *k* HSPs with score $S > S_t$ is given by :

$$p(k) = e^{-E_t} \cdot \frac{E_t^{\,k}}{k!}$$

where $E_t = K \cdot m \cdot n \cdot e^{-\lambda St}$ is given by Eq. (2)

Hence, the probability of finding exactly zero HSP with score $S > S_t$ is given by

$$p(k)\,|_{k=0} = e^{-E_t} \cdot \frac{E_t^{\,k}}{k!}\,|_{k=0} = e^{-E_t}$$

and so the probability of finding at least one HSP with S > St is

$$p(k > 0) \,|= 1 - p(k)\,|_{k=0} = 1 - e^{-E_t}$$

We can set a certain confidence level, such as 5% or 10% to judge if the HSP is a significant search or just two segments that happens to match. When *E* is small, *e*-value is close to *p*-value.

The above theory about e-value and p-value are valid only for matching without gaps. However, many experiments indicate that the theory applies to matching with gaps as well [7].

Because getting *K* and λ requires collecting random samples, and we have to select a confidence level to judge if HSP is significant or not, and in our experiment, *m* and *n* are similar, we can just use *e*-value of a HSP to compare it with a pre-defined threshold *t*.

3.4. Dealing with a circular sequence

The algorithm mentioned above is in general for matching two sequences. A contour, when represented by feature points, is actually a circular sequence. When we represent the contour using a sequence of feature points, a certain feature point has to be selected as the starting point. In this case, the enclosed contour is broken into an open curve. To address this issue, we extend one of the two matching shape by duplicating part of the sequence. After retrieving HSPs in Section 3.3, duplicate entries corresponding to same feature points of H matrix will be set to zero at the same time. Please note that the method of duplicating part of the sequence won't work well when the number of the extended shape's feature points is much smaller than the other shape because the extended part might match part of the second shape and add similarity score incorrectly. In our experiment, we use similar number of feature points to represent the shapes.

3.5. Choosing the similarity function and gap function

We use a distance measure for two feature points as [2, 3].

$$c_{DAS}'(a_i, b_j) = \frac{|\,DAS(a_i) - DAS(b_j)\,|}{DAS(a_i) + DAS(b_j) + \varepsilon}$$

$$c_{TA}'(a_i, b_j) = \frac{|\,TA(a_i) - TA(b_j)\,|}{|TA(a_i)| + |TA(b_j)| + \varepsilon}$$

where $DAS(\cdot)$ and $TA(\cdot)$ are the DAS and TA features of the feature point respectively, and ε is a small constant to make the denominators non-zero. Two similar functions are defined as

$$f_{DAS}(a_i, b_j) = c_{DAS} - c_{DAS}'(a_i, b_j)$$

$$f_{TA}(a_i, b_j) = c_{TA} - c_{TA}'(a_i, b_j)$$

where c_{DAS} and c_{TA} are two positive constants. DAS feature is not completely local (See section 2.2). To generate a local similarity function, we weight these two functions by w_1 and w_2, i.e.,

$$f(a_i, b_j) = \frac{w_1}{w_1 + w_2} \cdot f_{DAS}(a_i, b_j) + \frac{w_2}{w_1 + w_2} \cdot f_{TA}(a_i, b_j)$$

In this paper, we use

$$w_1 = 0.1 + |\,TA(a_i)\,| + |\,TA(b_j)\,|$$

and $w_2 = \min\left(1, \dfrac{1}{\min(DAS(a_i), DAS(b_j))}\right)$

The matching result largely depends on the similarity functions. In order to make the similarity function meaningful, the expected score for two random feature points should be negative [1].

We can define the gap cost to be a const value *d*.

$$d'(a_i) = d$$

In practice, in order to limit consecutive delete/insert operations, we give a penalty to consecutive delete/insert operations as:

$$d(a_i) = d \cdot (1 + a)^L$$

Where *c*, *α* is a constant and *L* is the number of consecutive delete/insert operations. In computing the ***H*** matrix, we use a supplementary matrix ***P*** to save the operations and another matrix ***S*** to save the number of consecutive delete/insert operations for each cell.

4. Experiments

4.1 Computational efficiency

First we test our algorithm's efficiency, and compare it with previous methods (Table 1). In [2][3], around 200 feature points are selected and used for matching. By using our feature point selection method, we can achieve slightly better accuracy with much fewer feature points.

For rotation-invariant shape matching, previous methods have to use all possible points as a starting point. An approximation was used in [2] to reduce the computational cost where only one of every n_s points is used as the starting point. Because n_s is a constant, the complexity of [2]'s method is still $O(m^2n)$ where m, n are the number of feature points of two contours respectively. To compare the computational cost of previous methods with our method, we run both algorithms for 100 times, and our algorithm is 10 to 20 faster than previous methods, depending on the value of n_s.

Method	# of Point	# of feat.	Complexity
SC+DP[3]	200	60	$O(m^2n)$
IDSC+DP[2]	200	60	$O(m^2n)$
DAS + DP	around 170	1	$O(m^2n)$
DAS3 + DP	around 170	1	$O(m^2n)$
DAS + SW	around 170	3	$O(mn)$
DAS-TA + SW	around 170	3	$O(mn)$

Table 1. Comparison of computational efficiency

4.2 Articulation shape database

We thank the authors of [2] for making their articulation dataset available to us and we list the comparison in Table 2.. To make a fair comparison, we use the same evaluation methodology as in [2]. We fixed the bottom left point as the starting point. Each shape is used as query shape and the retrieval result is sorted by their similarity and summarized as the number of 1st, 2nd, and 3rd similar shape of the same category. From the comparison, it is clear that our method outperforms both shape context and inner-distance features. And the error rate is only 1/3 of inner-distance from [2].

Distance Type	Top1 of 40	Top2 of 40	Top3 of 40	Top4 of 40	Total Missed
SC+DP[2][3]	20	30	29	35	114
IDSC+DP[2]	0	6	5	13	24
DAS + DP	0	0	1	7	**8**
DAS + LM	0	0	1	7	**8**

Table2. Retrieval error on articulation shape database [2], DP:Global Matching, LM:Local Matching

Our explanation of our method's advantage is that inner-distance feature, like shape context features, is a global feature that has 60 feature values. Caetano [21] pointed out that, out of all these 60 features, there are some features more discriminative than other features. If the similarity function (or distance function) treats all 60 features equally, as in [2][3], the final similarity function might not be as discriminative as it could possibly be obtained.

4.3 Kimia database

We also test our algorithm on Kimia's datasets (Table 3), which have been frequently used for evaluating shape matching algorithms [2][3][4][10][15].

Method	Top1 of 25	Top2 of 25	Top3 of 25
Sharvit et. al[15]	23	21	20
Gdalyahu and Weinshall[9]	25	21	19
Belongie et. al[3]	25	24	25
Haibing[2]	25	24	25
DAS + DP	25	25	24
DAS + Local Matching	25	25	24

Table 3. Retrieval result on Kimia Dataset 1[15]

In Kimia's first dataset [15], there are 25 images in 5 categories. Our method outperforms all other algorithms.

4.4 HSP, *e*-value and *p*-value

The *e*-value and *p*-value are used to evaluate the significance of a HSP (see eq.2, eq.3, and eq.3). To calculate K and λ, we collected 500 pairs of random shapes of different categories from Kimia's dataset. We fixed the similarity functions of eq.4 and eq.5, and fixed the number of feature points, i.e. the m, n of eq.3, and we use linear regression to estimate K and λ using eq.3. We plot the relation of $\ln E$ and S in the following figure and the regression result.

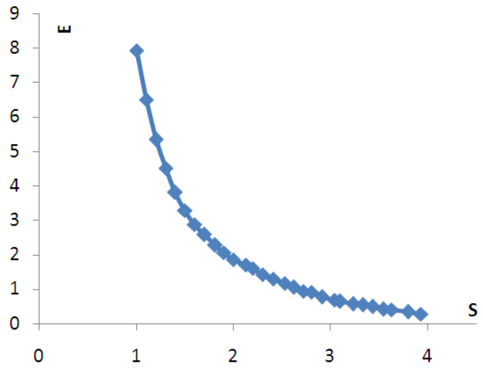


Figure 4. The relation of S, the score of HSP and E, the expected number of HSP, with fixed m=n=200.

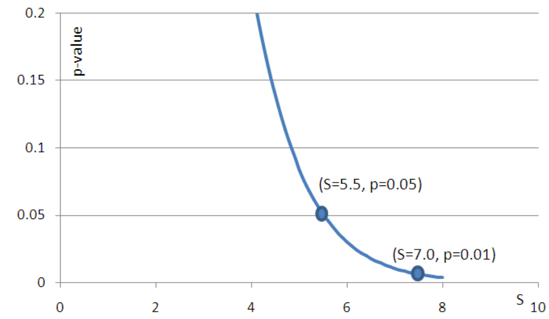


Figure 5. The relation of the score of HSP(S) and the p-value, with fixed m=n=200.

4.5 Occluded shape recognition with *p*-value

We did the similar occluded shape recognition experiment with Sebastian et al. [4]. We obtained the similar curve with [4]. The advantage of our method is that we can give the *p*-value of the matched shapes, which might be used as a confidence level of the matching results. From the figure, the original shape matches all occluded shapes with *p*-value less than 0.01, which means the probability that the original shape and occluded shape are two unrelated shapes is less than 0.01.

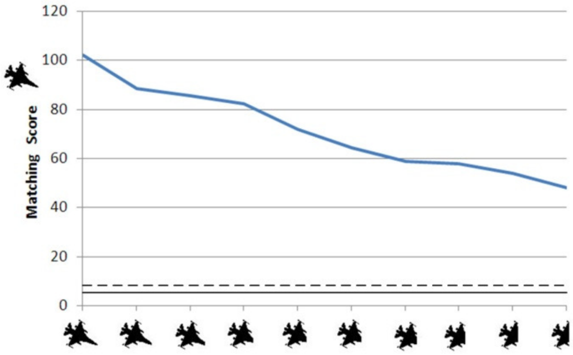


Figure 6. Similar experiment as in [4], the difference is that we have the p-value estimation. The blue curve is the matching scores, the black solid line is the score 5.5 with *p*-value 0.05, and the dash line the score 7.0 with *p*-value 0.01 (see Figure 5).

5. Conclusion

We show that our partial shape matching method, which is based only on similar parts of two shapes, is robust to shape variances, including rotation, scaling, and distortion and improve the recognition rate significantly.

Future work includes augmenting appearance features with shape descriptor for object detection; shape/object indexing for fast retrieval, and learning-based shape/object recognition.

References

[1] Smith TF, Waterman MS, Identification of Common Molecular Subsequences, Journal of Molecular Biology 147: 195-197. 1981.

[2] Haibin Ling, David W. Jacobs, Using the Inner-Distance for Classification of Articulated Shapes, CVPR 2005:

[3] S. Belongie, J. Malik, and J. Puzicha. "Shape Matching and Object Recognition Using Shape Contexts" PAMI 2002

[4] T. B. Sebastian, P. N. Klein and B. B. Kimia. Recognition of Shapes by Editing Their Shock Graphs, PAMI, 26(5):550- 571, 2004

[5] F. Bookstein. "Principal Warps: Thin-Plate-Splines and Decomposition of Deformations", PAMI, 11(6):567-585, 1989.

[6] Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990). "Basic local alignment search tool". J Mol Biol 215 (3): 403-410. PMID 2231712.

[7] Casey, RM, "BLAST Sequences Aid in Genomics and Proteomics". Business Intelligence Network, 2005.

[8] T.K. Peucker D.H. Douglas. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. The Canadian Cartographer, 10(2):112-122, 1973.

[9] Y.Gdalyahu and D.Weinshall, Flexible Syntactic Matching of Curves and its application to automatic hierarchical classification of Silhouettes, PAMI, 1999

[10] T. B. Sebastian, P. N. Klein and B. B. Kimia. "On Aligning Curves", PAMI, 25(1):116-125, 2003.

[11] E. G. M. Petrakis, A. Diplaros and E. Milios. "Matching and Retrieval of Distorted and Occluded Shapes Using Dynamic Programming", PAMI, 24(11):1501-1516, 2002.

[12] G. Mori and J. Malik, "Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA", CVPR, I:1063-6919, 2003.

[13] L. J. Latecki, R. Lakamper, and U. Eckhardt, "Shape Descriptors for Non-rigid Shapes with a Single Closed Contour", CVPR, :424-429, 2000.

[14] Y. Chi and M. K. H. Leung, "Part based object retrieval in cluttered environment," PAMI, vol. 29, pp. 1-6, 2007.

[15] D. Sharvit J. Chan, H. Tek, and B. Kimia. "Symmetry-based Indexing of Image Database", J. Visual Communication and Image Representation, 9(4):366-380, 1998.

[16] Remco C. Veltkamp Mirela Tanase. Part-based shape retrieval, 2005. Proceedings ACM Multimedia 2005, pp 543-546.2

[17] M. Bober, J. D. Kim, H. K. Kim, Y. S. Kim,W.-Y. Kim, and K. Muller. Summary of the results in shape descriptor core experiment. MPEG-7, ISO/IEC JTC1/SC29/WG11/ MPEG99/M4869, Vancouver, July 1999.

[18] F. Mokhtarian, S. Abbasi, and J. Kittler. Efficient and robust retrieval by shape content through curvature scale space. In A. W. M. Smeulders and R. Jain, editors, Image Databases and Multi-Media Search, pages 51-58. World Scientific Publishing, Singapore, 1997.

[19] L. J. Latecki and R. Lakamper. Shape similarity measure based on correspondence of visual parts. IEEE Trans. Pattern Analysis and Machine Intelligence, 10:1185-1190, 2000

[20] R. Basri, L. Costa, D. Geiger, and D. Jacobs, Determining the Similarity of Deformable Shapes, Vision Research 38:2365-2385, 1998

[21] T. S. Caetano, L. Cheng, Q. V. Le, and A. J. Smola. Learning Graph Matching. In International Conference on Computer Vision, 2007.

[22] Gumbel, E. J. Statistics of extremes. Columbia University Press, New York, NY. 1958

[23] Volker Roth and Bjorn Ommer, Exploiting Low-level Image Segmentation for Object Recognition, German Pattern Recognition Symposium, 2006

[24] Riedel, D. E., Venkatesh, S., and Liu, W. A, Smith-Waterman Local Alignment Approach for Spatial Activity Recognition. In Proceedings of the IEEE international Conference on Video and Signal Based Surveillance, 2006.