Lab Assignment - Shape Context

Kaiyue Shen

December 4, 2020

1 Introduction of the Pipeline of Shape Matching

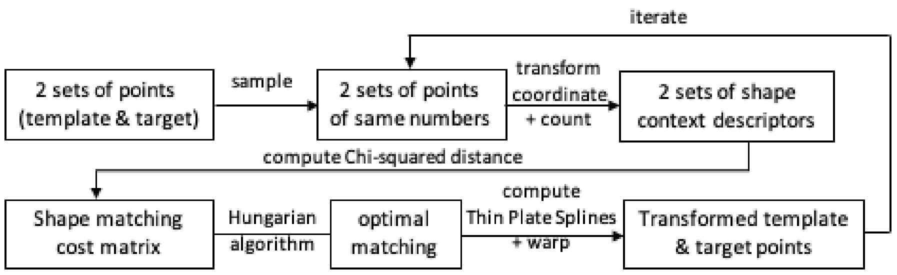

iterate
2 sets of points (template & target)
sample
2 sets of points of same numbers
transform coordinate + count
2 sets of shape context descriptors
compute Chi-squared distance
Shape matching cost matrix
Hungarian algorithm
optimal matching
compute Thin Plate Splines + warp
Transformed template & target points

Figure 1: Pipeline of Shape Matching

Based on my understanding, the Shape Matching is consisted of a sequence of operations:

1. **Sample**. Given two sets of 2D image points from the template and target, we draw samples from them in order to make sure two sets have same amount of points since Hungarian algorithm requires a square cost matrix.
2. **Transform coordinate + count**. We extract the shape context descriptors for all points, which are done by two steps: 1. do a local log-polar coordinate transformation and separate the coordinate space into wanted number of bins, 2. count the number of points in each bin.
3. **Compute χ^2 distance**. We compute the Chi-squared distance between two sets of shape context descriptors to get a shape matching cost matrix. The detailed formula is shown in formula 1, where g and h are two extracted shape context descriptors from template and target, K=*nbBins_theta* $\times$ *nbBins_r*.
4. **Find optimal matching**. We use Hungarian algorithm to solve the bipartite graph matching, in our case, is to find the optimal matching between template points and target points.
5. **Compute thin place slines + warp**. Given correspondences between points on two shapes, we can proceed to estimate a plane transformation. We compute two thin plate splines (TPS) models separately for x and y coordinate transformation and use the models to warp the original template. It has been shown in the given paper that TPS interpolant $f(x, y)$ that minimizes the bending energy has the following form:

$$f(x,y) = a_1 + a_x x + a_y y + \sum_{i=1}^{n} w_i U\left(\|(x_i, y_i) - (x, y)\|\right)$$

After a series of derivation, we only need to solve a system of the form as shown in formula 2, where $K_{ij} = U\left(\|(x_i, y_i) - (x_j, y_j)\|\right), U(r) = r^2 \log r^2$, the i-th row of P is $(1, x_i, y_i), \omega$ and a are parameters of the TPS model, which are what we want. v are points on the target shape and λ is the square of the mean distance between two target points used as regularizer. The matching cost, in other words, bending energy can be computed using formula 3.

6. **Iterate**. In many cases, the initial estimate of the correspondences contains some errors which could degrade the quality of the transformation estimate. The steps of recovering correspondences and estimating transformations can be iterated to overcome this problem. We use a fixed number of iterations.

$$C_{gh} = \frac{1}{2} \sum_{k=1}^{K} \frac{[g(k) - h(k)]^2}{g(k) + h(k)} \tag{1}$$

$$\begin{pmatrix} K + \lambda I & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} \omega \\ a \end{pmatrix} = \begin{pmatrix} v \\ 0 \end{pmatrix} \tag{2}$$

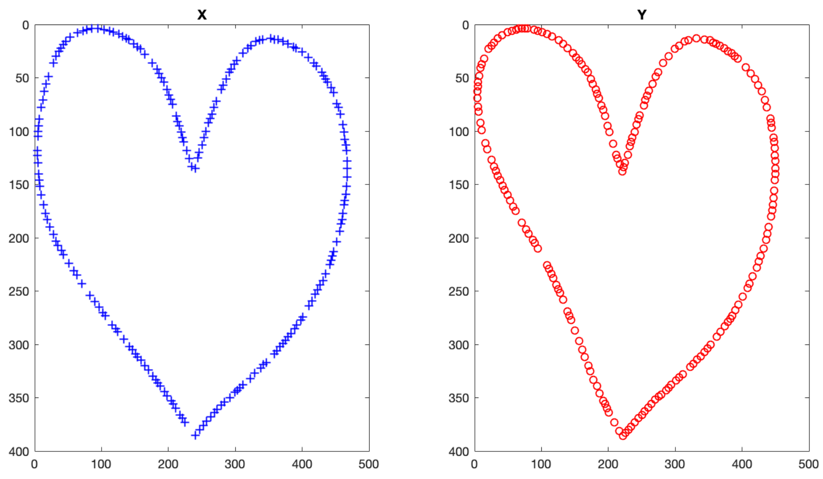
$$E = \omega_x^T K \omega_x + \omega_y^T K \omega_y \tag{3}$$

I implement the whole pipeline as shown in figure 1 and the results of main steps are shown in the following section.

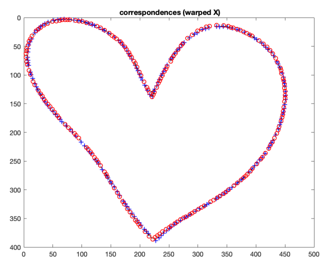
2 Experiment results

I draw 200 samples from each shape in this experiment part.

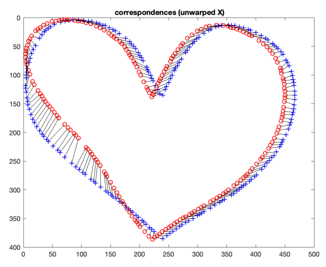
2.1 Heart



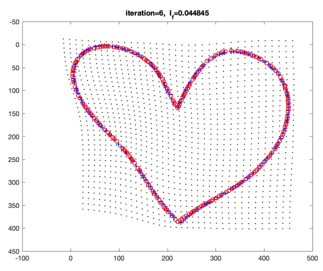
(a) Left: template samples, Right: target samples



(b) Warped correspondences

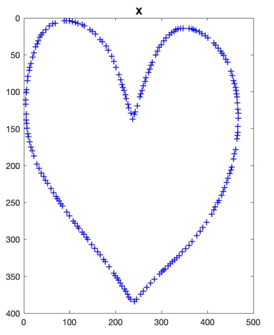


(c) Unwarped correspondences

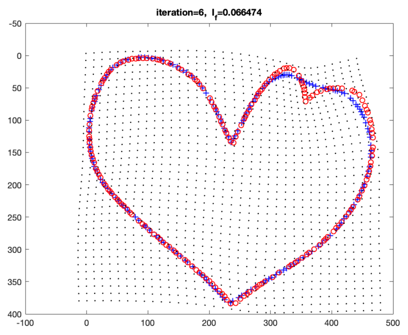


(d) Warped coordinate grid

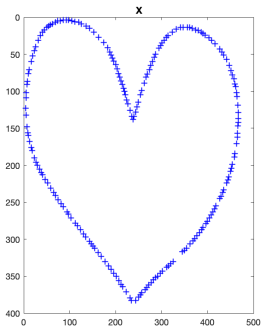
Figure 2: Shape matching result at iteration 6, Template=1, Target=2, matching cost=0.044845



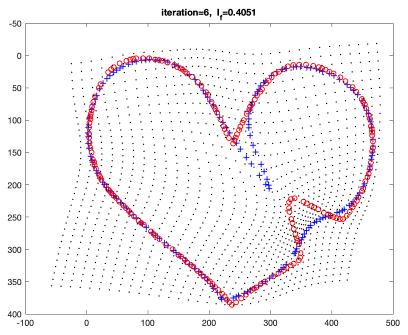
(a) Left: template samples, Right: target samples



(b) Warped coordinate grid



(c) Left: template samples, Right: target samples



(d) Warped coordinate grid

Figure 3: Shape matching result at iteration 6, (a)(b) Template=1, Target=3, matching cost=0.066474, (c)(d) Template=1, Target=5, matching cost=0.4051

In figure 2, we show one shape matching result of "Heart" and we can see the matching goes well: the template almost transform into the target shape. In figure 3, we display some other matching results in "Heart" dataset. It can be seen that depend on how template and template shape differs, the matching result varies. More different they are, poorer matching we may get, which can be verified both qualitatively and quantitively.

2.2 Fork

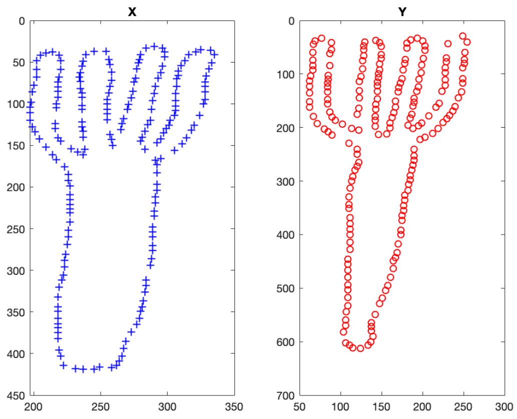
In figure 4, we show two pairs of shape matching results from "Fork" dataset. The first pair matches well while the second one shows bad performance. As in figure 4b, the template and the target are almost upside down, the matching result is bad, which shows the limit of this method.

2.3 Watch

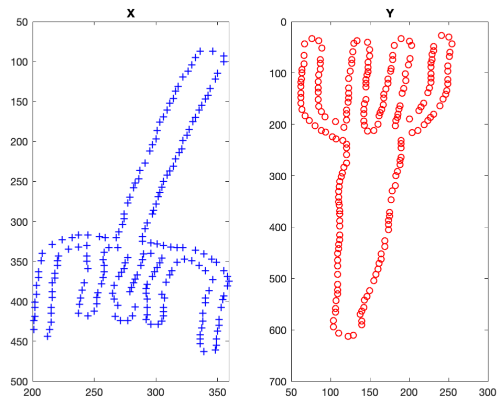
In figure 5, we display one shape matching result from "Watch" dataset. Generally, it works fine.

2.4 Additional Question

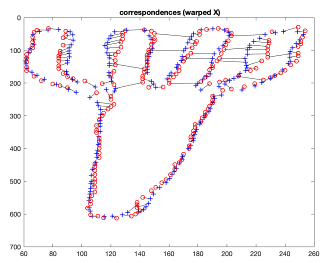
Is the shape context descriptor scale-invariant?
Yes. Since we implement the normalization of all radial distances by the mean distance of the distances between all point pairs in the shape when computing the shape context descriptors. If we scale the input shape by a factor of α, both the radial distance and mean will be scaled by α. So the ratio between them keep fixed however we vary the scale.



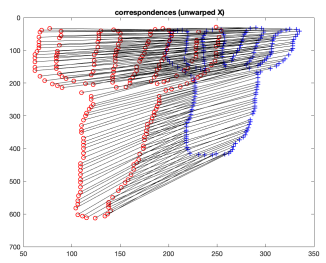
(a) Left: template=8, Right: target=6



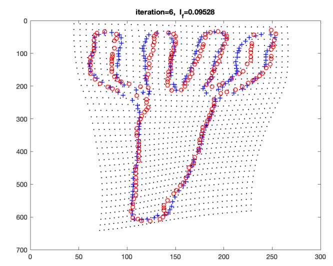
(b) Left: template=7, Right: target=6



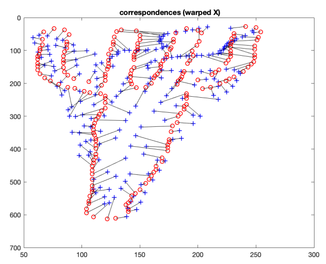
(c) Warped correspondences



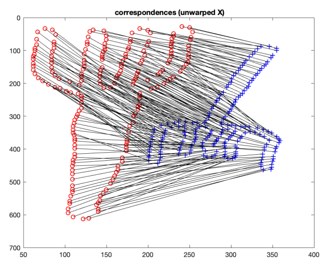
(d) Unwarped correspondences



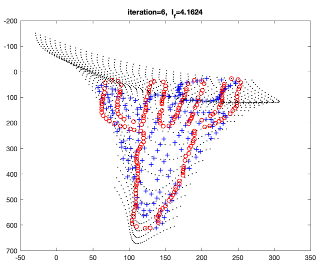
(e) Warped coordinate grid



(f) Warped correspondences

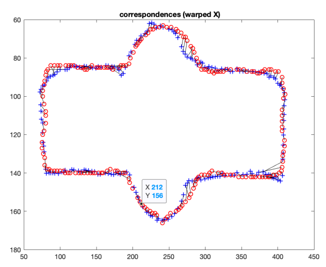


(g) Unwarped correspondences

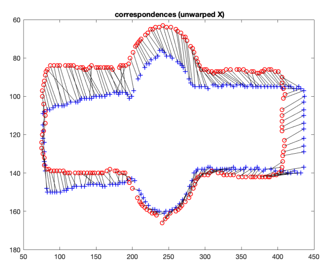


(h) Warped coordinate grid

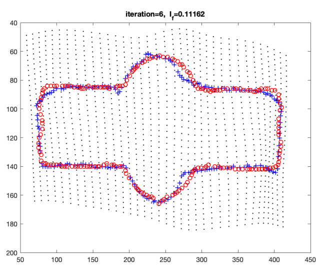
Figure 4: Shape matching result at iteration 6, (a)(c)(d)(e) template=8, target=6, matching cost=0.09528, (b)(f)(g)(h) template=7, target=6, matching cost=4.1624



(a) Warped correspondences



(b) Unwarped correspondences



(c) Warped coordinate grid

Figure 5: Shape matching result at iteration 6, Template=12, Target=14, matching cost=0.11162