An Automatic Stroke Extraction Method using Manifold Learning

Xudong Chen, Zhouhui Lian[†], Yingmin Tang, Jianguo Xiao

Institute of Computer Science and Technology, Peking University, PR China

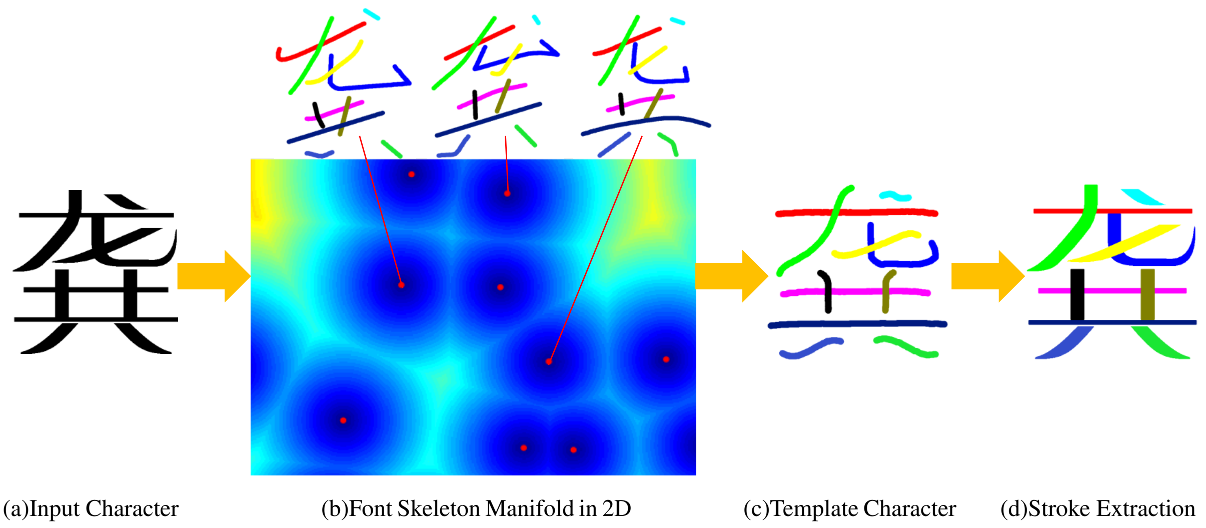

Figure 1: Overview of the proposed method for stroke extraction. (a) The input image of the Chinese character ’gong’; (b) Building a 2D manifold of the character learnt from 28 fonts. Every point in the manifold stands for a character in an unknown font style. The darker color in the manifold indicates a higher probability of a location containing a good font style. The red dots show the existing font styles, and the characters above are created from the corresponding locations. (c) The template character that is the most similar to the input character in font style. By traversing the manifold, we find the corresponding location that creates the template character. (d) Final stroke extraction results by registering points of the template character to the input character.

Abstract

Stroke extraction is one of the most important tasks in areas of computer graphics and document analysis. So far, data-driven methods are believed to perform relatively well, which use the pre-processed characters as templates. However, how to accurately extract strokes of characters is still a tough and challenging task because there are various styles of characters, which may vary a lot from the template character. To solve this problem, we build a font skeleton manifold in which we can always find a most similar character as a template by traversing the locations in the manifold. Because of the similar structure and font style, the point set registration of the template character with the target character would be much more effective and accurate. Experimental results on characters in both printing style and handwriting style reveal that our method using manifold learning has a better performance in the application of stroke extraction for Chinese characters.

Categories and Subject Descriptors (according to ACM CCS): I.4.6 [Image Processing and Computer Vision]: Segmentation—Region growing, partitioning

1. Introduction

In the task of character shape decomposition, a character shape is usually segmented into several basic parts, each part representing

[†] Corresponding author. E-mail: lianzhouhui@pku.edu.cn

a stroke designed by artists. Thus, character shape decomposition is also called stroke extraction, which is significant in applications such as offline character recognition, writing style analysis and new font synthesis [LZX16]. Yet, the problem of stroke extraction has not been well solved so far in the literature mainly because of the complexity of the character structure. Take Chinese characters as an example, many characters are so complicated with strokes touching and crossing with each other, that even a professional artist could not extract the strokes well without efforts.

Currently, existing methods for stroke extraction of characters can be mainly classified into two categories: unsupervised methods and data-driven methods. Most unsupervised methods use feature points and contour information without using template characters. For instance, [WLSL13] extracts the contours of the character first, and then the region within the boundary is converted to triangular meshes by using Constrained Delaunay Triangulation, after which the character shape is segmented into sub-strokes by junction triangles. According to the stroke continuity analysis, a stroke is finally extracted by composing sub-strokes. In [YWY12] and [SQX14], the contours of the character are segmented by analyzing feature points and tracing orders, and they directly use the contour segments to obtain the corresponding stroke. The inherent problem of the unsupervised methods is that only basic strokes (e.g., the horizontal stroke) can be extracted and other composed strokes would be wrongly segmented into several basic strokes. However, data-driven methods avoid the problem by using reference data. In [CLTX16] [LZX16] [WLTX13], a standard GB2312 Kaiti font library is used as the reference data which has independent stroke information with corresponding stroke order. The thinning algorithm is applied first, and then the authors use point set registration algorithms (e.g., Coherent Point Drift (CPD) [MS10]) to match the skeleton points of the target character to those of the reference character (or template character). Because the stroke information of the reference character is clear, strokes of the target character can be easily obtained according to the point set registration results.

The stroke extraction results of the data-driven methods mentioned above mainly depend on the correspondence that the point set registration algorithm finds between the target and the template character. However, structures of the same character in different font styles vary a lot from each other, which makes it difficult for point set registering algorithms to get the correct correspondence. It is obvious that better correspondences can be found between characters in similar structures and font styles. To this end, this paper aims to promote the stroke extraction accuracy by finding a proper character template similar in structure and font style to the target. As shown in Figure 1, for each input character with any font style, we build a font skeleton manifold learnt from 28 fonts including both printing styles and handwriting styles, where each location indicates a novel character skeleton in an unknown font style. By traversing in the manifold and comparing the feature similarities between the corresponding character of each location and the target character, we select the most similar one as the template character. Then, reliable correspondences between the stroke skeleton points in the template character and in the target character can be found by applying the point set registration algorithm. Finally, according to the correspondences, our method can effectively and accurately extract strokes from the target character. Although this paper concentrates on Chinese characters, the proposed method can be generalized to other kinds of characters (e.g., English).

2. Method Description

2.1. Learning a Font Skeleton Manifold

The thinning algorithm is always applied to get the writing trajectory or the skeleton of a character, which is considered as one of the most important features of a stroke. Using the skeleton instead of the contour of a character simplifies the complexity of building a font manifold. We match the skeleton points of each individual character across all fonts using a point set registration procedure with our stroke skeleton models. Then we use the dense skeleton point correspondences for each character as a basis to fit a nonlinear manifold that ties the characters in different font styles together into a single space.

2.1.1. Character Matching

Specifically, we construct a high dimensional vector that contains the stroke skeleton points to identify each character. Here comes the problem of finding point correspondences among different fonts which is necessary for learning a manifold. Instead of using the energy model [CK14], we choose a more effective way by using the point set registration algorithm. To figure out the exact correspondence, we build a stroke model database which classifies the strokes of all Chinese characters into 339 categories. We manually select the key points on the writing trajectory of each stroke model and then get its revised skeleton with no fork points. There are mainly three kinds of skeleton key points: start point, end point and corner point, which contains information of writing styles. The skeleton segment between each key point pair is then uniformly sampled into a certain number of points. We applied the tool in [CLTX16] and manually obtained the stroke skeleton points of each character across 28 selected fonts. In the next step, we directly use CPD to register the skeleton points of the input stroke to its corresponding stroke model. As we see in Figure 2, the key points are aligned to the model and the skeleton segment between the key point pair is sampled into the same number of points.

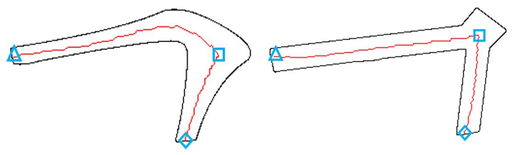

Figure 2: Key points alignment between the stroke model in our database (left) and the stroke in FS font style (right). The triangles, diamonds and squares denote the corresponding start points, end points and corner points of the two stroke skeletons above.

Since the points on every stroke skeleton have been aligned with each other, the correspondences of each character among 28 selected fonts are established.

2.1.2. Training the GP-LVM

The Gaussian Process Latent Variable Model (GP-LVM) [Law05] is an effective non-linear, dimensionality reduction technique. It produces a probabilistic model of a high dimensional dataset Y with a low dimensional dataset X which is 'latent'. We are working in a very high dimensional space at the beginning since there are about 600 skeleton points in each character on average. Therefore, the GP-LVM is well suited to learning such a font skeleton manifold.

Suppose there are M fonts, to generate M high dimensional vectors for each character, each vector is composed by putting all stroke skeleton samples together in stroke order as

$$\mathbf{v}_m = \left[\begin{bmatrix} \mathbf{v}_{1,1}^m \\ \mathbf{v}_{1,2}^m \\ \vdots \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \mathbf{v}_{2,1}^m \\ \mathbf{v}_{2,2}^m \\ \vdots \end{bmatrix}^{\mathrm{T}} \cdots \begin{bmatrix} \mathbf{v}_{n,1}^m \\ \mathbf{v}_{n,2}^m \\ \vdots \end{bmatrix}^{\mathrm{T}} \right]^{\mathrm{T}}, \quad (1)$$

where $\mathbf{v}_{i,j}^m$ denotes the j-th point on the i-th stroke skeleton in font style m and n is the stroke number of the character. The value of the point coordinates should be normalized into $[0,1]$. To apply GP-LVM, we subtract the mean vector $\bar{\mathbf{v}}$ off to get the vector $\mathbf{y}_m$ as

$$\mathbf{y}_m = \mathbf{v}_m - \bar{\mathbf{v}}, \bar{\mathbf{v}} = \mathbf{E}_m[\mathbf{v}_m]. \quad (2)$$

This allows us to use a zero mean function, and the mean vector $\bar{\mathbf{v}}$ is actually the average character skeleton from the training data. Then the high dimensional dataset Y can be formulated as

$$Y = \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_i & \cdots \end{bmatrix}^{\mathrm{T}}. \quad (3)$$

Using these skeleton vectors for Y, the training process for the GP-LVM considers the likelihood of Y as

$$P(Y|X,\theta) = \prod_{i=1}^{M} N(\mathbf{y}_i|\mathbf{0}, C(X,X|\theta) + \sigma^2 I), \quad (4)$$

where I is the identity matrix, M is the total font number, and σ is a noise variance that accounts for miss matches. In our experiments, the manifold works well only when the parameter σ is set to be small (e.g., 0.1), which suggests that the font skeletons actually lie on a low dimensional manifold. We get the latent variables by maximizing the likelihood below jointly over the latent vectors $X = [\cdots \mathbf{x}_j \cdots]^{\mathrm{T}}$ as well as the hyperparameters θ so that

$$X^*, \theta^* = \arg\max_{X,\theta} [\log(P(Y|X,\theta))]. \quad (5)$$

Generating a new skeleton font from the manifold is straightforward and simple with the latent variables X^* and hyperparameters θ^* [Law05]. Suppose $\hat{\mathbf{x}}$ is the target location on the manifold, thus the corresponding high dimensional vector $\hat{\mathbf{y}}$ can be calculated by the following equation as

$$\hat{\mathbf{y}} = C(\hat{\mathbf{x}}, X^*|\theta^*)[C(X^*, X^*|\theta^*)]^{-1} Y, \quad (6)$$

where $C(\hat{\mathbf{x}}, X^*|\theta^*)$ denotes the covariance between vectors and $[C(X^*, X^*|\theta^*)]^{-1}$ is precomputed. We add the mean vector $\bar{\mathbf{v}}$ from Equation (2) to $\hat{\mathbf{y}}$, to get $\hat{\mathbf{v}}$ that contains the skeleton points of the character in a new font style.

2.2. Finding the Template from the Manifold

Once we have learnt the font skeleton manifold for the target character, numerous font skeletons can be generated, from which we can find a most similar one as the template for stroke extraction. It is not possible to traverse everywhere in the manifold since the manifold space is continuous and infinite. However, it is observed that locations nearby tend to generate similar character skeletons and the skeleton changes continuously from one location to another. Let $\mathbf{z}$ denote the location that generates the most similar skeleton to the target character among the existing 28 fonts in the manifold, we speculate that the template character is likely to be located in the neighbourhood of $\mathbf{z}$.

We use the thinning directional feature [JG04] to measure the similarity of different fonts. Specifically, given the skeleton of a character, we divide the image into $8 \times 8 = 64$ meshes and then count the number of points in four directions (the horizontal direction, the vertical direction, the left falling direction and the right falling direction) in each mesh. Thus, we get feature vectors of $8 \times 8 \times 4 = 256$ dimensions for each character. By calculating the minimum Euclidean distance between the feature vectors of the target character and the existing 28 fonts, we get the value of $\mathbf{z}$. From Equation (6), we can generate the corresponding character skeleton when traversing around the location of $\mathbf{z}$. In our experiment, we traverse in circles centered in $\mathbf{z}$ and we increase its radius gradually until reaching the threshold τ. For each character skeleton we generate, the thinning directional feature is calculated in the same way as mentioned above and we choose the most similar one as the template. (See Figure 1(c))

2.3. Stroke Extraction

Given a target character in any font style, now we get the most similar template and afterwards, we apply the CPD [MS10] algorithm to implement non-rigid point set registration between the skeletons of the template and target characters. More specifically, let $X = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^{\mathrm{T}}$ be the point set of the template skeletons with N points, and $Y = [\mathbf{y}_1, \cdots, \mathbf{y}_M]^{\mathrm{T}}$ be the target point set with a total of M points on the skeleton. Here we can regard the vector X as the Gaussian Mixture Models (GMM) centroids, and registering the template point set X with target point set Y equals to determining the latent variables θ and the equal isotropic covariances of GMM distributions (e.g., σ^2) by minimizing the following function

$$E(\theta, \sigma^2) = -\sum_{i=1}^{M} \log(\sum_{j=1}^{N} \frac{1}{N} \frac{1}{2\pi\sigma^2} \exp - \frac{\|\mathbf{x}_j - \mathbf{y}_i\|^2}{2\sigma^2} + \frac{1}{N}). \quad (7)$$

Based on the non-rigid point set registration results, the skeleton of the character can be segmented into skeletons of the corresponding strokes. Afterwards, the contour points of the target character are assigned to the points on the stroke skeletons with the minimum Euclidean distance. At last, the target character is decomposed into several strokes accurately by completing and smoothing those segmented contours.

3. Experimental Results

In this section, we give the comparison of our method referencing the template generated from the font skeleton manifold with the method in [LZX16] referencing the standard Kaiti font library which was designed by Founder Group on the benchmark proposed in [CLTX16]. To make it fair and convincing, we evaluate the stroke extraction results on six font libraries including three printing styles

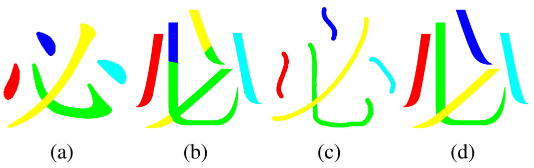

Figure 3: Comparison of the stroke extraction results for the Chinese character 'bi' in FZZQ font style. (a) The standard Kaiti font template; (b) Stroke extraction results referencing (a) ([LZX16]); (c) The template generated from the font skeleton manifold; (d) Stroke extraction results referencing (c).

(FZWB, FZTJLS, and FZZQ) and three handwriting styles (FZYBKS, FZJL and FZZZG). We randomly select 22 characters from the benchmark database for each font as the test dataset.

As shown in Figure 3, using the same point set registration algorithm while referencing different templates can bring great differences to the stroke extraction results. Figure 2.3 is the Chinese character 'bi' in the standard Kaiti font style, the structure of which is quite different from Figure 2.3, the character in FZZQ font style. Even excellent point set registration algorithms cannot find a proper correspondence between their skeleton points since both the location and the length of every stroke varies a lot from each other. The character shape in Figure 2.3 is terribly segmented except for the strokes in red and light blue. Figure 2.3 is the skeleton template found in the manifold which is obviously much more similar to the original FZZQ font style. Take the stroke in green color for instance, the start and the end position of the two strokes are nearly overlapped, which makes the point correspondence easier to establish. Figure 2.3 shows the exact stroke extraction results without error.

More experiments have been carried out on datasets of six font styles. Even for handwriting font styles such as FZYBKS, FZJL and FZZZG that usually have different writing skills and different structures from the standard Kaiti font library, our method still works well because we can always find a similar template in our manifold. Figure 4 shows more stroke extraction results of different characters in different font styles. It can be observed that most of the strokes have been accurately extracted.

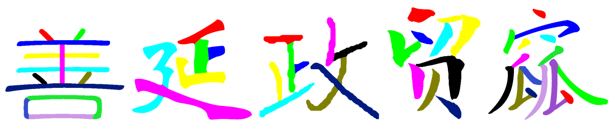

Figure 4: Stroke extraction results for Chinese characters in FZTJLS, FZWB, FZJL, FZZZG, FZYBKS font styles respectively.

Based on the benchmark proposed in [CLTX16], we give the metric of Hamming Distance error to quantitatively evaluate the stroke extraction methods with different templates. From Table 1 we can conclude that using a template from the manifold in a more similar style promotes the registration performance and therefore improves the precision of the stroke extraction methods.

	FZZQ	FZTJLS	FZWB	FZJL	FZZZG	FZYBKS
Kaiti	0.159	0.108	0.132	0.912	0.077	0.111
Mani	0.056	0.063	0.126	0.212	0.069	0.059

Table 1: Hamming Distance error of stroke extraction methods referencing Standard Kaiti templates and manifold generated templates respectively

4. Conclusion

This paper aims to solve the problem of extracting strokes from the perspective of giving more useful referencing data considering various styles of the target characters. Our solution is to construct a font skeleton manifold using a GP-LVM model to find a most similar template for a better registration result to the target character. Experimental results prove that our method works effectively and using our manifold significantly improves the precision of the data-driven stroke extraction methods. The manifold is flexible with accurate character matching and can be easily expanded by adding more fonts.

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China (Grant No.: 61472015, 61672056 and 61672043), Beijing Natural Science Foundation (Grant No.: 4152022) and National Language Committee of China (Grant No.: ZDI135-9).

References

[CK14] CAMPBELL N. D., KAUTZ J.: Learning a manifold of fonts. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 91. 2

[CLTX16] CHEN X., LIAN Z., TANG Y., XIAO J.: A benchmark for stroke extraction of chinese characters. *Acta Scientiarum Naturalium Universitatis Pekinensis* (2016). 2, 3, 4

[JG04] JIN L. W., GAO X.: Study of several handwritten chinese character directional feature extraction approaches. *Application Research of Computers 21*, 11 (2004), 38–40. 3

[Law05] LAWRENCE N.: Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research 6*, Nov (2005), 1783–1816. 2, 3

[LZX16] LIAN Z., ZHAO B., XIAO J.: Automatic generation of large-scale handwriting fonts via style learning. In *SIGGRAPH ASIA 2016 Technical Briefs* (2016), ACM, p. 12. 2, 3, 4

[MS10] MYRONENKO A., SONG X.: Point set registration: Coherent point drift. *IEEE PAMI 32*, 12 (2010), 2262–2275. 2, 3

[SQX14] SUN Y., QIAN H., XU Y.: A geometric approach to stroke extraction for the chinese calligraphy robot. In *ICRA 2014* (2014), IEEE, pp. 3207–3212. 2

[WLSL13] WANG X., LIANG X., SUN L., LIU M.: Triangular mesh based stroke segmentation for chinese calligraphy. In *ICDAR 2013* (2013), IEEE, pp. 1155–1159. 2

[WLTX13] WANG C., LIAN Z., TANG Y., XIAO J.: Automatic correspondence finding for chinese characters using graph matching. In *ICIG 2013* (2013), IEEE, pp. 545–550. 2

[YWY12] YU K., WU J., YUAN Z.: Stroke extraction for chinese calligraphy characters. *Journal of Computational Information Systems 8*, 6 (2012), 2493–2500. 2