Yue Pan
yuepan@student.ethz.ch

Assignment 8

Computer Vision
November 30, 2019

Task 1: Shape Matching

In this task, we follow the algorithm proposed in the paper 'Belongie, S., Malik, J. and Puzicha, J., 2002, Shape Matching and Object Recognition Using Shape Contexts' to accomplish shape matching. The workflow of this algorithm is summarized in Fig.1.

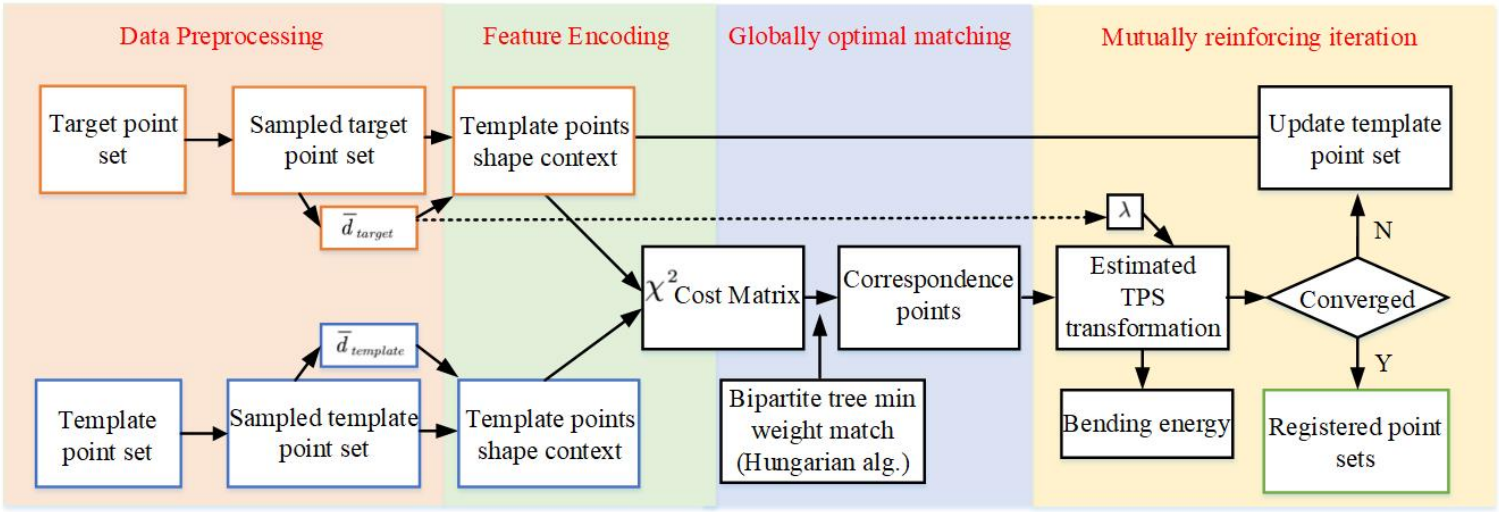


Figure 1: Framework of the shape matching algorithm of assignment 8

Follow the idea of shape context, for each point of the shape, we can compute a histogram of the relative coordinates (in polar coordinate system) of the rest points. By dividing θ and normalized r into several fractions, we can get several bins. Then we count the point number in each bin and get the histogram, which is regarded as the shape context of this point. Some examples are shown in Fig.2, in which the gray value of a pixel indicates the count of points in such bin.

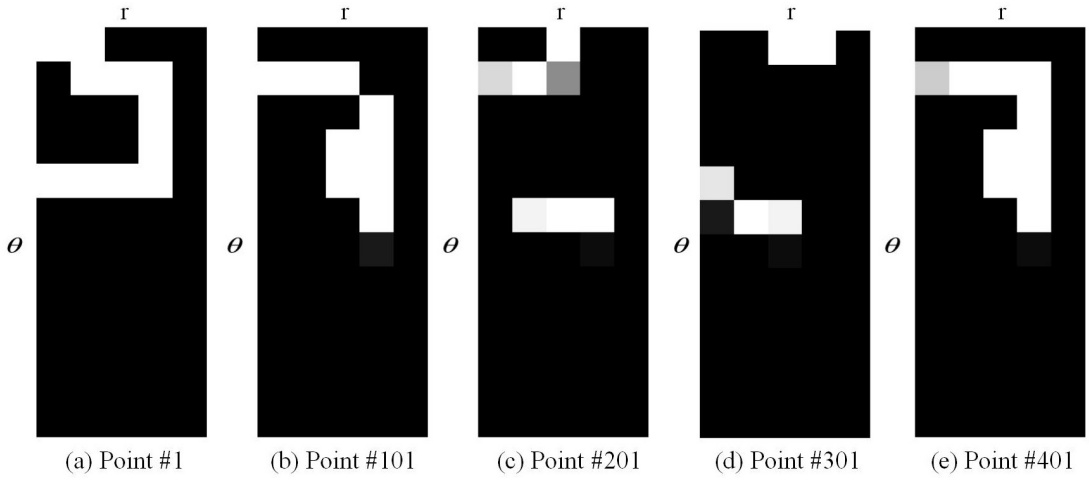


Figure 2: Examples of shape context generated in Heart Dataset #1 at the first iteration

The calculation of the cost matrix follow the chi-square cost calculation as shown in 1.

$$C_{i,j} = C\left(p_i, q_j\right) = \frac{1}{2} \sum_{k=1}^{\text{nbins\_}\theta \times \text{nbins\_r}} \frac{\left(s_{p,i}\left(k\right) - s_{q,j}\left(k\right)\right)^2}{s_{p,i}\left(k\right) + s_{q,j}\left(k\right)} \tag{1}$$

The calculation of the cost matrix follow the chi-square cost calculation as shown in Eq. 1.

Given the mutual cost matrix, the correspondence determination is done by achieving a bipartite graph minimum weight match, which can be solved using KM algorithm (or called Hungary algorithm). A problem is that the cost matrix (adjacent matrix of the bipartite graph) should be a square matrix in order to be properly processed by Hungary algorithm. So we have to do the sampling of the original point sets to make sure the point number in the template and the target point sets are identical. In practice, the sampling number is set as 1000 for Heart dataset and Fork dataset, 750 for Watch dataset.

Then we can estimate the transformation from the correspondences based on the Thin Plate Splines (TPS) model. The calculation follows the equations listed below: Eq.2-Eq.7. For correspondences set $\{p_i, p_j\}$, we can calculate the TPS transformation parameters $\omega_x, a^{(x)}, \omega_y, a^{(y)}$ using least square estimation. Using such TPS parameters, we can further transform the template point set. Besides, the bending energy can be calculated as Eq.7, which can be used to evaluate the shape matching result.

In most of the cases, the initial estimate of the correspondences contain some errors which could degrade the quality if transformation estimation. So we can iteratively do the recovering of correspondence and the estimation of transformation so as to overcome the problem. In practice, the final iteration number is set as 6.

As for the experiments of this algorithm, the datasets used is shown in Fig.3. The shape matching reuslt (correspondences, warped and unwarped shape of template point set) for all of the datasets are shown in Fig.4-Fig.6. Only the results of the first and the last iteration are demonstrated. For details about other iterations, please refer to the *results* folder. From the results, we can find for all of the three datasets, a decent shape matching is achieved. For the heart dataset, you can only notice one obvious false matching for the 6th iteration and the final bending energy is quite small (0.107). For the fork and watch dataset, due to the repetitive geometry, there's much more false correspondence output by Hungarian algorithm. However, for the fork dataset, the template point sets still manage to turn about 180 degree to have a good match with the target point set.

Generally speaking, the bending error would decrease throughout the iterative process, as shown in Fig.7.

$$U\left(t\right) = \begin{cases} t^2 \ln\left(t^2\right) = 2t^2 \ln\left(t\right), & t \neq 0 \\ 0, \; t = 0 \end{cases} \tag{2}$$

$$K_{i,j} = U\left(dist\left(p_i, p_j\right)\right) \tag{3}$$

$$\begin{bmatrix} \underset{n\times n}{K} + \lambda \underset{n\times n}{I} & \underset{n\times 3}{P} \\ \underset{3\times n}{P^T} & \underset{n\times 3}{0} \end{bmatrix} \begin{bmatrix} \underset{n\times 1}{\omega_x} \\ \underset{3\times 1}{a^{(x)}} \end{bmatrix} = \begin{bmatrix} \underset{n\times 1}{v_x} \\ \underset{3\times 1}{0} \end{bmatrix} \tag{4}$$

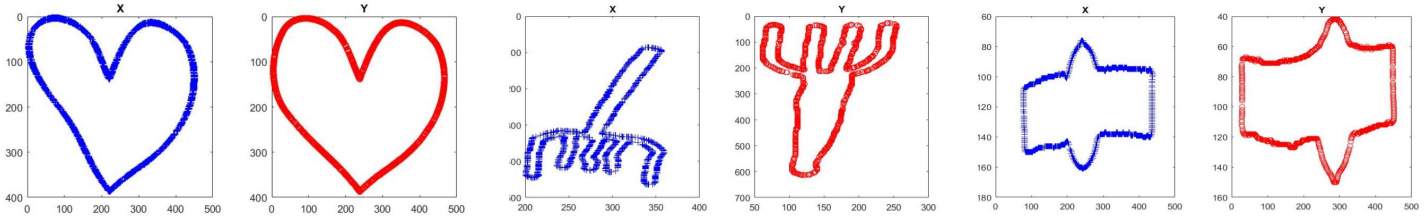
$$\begin{bmatrix} \underset{n\times n}{K} + \lambda \underset{n\times n}{I} & \underset{n\times 3}{P} \\ \underset{3\times n}{P^T} & \underset{n\times 3}{0} \end{bmatrix} \begin{bmatrix} \underset{n\times 1}{\omega_y} \\ \underset{3\times 1}{a^{(y)}} \end{bmatrix} = \begin{bmatrix} \underset{n\times 1}{v_y} \\ \underset{3\times 1}{0} \end{bmatrix} \tag{5}$$

$$P_i = \left(x_i^{(p)}, y_i^{(p)}, 1\right) \tag{6}$$

$$v_j^{(x)} = x_j^{(q)}, v_j^{(y)} = y_j^{(q)} \tag{7}$$

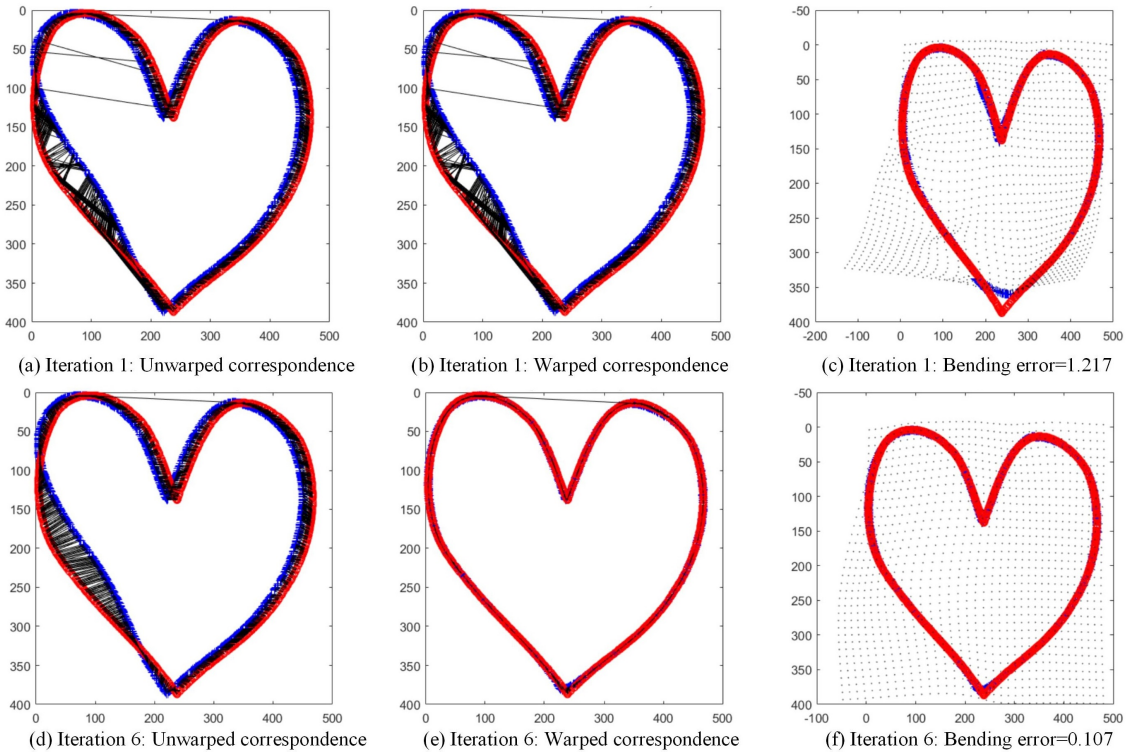
$$a^{(x)} = \left(a_x^{(x)}, a_y^{(x)}, a_1^{(x)}\right), a^{(y)} = \left(a_x^{(y)}, a_y^{(y)}, a_1^{(y)}\right) \tag{8}$$

$$E = E_x + E_y = w_x^T K w_x + w_y^T K w_y \tag{9}$$



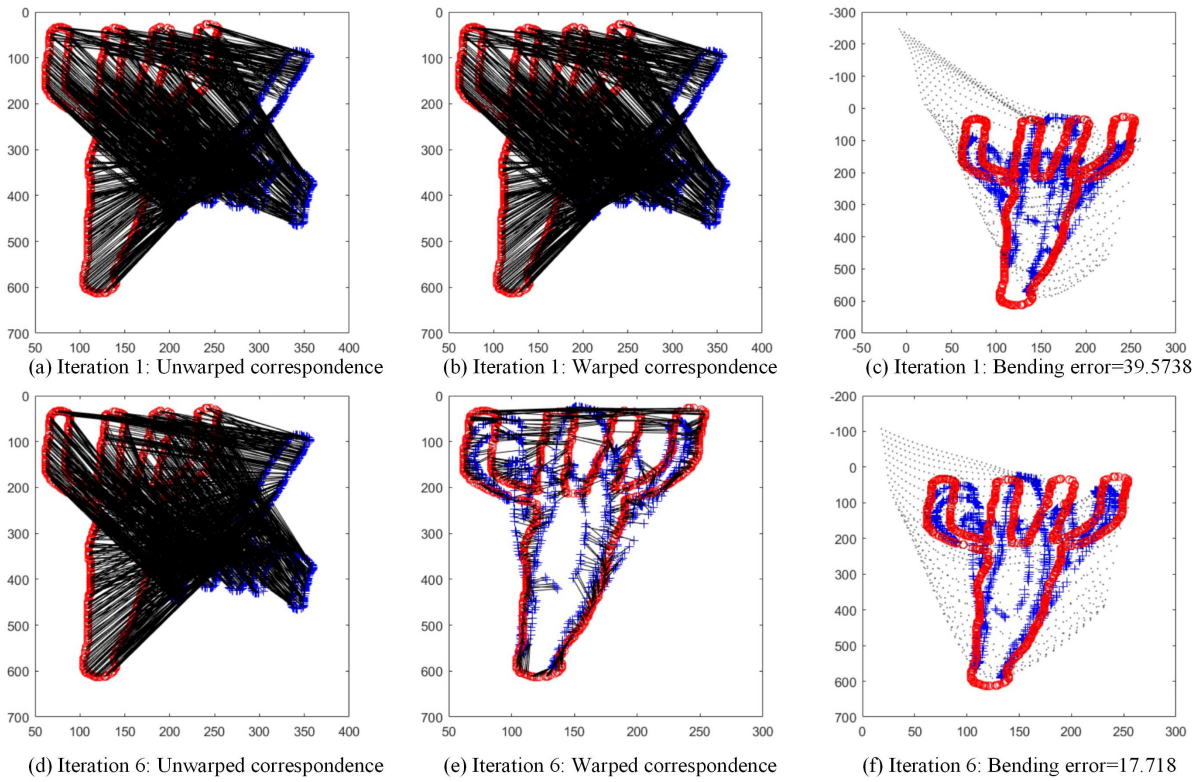
(a) Heart Dataset (Target: #1, Template: #2, Sample:1000)
(b) Fork Dataset (Target: #6, Template: #7, Sample:1000)
(c) Watch Dataset (Target: #11, Template: #12, Sample:750)

Figure 3: Experiment Data



(a) Iteration 1: Unwarped correspondence
(b) Iteration 1: Warped correspondence
(c) Iteration 1: Bending error=1.217
(d) Iteration 6: Unwarped correspondence
(e) Iteration 6: Warped correspondence
(f) Iteration 6: Bending error=0.107

Figure 4: Heart Dataset (Only Iteration 1 and 6 are shown)

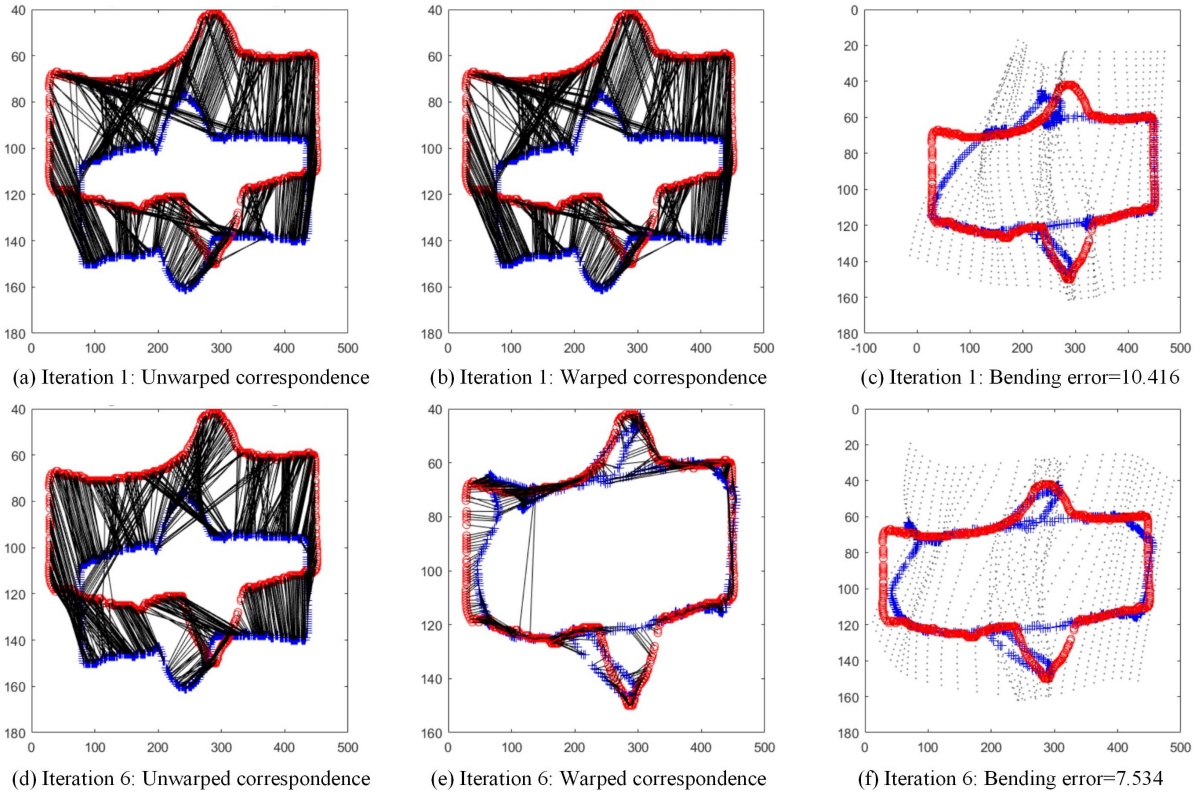


(a) Iteration 1: Unwarped correspondence (b) Iteration 1: Warped correspondence (c) Iteration 1: Bending error=39.5738

(d) Iteration 6: Unwarped correspondence (e) Iteration 6: Warped correspondence (f) Iteration 6: Bending error=17.718

Figure 5: Fork Dataset (Only Iteration 1 and 6 are shown)

Question: Is the shape context descriptor scale-invariant? Yes, it's scale-invariant. This is mainly due to the normalization done to distance r to the processing point. The normalized scale is calculated using the average distance between all points in the point set. So scaling the shape by a factor would not change the distance that we used to generate the shape context.



(a) Iteration 1: Unwarped correspondence (b) Iteration 1: Warped correspondence (c) Iteration 1: Bending error=10.416

(d) Iteration 6: Unwarped correspondence (e) Iteration 6: Warped correspondence (f) Iteration 6: Bending error=7.534

Figure 6: Watch Dataset (Only Iteration 1 and 6 are shown)

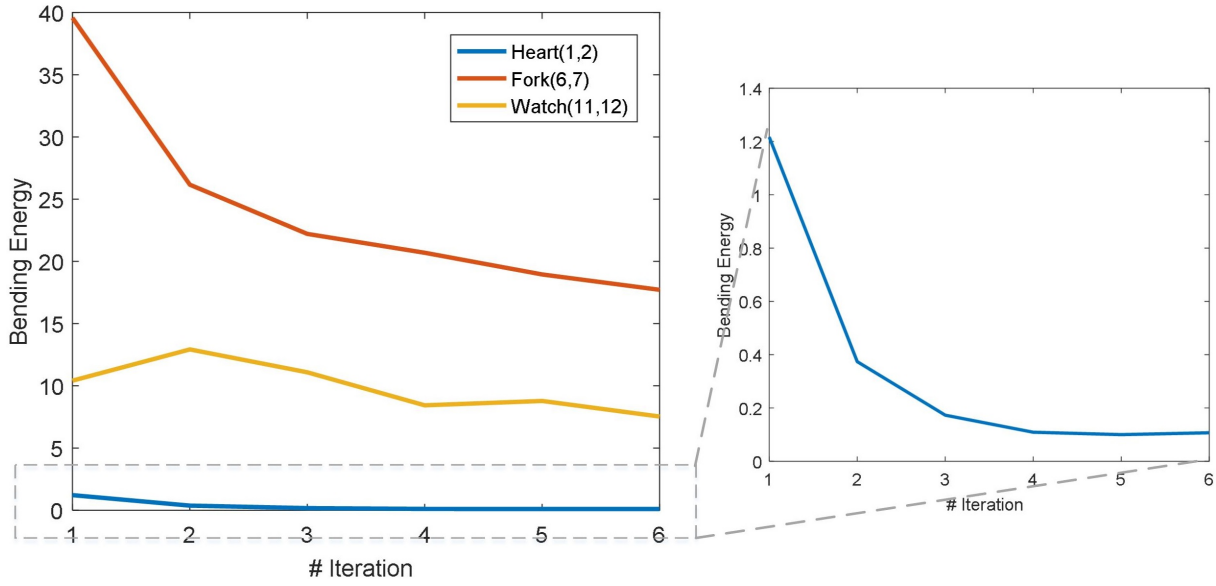


Figure 7: Bending energy w.r.t. iteration number for the three datasets