Computer Vision

Shape Context

Student: Yuchang Jiang

Date: 4 Dec. 2020

1. Shape Matching

1.1 Implementation

This shape matching method can be divided into the following steps:

1. compute shape context descriptors for both template and target
2. with shape context descriptors, compute their cost matrix and use Hungarian algorithm to find correspondence
3. with correspondence, compute transformation from template to target

1.1.1 Step 1: shape context descriptors

For each point in template and target set, shape context descriptor is computed. The parameters used for shape context descriptor are:

- number of bins in the angular dimension, $nbBins\_theta$
- number of bins in the radial dimension, $nbBins\_r$
- the length of the smallest radius, $smallest\_r$
- the length of the biggest radius, $biggest\_r$

When computing the descriptor for each point, take this point as the origin of log polar coordinate system and use smallest and biggest radius to define the range. Then use number of bins in angular dimension to divide 2π into intervals and use number of bins in radial dimension to divide the range from biggest radius to smallest radius into intervals. In this way, each point has $nbBins\_theta * nbBins\_r$ intervals and the number of point falling into each interval are counted. The count number of intervals can form a histogram, which is the shape context descriptor for one certain point. To increase robustness, the normalization of all radial distances is implemented by the mean distance of the distances between all point pairs in the shape.

1.1.2 Step 2: cost matrix

With computed descriptors for template and target, the matching cost matrix between them can be computed using chi-squared distance:

$$C_{ij} = \frac{1}{2}\sum_{k}\frac{(p_i(k) - p_j(k))^2}{p_i(k) + p_j(k)} \tag{1.1}$$

where $C_{i,j}$ is the i-row j-column element in cost matrix, p_i and p_j are points from template and target. k means one certain bin in descriptor.

With the built matching cost matrix, Hungarian algorithm is used to find the point correspondence by minimizing the total cost.

1.1.3 Step 3: transformation

For transformation from template to target, Thin Plate Spline model is used:

$$f(x, y) = a_1 + a_x x + a_y y + \sum_i w_i U(||(x_i, y_i) - (x, y)||) \tag{1.2}$$

where $f(x, y)$ is the transformation function of point (x, y) in the shape for one dimension. U is the distance measurement and $U(t) = t^2(log(t^2)), U(0) = 0$.

In order to model the two dimensions, the formula used is:

$$T(x, y) = (f_x(x, y), f_y(x, y)) \tag{1.3}$$

Thus, the linear system, $Ax = b$ for this problem can be formed:

$$A = \begin{pmatrix} K + \lambda I & P \\ P^T & 0 \end{pmatrix},\ x = (w, a)^T,\ b = (v, 0)^T \tag{1.4}$$

where λ is the squared of mean distance of two target points, $K_{ij} = U(||(x_i, y_i) - (x_j, y_j)||)$ and in P, the i-th row is $(1, x_i, y_i)$. Thus, A matrix can be formed with known values. As for v, it is the coordinate x or y value for point in target. Then (w_x, a_x) and (w_y, a_y) can be solved with A, v_x, v_y. For bending energy, it can be computed as:

$$E = w_x^T K w_x + w_y^T K w_y \tag{1.5}$$

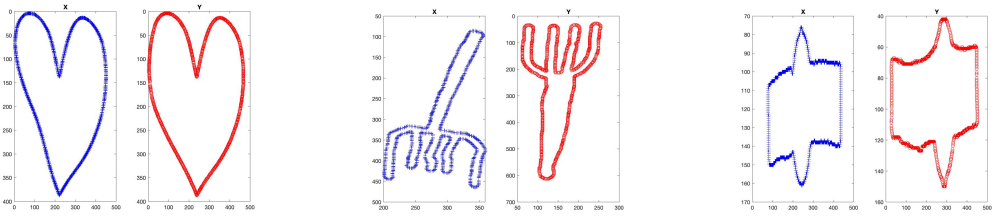
To summarize, the whole process is as the following:

- use $get\_samples.m$ to get points with defined sample number from template and target

- use $sc\_compute.m$ to compute shape context descriptors for both sets
- use $chi2\_cost.m$ to compute cost matrix
- use $hungarian.m$ to get correspondence matrix
- use $tps\_model.m$ to transform template into target
- update template coordinates and repeat shape matching steps until termination

1.2 Results and Discussion

Shape matching is performed on three shapes: heart, fork and watch with 6 iterations. The sample number used for heart, fork and watch are 1000, 1000 and 790. As shown in Figure 1, two images are chosen for each shape type. It is noticeable that template and target of heart and watch are quite similar but those of fork have different rotation angles.



(a) Heart (b) Fork (c) Watch

Figure 1: 3 shapes for experiment

For each shape, 6 iterations are tried for shape matching and results of first and last iteration are shown in Figure 2 - Figure 4 (column 1 is correspondence for warped points, column 2 is correspondence for unwarped points and column 3 shows the transformed points). Comparing three shapes, the result of heart is better than others. It is because the template and target of heart shape are already very similar. Although the result of fork is not as good as that of heart, it can capture the different rotation and transform template to target. Comparing different iterations, the result of last iteration seems to be better than that of first iteration. Based on Figure 5, the energy cost is generally decreasing with the increase of iteration.

For Question:

- Is the shape context descriptor scale-invariant?

My answer: Yes. When using log polar coordinate system in shape descriptor step, distances are normalized by mean distance of all point pairs. In this way, the shape with different scale will be normalized and so descriptor is scale-invariant.

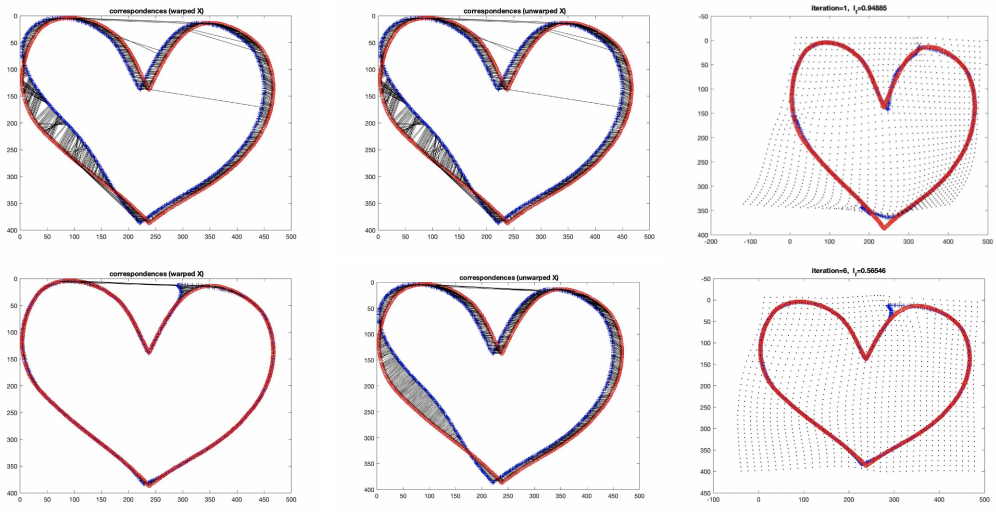


Figure 2: Result of heart shape

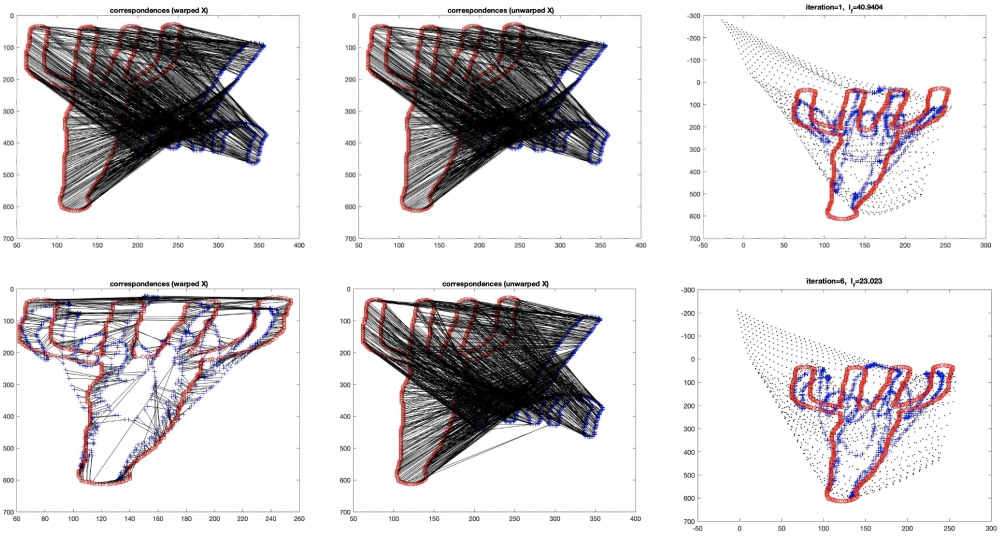


Figure 3: Result of heart shape

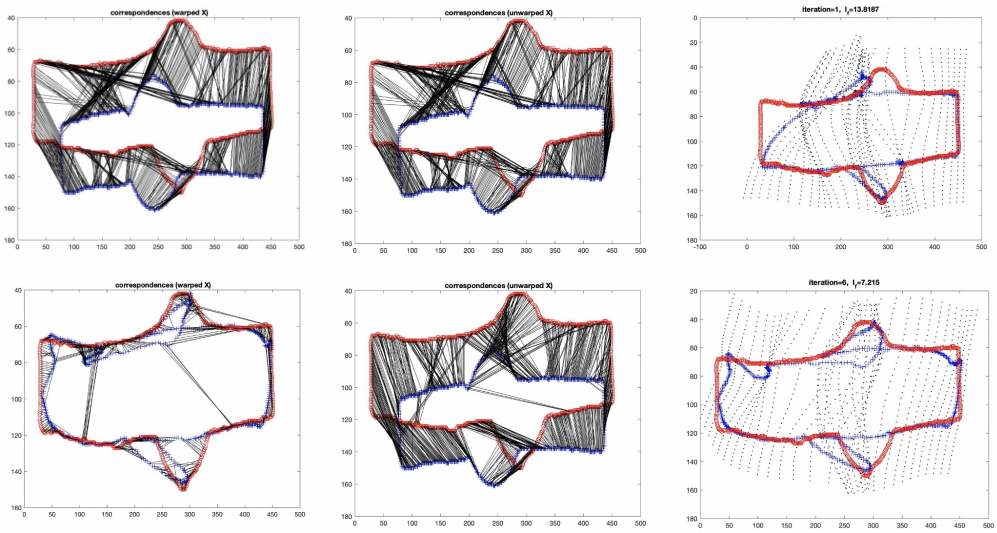


Figure 4: Result of heart shape

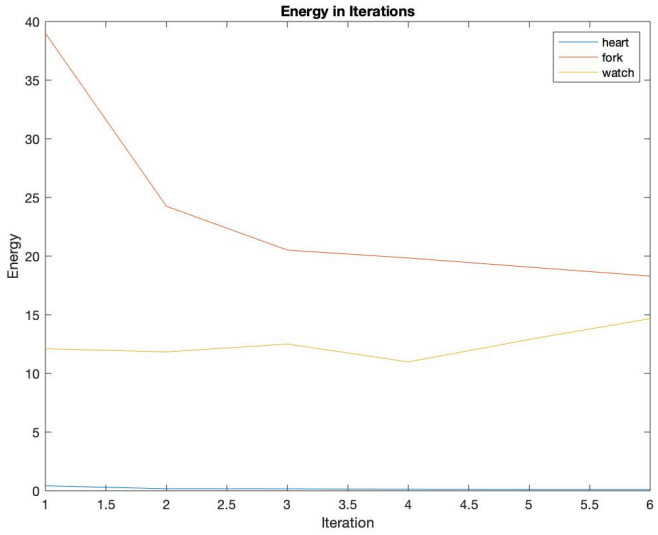


Figure 5: Plot of resulted energy