ETH ZURICH, D-INFK

Computer Vision Exercise 7: Shape Context

Xingchang Huang

December 7, 2017

1 SHAPE MATCHING

Shape Context method is designed to measure similarity between shapes and recognition. Generally, we should sample points from two compared shapes, get the descriptors of both and matching the descriptors by cost matrix, Hungarian algorithm, and reduce cost by Thin Plate Splines.

1.1 SHAPE CONTEXT DESCRIPTOR

First, we should use sampling points to detect descriptors for each two shapes. That is, we need to iterate all the image pairs in the dataset and each time we choose two of them to compute the shape context descriptors, cost matrix and then match the corresponding points to reduce the cost if they are in the same shape (e.g. Heart). Therefore, for the chosen 100 sample points, we should compute the descriptor for each point by calculating the distances and angles to other points as drawn in the slides. In details, in the log space between 1.8 and 3 with 5 bins, my implementation is that if the points is within a bin, it should be recorded an index corresponding to that bin. Thus we have 5 bins for distance. Then for the angles, I compute the arctan value between points and change the value to $[0,2\pi)$ and find the bin from 0 to 360 with 12 bins. In this way, in each shape we both find two bin matrices, one is for distance and the other is for angles. Next we can combine them into a final histogram with $12*5 = 60$ bins in total so we just need to add values for position indexed by corresponding elements of two matrices and we can get the numbers representing which bin those points belong to for a reference point and we return a descriptor matrix of $N*bins$ for all sample points.

1.2 Cost Matrix

After getting the descriptor matrix for both two shapes, we need to calculate the cost matrix between descriptors in order to use Hungarian Algorithm, which has been given. Here we want to return a $N \times N$ matrix for corresponding cost, where each row represent the chi-square costs to other descriptors. Based on that cost matrix we can use Hungarian algorithm then. Here it should be noticed that while calculating the chi-square cost, the denominator may become zeros so we need to add an epsilon to it.

1.3 Thin Plate Splines

Next, we can use Thin Plate Splines (TPS) model to transform the matched points to reshape that shape in order to reduce the cost between similar shapes. Following the equation, we need to solve two TPS model, one for x-coordinate and another for y-coordinate. Thus, we should solve the system with w_x and v_x, w_y and v_y, respectively. And then we should aggregate the cost term E by summing up E_x and E_y. It should be noticed that the returned w_x should contain vector (a_1, a_x, a_y) as well. The K matrix can be computed efficiently using given dist2 function. And here are some examples for transforming the shapes as shown in Figure 1. As shown in the following graphs examples, those blue points would be warped to those matched red points. However, similar shape or the same shape would match and wrap better with much less error I_f.

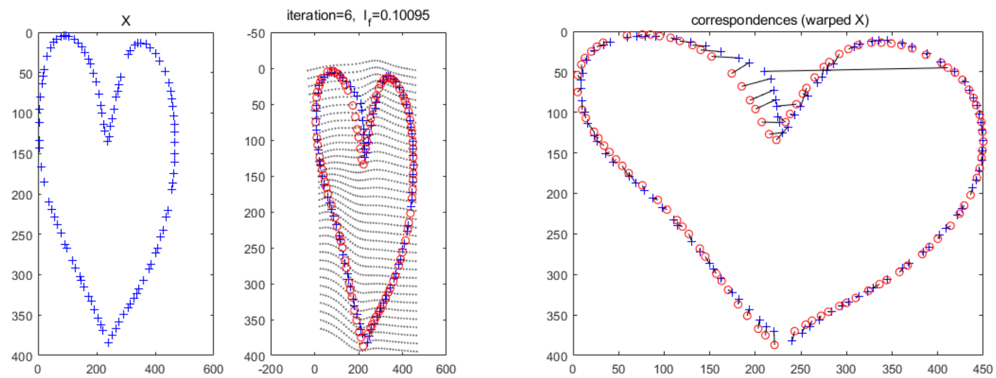

Figure 1.1: Heart-Heart Matching, Warped.

2 Shape Classification

In this step and can use the cost matrix to classify that whether one of the 15 objects can be classified correctly to its true label, where the element of cost matrix (15×15) has been calculated in TPS model as E. We can therefore easily find the k-nearest neighbours with smaller cost values. If $k = 1$, we just need to find the index with smallest value. Otherwise, I would use majority voting for those chosen candidates with $k = 3, 5, 7....$ Noticed that k

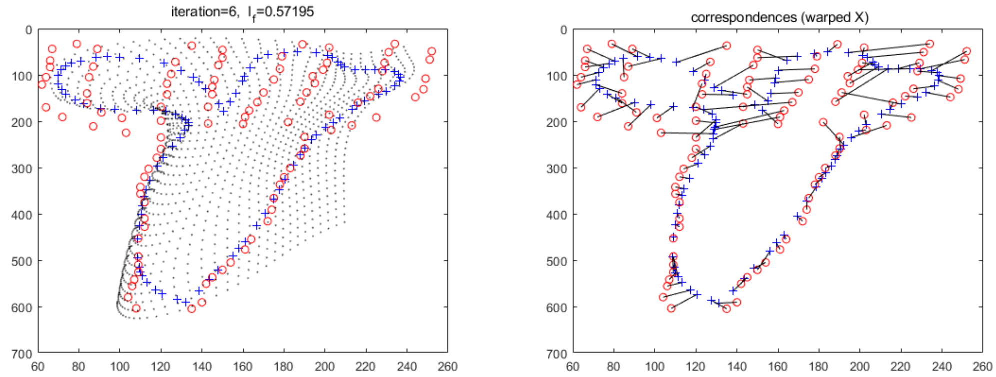

Figure 1.2: Heart-Fork Matching, Warped.

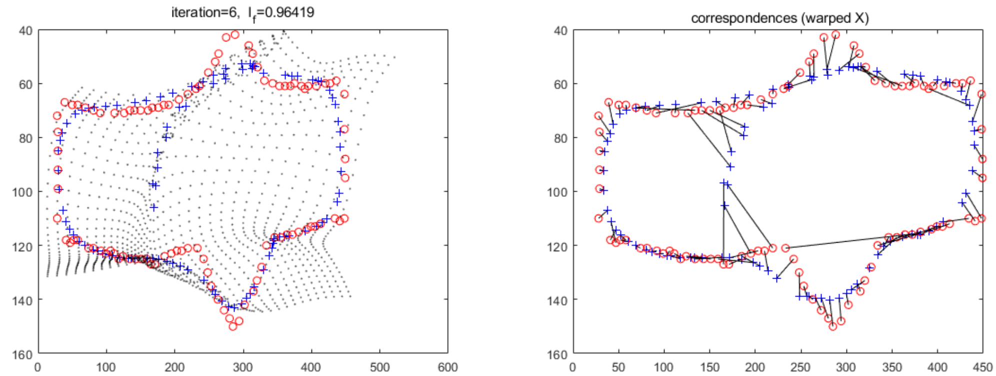

Figure 1.3: Heart-Watch Matching, Warped.

should be a odd number because it would be ambiguous in the case that if k is even with the same votings for two different labels. Here we have another variable, which is the way of sampling points. We can use both random sample and the sampling method used in this paper. Here is the parameters and the corresponding results. For $k = 1$, the true positive is like equal to 14/15, and the table shows the result we can reach. Sample 1 method means random sampling and 2 means the proposed method used in the paper. The iteration for all experiments are set to be default value as 6 and 1 as well. The results of them are both listed in tables below. From the result we can conclude that the sampling method proposed in

sample	k=1	k=3	k=5	k=7
1	10,14/15	13,14/15	10,13/15	9,11/15
2	14/15	13/15	13,14/15	10,13/15

Table 2.1: Iteration=6: Parameters of k and sampling methods comparison for true positive results.

sample	k=1	k=3	k=5	k=7
1	13/15	14/15	14/15	14/15
2	14/15	14/15	14/15	14/15

Table 2.2: Iteration=1: Parameters of k and sampling methods comparison for true positive results.

this paper is slightly more robust for finding the true correspondences as the true positive is stable and $k = 1$ would be also a better choice for this dataset. In the next section I would talk about the advantage of the proposed sampling method.

3 QUESTIONS ANSWERING

1. Is the shape context descriptor scale-invariant? Explain why or why not. I suppose that the context descriptor is scale-invariant. Because if all the points are scaled by a factor, the value of arctan between p_i and q_i would not change. And for distance would not change as well because we have implemented the normalization of all radial distances by the mean distance between all pairs sampled from the shape for increasing robustness. In this way, suppose we have distance d_1, d_2 in log space. The normalized value of d_1 should be $\frac{2d_1}{d_1+d_2}$. After scaling, the distance normalized value of d_1 should be $\frac{2sd_1}{sd_1+sd_2}$, which is the same by eliminating the s from both the nominator and denominator.

2. What is the average accuracy of your classifier? I have repetitions for experiments when $k = 1$ and I found that my implementation's average accuracy with random sampling should be 12/15 and the average accuracy with proposed sampling method should be 14/15. Also, I have tried iteration that is equal to 1, the result can also reach 14/15 as mentioned above. But in general, it looks better when k is less than 7 and the average performance for proposed sampling method would be better.

3. How does your classification accuracy vary with the number of neighbours k? As shown in table 2.1, when k increases to 7, the accuracy decreased to 9/15 using random sampling when the number of iterations is 6. And from the table, we can see that $k = 1$ for proposed sampling method should be a better and more stable choice for this dataset. And the best result I can reach may be 14/15 in true positive.

4. If instead of your own sampling function, you use the one that we provide (get_samples_1.m), do you get better classification results? Why or why not? Yes. The proposed and provided sampling method can give better result for my implementation. As shown in the paper, they have found that the sample points having a minimum distance between them would ensure sampling along the contours to be somewhat uniform. However, the random sample can not guarantee that for each sampling process we can sample uniform points for the shape contour. This should be the difference of these two methods that makes the result to be different.