考慮字間相連性之資料導向手寫字合成技術

林則如* 陳心怡* 沈奕超† 陳炳宇‡
*‡國立臺灣大學 †中央研究院
*{lintseju, fensi}@cmlab.csie.ntu.edu.tw †joshshen@citi.sinica.edu.tw
‡robin@ntu.edu.tw

ABSTRACT
一個人的英文筆跡一般都會在一定的範圍內變動，並且每個字母的手寫模樣與相鄰的字母彼此間也通常都有相當複雜的關係，而這種關係會使得手寫字以及手寫段落的自動生成非常具有挑戰性。在這篇論文中，我們提出一個根據書寫者的筆跡合成手寫字的方法。合成演算法包含兩個部分：第一，我們根據書寫者的筆跡樣本，對每一個字母建立一個多維度的可變模型。第二，我們根據同一筆跡樣本計算字母與字母間的相連機率，決定相鄰的字母間彼此是否要相連。因此，我們提出一個新的模型用來計算字母間的相連軌跡，並同時考慮了上述兩個部分。並且，這個模型亦會依據蒐集到的筆跡樣本合成單字。此外，段落的排版也會根據此筆跡樣本自動生成。這個方法所產生的結果可以成功地合成與書寫者提供的樣本具有相似筆跡的段落。

Categories and Subject Descriptors
I.3.3 [**Computer Graphics**]: Picture/Image Generation;; I.4.9 [**Image Processing and Computer Vision**]: Applications;

General Terms
Algorithms

1. INTRODUCTION
Though printed characters are easy to read, most people still like handwriting styles due to the uniqueness and aesthetics, so handwriting words are getting more and more popular in various design applications, from greeting card decoration to personal website stylization. As an alternative to standard printed characters, synthesizing handwriting-style content from collected database has attracted many attentions [2–5].

However, automatic synthesis of characters and paragraphs that mimic one's writing style is challenging. An individual's handwriting appears differently within a typical range of variations, and the shape of individual character also shows complex interaction with nearby neighbors. That is, the handwriting characters look different when they are written in a conjoined manner (see example in Figure 1). Such local trajectory variation between neighboring characters, as an important consideration in human writing, is less addressed in previous data analysis and handwriting synthesis approaches. In addition, the arrangement of the writing and the symbolism of margins, zones, spacing and slant must be taken into consideration in synthesizing paragraphs.

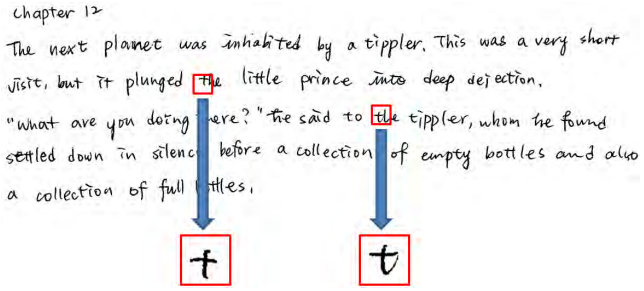
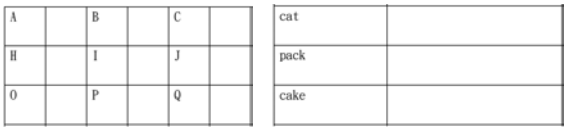


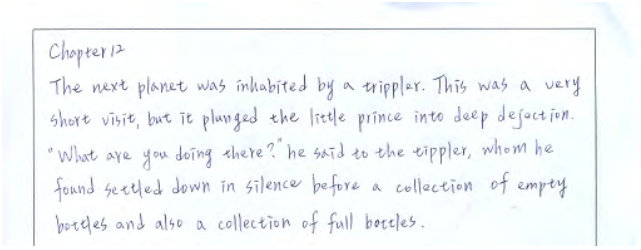
Figure 1: Two characters "t" in the same word "the". One conjoins with its neighbor "h", and the other does not. These two "t" look different, though they are both in the same word "the", and written by the same writer in the same article and very close time.

In this paper, we present a novel method for synthesizing handwriting text according to the writer's style while considering character's conjoined property. Our method is composed of two phases: The first phase is creating the morphable model of different characters, which enables us to synthesize various characters while facsimileing a person's writing style. In the second phase, a trajectory synthesis algorithm is used to produce smooth trajectories for specified words. The parameters of the trajectory synthesis algorithm are trained automatically from the collected samples using the coordinate descent method. Given the input ASCII words, the corresponding handwriting words are automatically synthesized by the proposed method. In addition, the global features, such as the angle of the handwritten text line with respect to the horizontal direction, word and line spacing, are also adjusted accordingly to reflect the writer patterns in the paragraph generating process.

Contribution. The proposed method utilizes a statistic learning approach to compactly model the distinctness of individual character as well as the conjoined properties in accordance with the collected dataset. With a small dataset provided by a writer, we are able to synthesize unexisted scripts that mimic a person's handwriting, both at the level of individual character and complete words. Specifically,



(a) Individual sample. (b) Integral sample - Word



(c) Integral sample - Paragraph

Figure 2: The sample document set. We asked users to write (a) individual characters including 26 English letters, in both lower and upper case, and 10 Arabic numbers, (b) individual words, and (c) a sample paragraph of an article.

the contributions of this paper include:

- A data analysis stage that offers an understanding of how characters interact with nearby neighbors, which reflects individual writing style.
- A character grouping method to guide the computation of cursive possibility for each character pair that decide whether adjacent characters are conjoining together or not.
- A novel data-driven optimization method which can be used to synthesize unexisted paragraphs from a small set of training data, where the distinctness of individual's writing as well as the natural character conjoined property are modeled jointly.

2. DATA COLLECTION

We collected handwriting samples by asking users to write a set of "sample documents". During the designing phase of this sample document set, we observed that: (1) different instances of the same character tend to be more consistent when they are written separately (i.e., the variation among them are smaller) comparing with those when they are written conjointly, and (2) the layouts of paragraphs, such as border width, line spacing or line parallelism, are different from person to person. Based on this observation, we have concluded the following criteria in designing the sample document set: (1) Each character must be appeared several times to capture its variation of handwriting style. (2) The sample paragraphs should cover as many as common character pairs as possible to capture the local deformations and trajectories of the conjoined character pairs. (3) While collecting integral samples, we cannot over-constrain the text placements to make personal style disappeared.

To support the above criteria, there are two different parts in the sample document set. In the first part, we asked users to write individual characters, including 26 English letters, in both lower and upper cases, and 10 Arabic numbers (Figure 2(a)). In order to capture the variation of writing styles, the users were asked to write the same content twice. In the second part, we target to collect writing samples with a conjoined manner, i.e., character pairs composed of at least two characters. Specifically, we asked users to first write the Chapter 12 of *The Little Prince* as the sample paragraph (Figure 2(c)). In addition, we also choose additional 15 words[1] to cover more character pairs (Figure 2(b)). (See Appendix for detilas.)

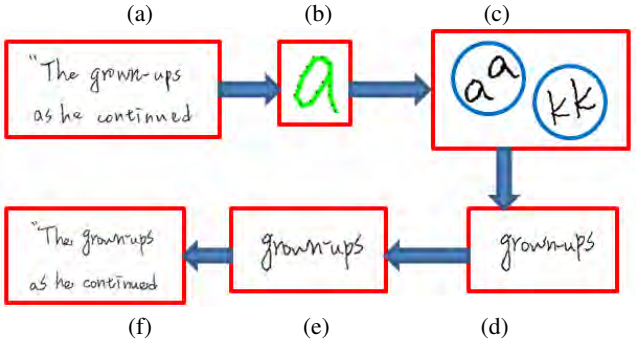


Figure 3: Work flow of our system. (a) Scanning input images. (b) Parameterizing each character. (c) Building the shape model for each character. (d) Synthesizing characters, (e) synthesizing words by optimizing smooth trajectory from synthesized characters, and (f) synthesizing paragraphs while preserving the writer's style.

3. HANDWRITING SYNTHESIS

3.1 Overview

The work flow of our method is shown in Figure 3. Given a handwritten source document from a user, we first scan (Figure 3(a)) and parameterize (Figure 3(b)) the characters in the document (Section 3.2) to obtain the control points representing the characters for further processing. Then, we build a statistical shape model from the control points (Figure 3(c)) to represent each character c_i using a set of shape parameters $\boldsymbol{\alpha}_i$ (Section 3.3). By investigating the collected data and grouping characters according to their structural similarities, we can also calculate the conjoined probability $P(i,j)$ for every pair of character combination (c_i, c_j) (Section 3.4). Given an input text, our synthesis algorithm (Section 3.5) performed as following: (a) For each individual character c_i, we synthesize similar but different character shapes $\hat{c}_i$ by varying corresponding multidimensional parameters $\boldsymbol{\alpha}_i$ in model space (Figure 3(d)). (b) If the current character c_i and its next character c_{i+1} are existed conjointly in the collected data, their conjoined property $P(i,j)$ is used to adjust the shape of $\hat{c}_i$ to connect or not to connect with $\hat{c}_{i+1}$. (c) However, if no conjoined information of pair (c_i, c_j) can be found in the collected data, we synthesize the smooth trajectories between $\hat{c}_i$ and $\hat{c}_{i+1}$ according to the stroke-based structural similarities to adjust the shape of $\hat{c}_i$. By formulating (b) and (c) as an energy minimization framework, we can synthesize novel characters and words (Figure 3(e)) with personalized cursiveness property. Finally, we imitate the paragraph layout (Figure 3(f)) extracted from the integral samples (Section 3.6).

3.2 Character Parametrization

After scanning the input images containing handwriting characters as shown in (Figure 3(a)), we first resample the scanned characters using uniform sampling with the distance of three pixels, and

[1]The 15 words are: cake, cars, cat, caveolae, dig, love, moon, no, pack, pick, poignant, rabbit, toss, troop, world.

adopt the sampled points as control points. We then represent each character $\mathbf{c}$ as:

$$\mathbf{c} = \{ p_1, p_2, ..., p_N \}. \quad (1)$$

where $p_x \in \mathbb{R}^2$ indicating the relative coordinate of p_x from the center, and N is the number of control points.

3.3 Shape Model Generation

Our method relies on the morphable models trained by the input handwritings, which is capable for morphing between a set of examples to synthesize previously unexisted shapes that are consistent with individual handwriting styles. The assumption of the shape model is that all the handwritten examples from a single user of each character (e.g. "a") lie in a low-dimensional space whose axes represent shape variation. For each character c_i, we take the collected samples $\mathbf{C}_i = \{\mathbf{c}_i^k\}_{k=1}^{S_i}$ as input for building the shape model, where S_i denotes the number of samples for character c_i. In addition, we arbitrarily select one example, say $\mathbf{c}_i^1$, as a reference, and compute displacement vectors $\mathbf{D}_i = \{\mathbf{d}_i^k\}_{k=1}^{S_i}$ by performing shape matching between other examples and the reference. We totally construct 62 shape models (i.e., 26×2 for lowercase and uppercase English letters, and 10 for Arabic numbers).

Shape matching. To match the reference example with others, we compute the displacement vector $\mathbf{d}_i^k$ for each pair $(\mathbf{c}_i^1, \mathbf{c}_i^k)$ as:

$$\mathbf{d}_i^k = \{d_i^k(p_1), d_i^k(p_2), ..., d_i^k(p_N)\} \quad (2)$$

where $d_i^k(p_x)$ denotes the displacement of control point p_x between $\mathbf{c}_i^1$ to $\mathbf{c}_i^k$. The displacement between the reference example and itself is zero (i.e., $\mathbf{d}_i^1 = 0$).

To compute correspondences between the control points on different examples, the shape context [1] descriptor is adopted, which composed of a log-polar histogram that measures edge points' locations and orientations to provide a $\theta \times r$ feature vector. In our experiment, we use 12 angular (θ) and 4 radial (r) bins. In addition, we supplement the shape context descriptor with the control points' 2D positions, which leads to a $12 \times 4 + 2 = 50$ dimensional shape descriptor.

Statistical shape model. The morphable shape model for each character $\mathbf{c}_i$ is created by performing principal component analysis (PCA). For each character, we assume each example is described by N control points, so aligning vectors form a distribution in the $2N$ dimensional shape space. The co-variance matrix of the shape space is,

$$Var(\mathbf{D}_i) = \frac{1}{S_i - 1} \sum_{k=1}^{S_i} (\mathbf{d}_i^k - \boldsymbol{\mu}_i)(\mathbf{d}_i^k - \boldsymbol{\mu}_i)^T, \quad (3)$$

where $\boldsymbol{\mu}_i$ is the vector of mean displacements of character c_i. The probabilistic model of PCA can be written as:

$$\Phi^T(\mathbf{d}_k^i - \boldsymbol{\mu}_i) = \begin{pmatrix} I_r \\ 0_{(2N-r)\times r} \end{pmatrix} \boldsymbol{\alpha}_i + \boldsymbol{\epsilon}, \quad (4)$$

where Φ^T is a $2N \times 2N$ matrix whose row vectors are the eigenvectors of $Var(\mathbf{D}_i)$, I is a $r \times r$ identity matrix, $\boldsymbol{\epsilon}$ is an isotropic noise in the PCA space, and $\boldsymbol{\alpha}_i$ is a r-dimensional vector and r is the number of modes to train the PCA model , where $\eta = diag(\lambda_1, ..., \lambda_r)$, and λ_j is the j-th eigenvalue.

(a) (b)

Figure 4: (a) One of character examples "a" in the dataset. (b) Instances synthesized using our shape model with different $\boldsymbol{\alpha}_i$ in Eq. (5).

Hence, we can rewrite Eq. (4) as:

$$\mathbf{d}_k^i = \boldsymbol{\mu}_i + \Phi_r \boldsymbol{\alpha}_i + \Phi \boldsymbol{\epsilon}, \quad (5)$$

where each element in $\boldsymbol{\alpha}_i$ reflects a specific variation along the corresponding principle component axis. By varying the elements in $\boldsymbol{\alpha}_i$ (see Figure 4), the generated shape parameter $\hat{\boldsymbol{\alpha}}_i$ enables us to synthesize new handwriting character $\hat{c}_i$ by

$$\hat{c}_i = \boldsymbol{\mu}_i + \Phi_r \hat{\boldsymbol{\alpha}}_i + c_i^1, \quad (6)$$

where $\boldsymbol{\alpha}_i$ is the coefficient of shape parameter plus a variant $\delta \sim N(0, \frac{\sqrt{\eta}}{3})$, and c_i^1 is the control points of the reference sample of character c_i.

3.4 Character Grouping and Cursive Probability

Given the collected samples and their shape models, we then group these examples according to their structural similarity. We subsequently use both the grouping results and collected samples to define a cursive probability for each character pair that decide whether adjacent characters are conjoining together or not.

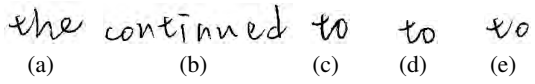
Character grouping from structural similarity. When people perform a flowing and uninterrupted movement, two adjacent characters have the property to conjoin together: the finishing position of the former character may link directly to the starting position of the later one. This implies that the starting and finishing positions are prominent factors in determining whether and how characters are linked. Based on this observation, we make an assumption: the character pairs which have similar starting-finishing positions may have similar and identical writing trajectories. This assumption is vital for the conjoining probability computation and trajectory shape synthesis (Section 3.5). For example, if a character pair (c_i, c_j) is absence in the collected data, we can find another character pair to synthesize similar writing trajectories with the same conjoining probability.

We group characters by their structural similarity, which is measured by the distance between their starting (and finishing) positions. Specifically, we divide the space on the writing sheet into six partitions (see Figure 5), say group 1 to group 6. Two characters c_i and c_j are clustered into the same starting/finishing group if their starting/finishing positions are located in the same partition (see Figure 5(a)/Figure 5(b)). We use $M_{ik}^s, k = 1, ..., 6$ to denote the starting group assignment of character c_i, and $M_{ik}^s = 1$ if c_i belongs to group k. In addition, we use $G_{ik}^s := \{c_i : M_{ik}^s = 1\}$ to denote all characters in starting group k clustered by their starting positions (M_{ik}^f and G_{ik}^f are defined in the same way for finishing groups).

Cursive probability. For each spatially adjacent character pair c_i and c_j, we use two different functions, P_{data} and P_{group}, to estimate its cursive probability $P(i, j)$. We define $P(i, j) = P_{data}(i, j)$

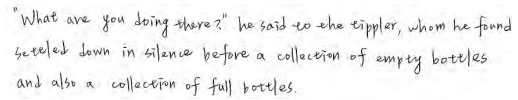
(a) (b)

Figure 5: We partition the writing sheet into 6 groups. (a) "h" and "l" are started from the same position G1, so they are in the same starting group. (b) "r" and "v" are finished at the same position G4, so they are in the same finishing group.

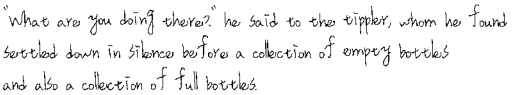


(a) (b) (c) (d) (e)

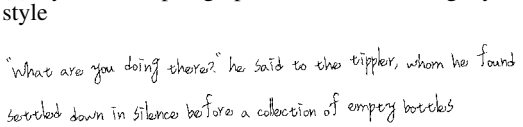
Figure 6: (a) and (b) The samples written by a user. (c) A word written by the same user. (d) The same word synthesized by our method while the trajectory between "t" and "o" can be generated smoothly. (e) The same word synthesized by directly pasting "t" from (a) and "o" from (b), so they cannot be connected naturally.



(a) Handwriting paragraph



(b) Synthesized paragraph without considering layout style



(c) Synthesized paragraph while considering layout style

Figure 7: A comparison of synthesized paragraphs with and without considering the handwriting layout style. In (b), the result looks like a typed paragraph using "handwriting" font. In (c), the result looks much nature compared with (b).

as the fraction of the character pair c_i and c_j if they existed conjointly in the collected data. Otherwise, we use $P(i,j) = P_{group}(G^f_{ik}, G^s_{jk'})$ to estimate its conjoining probability of character pairs in G^f_{ik} and $G^s_{jk'}$ in the collected data, which are the character sets in the same finishing and starting groups with c_i and c_j, respectively.

3.5 Word Synthesis with Trajectory

Given an input text, we firstly generate individual characters $\hat{c}_i$ using the morphable models in Eq. (6). However, when the characters are written in a flowing movement, they tend to conjoin together with their adjacent neighbors. This makes the characters look quite different from their regular shapes (when they are written separately). In order to mimic one's natural handwriting with cursive style, two concerns are evolved in this difficult task. Firstly, for an individual user, a probability model should be learned from the collected data to estimate the cursive probability when writing adjacent characters (Section 3.4). Secondly, the shape of the generated character $\hat{c}_i$ must be adjusted to be consistent with those in the collected samples in both appearance style and local conjoined trajectory.

Structure similarity constraint. To form a cursive writing given $P(i, i+1) > 0$, our goal is to adjust the shape parameter $\hat{\boldsymbol{\alpha}}_i$ to specify how to deform the shape model to synthesize the character $\hat{c}_i$ to connect with its neighbor $\hat{c}_{i+1}$ naturally. Our first requirement is that the shape parameter $\hat{\boldsymbol{\alpha}}_i$ should come from the character sample set $\mathbf{C}_i$ with similar conjoined style. Specifically, we would like to find the shape parameters $\boldsymbol{\beta}^k_i$ corresponding to the character samples $\mathbf{c}^k_i \in \mathbf{C}_i$, under the constraint that the pair $(\mathbf{c}^k_i, c_{i+1})$ existed conjointly in the collected data. At the same time, we would like the trajectory between the conjoined characters to be as smooth as possible, since unnecessary discontinuities in the path will result in unnatural handwriting. Hence, we attempt to achieve these objectives by minimizing the following function:

$$\hat{\boldsymbol{\alpha}}_i = \arg\min_{\boldsymbol{\alpha}_i, \boldsymbol{\beta}^k_i} \{\sum_i (\|\boldsymbol{\alpha}_i - \boldsymbol{\beta}^k_i\|_2 + \gamma(g^i_{p_N, p_{N-1}} - g^{i+1}_{p_1, p_2})^2\} \quad \text{s.t.} \quad p^i_N = p^{i+1}_1. \tag{7}$$

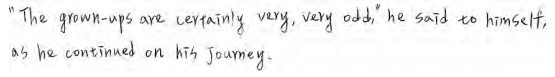
The first term is the data term, responsible for measuring the distance between the estimated shape parameters $\hat{\boldsymbol{\alpha}}_i$ and $\boldsymbol{\beta}^k_i$. However, if there is no such pair (c_i, c_{i+1}) in the collected data, we use the $\boldsymbol{\beta}^k_i$ from its structural similar group with c_{i+1}. The second term is the smoothness term, whose objective is to keep the slope of the trajectories between the finishing and starting positions of the conjoined characters as small as possible. The slope between control points P_j and $P_{j'}$ in character c_i is denoted as $g^i_{P_j, P_{j'}}$, and P^i_N denotes the N-th control point of character c_i. We also diminish the interval by constraining the finishing position of character c_i to be the same as the starting position of character c_{i+1}. The relative weight of the two terms is controlled by the parameter γ. In our current implementation, we use the default value $\gamma = 0.8$. The multidimensional non-linear optimization problem is solved by the coordinate descent algorithm, where the initial guess of the shape of character c_i is generated by Eq. (6). Figure 6(d) shows a synthesized result., which has smooth connection and preserve the style in the dataset.

3.6 Paragraph Synthesis

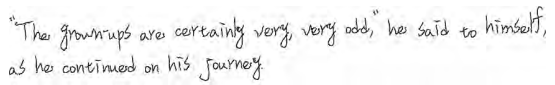
To composite a paragraph that is consistent with one's handwriting style, some important features, such as the heights and slopes of the characters, and the slopes of the entire lines, must be taken into consideration. It is because when people writing on a blank paper without any guide line or grid, it is not easy to write the characters with exactly correct positions, i.e., the words may slant heavily towards left or right. In our method, the relative height among characters and slant are preserved by our shape model. Words are rendered one by one, and the height and slope of each word is adjusted with two Gaussian distributions, $G_{height}(\mu_h, \sigma_h)$ and $G_{slope}(\mu_s, \sigma_s)$, respectively. To produce a smooth writing path, the parameters μ_h and μ_s are set as the height and slope of the preceding character, while σ_h and σ_s are set as 3 pixels and 2 degrees for all the results in our experiment. Meanwhile, the angle of each line with respect to the horizontal direction is also altered by a Gaussian function, $G_{line}(\mu_l, \sigma_l)$, where μ_l and σ_l are set as the mean and standard deviation of detected lines in the collected integral samples. In addition, our method can synthesize characters that faithfully reflect the size of the user-written characters. The result is shown in

(a) (b) (c) (d)

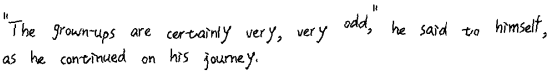
Figure 8: (a), (c), and (d) are collected samples in the dataset, while (b) is synthesized by the proposed method. Though there is no conjoined pair "r-i" in the dataset, we still can generate the trajectory by reffering the conjoining pairs "r-y" in (c) and "t-i" in (d) as described in Section 3.4.



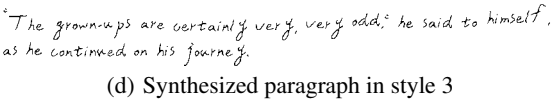
(a) Handwritten paragraph in style 1



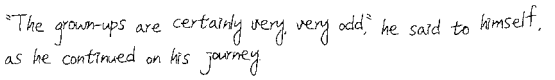
(b) Synthesized paragraph in style 1



(c) Synthesized paragraph in style 2



(d) Synthesized paragraph in style 3



(e) Synthesized paragraph in style 4

Figure 9: (a) A collected sample paragraph used as the training data. (b) The synthesized result using the trained model from (a), so their style is similar. (c), (d), and (e) are synthesized results using different datasets collected from other writers, so their styles are quite different.

Figure 7.

4. EXPERIMENTAL RESULT

4.1 Experiment Setting

We implement our method in MATLAB and all results were generated on a desktop PC with an Intel Core i7-3370 CPU and 16G RAM. Building shape model for our dataset described in Section 2 takes about 360 seconds while synthesizing a five-character words with conjoined manner takes about 1.2 seconds. To synthesize the Chapter 12 of *The Little Prince*, which contains 159 words (769 characters), takes about 190 seconds.

4.2 Visual Results

Figure 8 shows a conjoined result learned from existed pair in the collected dataset as well as an example leveraging the grouping method described in Section 3.4. Figure 8(b) is our synthesized result of the word "prince". For the "c-e" pair, it existed in the collected dataset, so we can generate its trajectory by learning from the existed samples. On the other hand, there is no "r-i" pair in the dataset. However, there are "r-y" and "t-i" pairs, where "i" and "y" are in the same starting group, and "r" and "t" are in the same finishing group.

The synthesized results of a small paragraph are shown in Figure 9. Figure 9(c) to Figure 9(e) are synthesized results of the same paragraph using different datasets collected from different writers. We can easily distinguish the different handwriting styles between them as well as the result shown in Figure 9(b). Meanwhile, the shape of characters and paragraph layout in the same style still remain consistent.

5. CONCLUSION

In this paper, we proposed a data-driven framework to synthesize unexisted paragraphs while preserving the writer's handwriting style. We first determine whether each pair of characters in the word are conjoined or not, by analyzing the cursiveness probability in the collected dataset. Furthermore, we learn the smooth conjoining trajectory from the same dataset, and formulate a novel trajectory optimization for synthesizing conjoining characters. Finally, we arrange the paragraph layout by imitating the writer's handwriting style. The result of user study shows that we can generate paragraphs that is hard to distinguish from the handwritten paragraphs.

Limitation and future work.. Currently, we assume that each character has only one topology, and cannot handle the characters with large deformation because of current character matching method. We believe a better matching method will help our method to handle more general dataset. In addition, our work does not consider pen pressure, which may affect the handwriting styles. In some cases, larger pen pressure makes the stroke color darker. We believe this is an interesting direction to obtain pen pressure from stroke colors and leverage it to generate better synthesis results.

6. 致謝

本論文感謝科技部經費補助，計畫編號：NSC101-2221-E-002-200-MY2。

7. REFERENCES

[1] S. Belongie and J. Malik. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[2] W.-D. Chang and J. Shin. A statistical handwriting model for style-preserving and variable character synthesis. *International Journal on Document Analysis and Recognition*, 15(1):1–19, 2012.

[3] Z. Lin and L. Wan. Style-preserving english handwriting synthesis. *Pattern Recognition*, 40(7):2097–2109, 2007.

[4] J. Lu, F. Yu, A. Finkelstein, and S. DiVerdi. HelpingHand: Example-based stroke stylization. *ACM Transactions on Graphics*, 31(4):46:1–46:10, 2012.

[5] Y. Xia, J. Wu, P. Gao, Y. Lin, and T. Mao. Ontology-based model for chinese calligraphy synthesis. *Computer Graphics Forum*, 32(7):11–20, 2013.

Appendix

English Character-Pair Analysis for Data Collection. Before performing the data collection as described in Section 2, we conducted an English character-pair analysis to design the writing task as shown in Figure 2(b) and Figure 2(c). To analyze the using frequency of English characters can be achieved via various approaches. In this paper, we analyze character distributions using the statistics provided by Project Gutenberg[2], which digitized the works of our predecessors. Most of the collected books are written in English and scanned after their copyright expired. Of all the scanned eBooks,

[2]http://www.gutenberg.org/

the $36,663$ most commonly used words are listed until 2006-04-16. To obtain the frequency list of character pairs, we perform a straight frequency counting on the commonly used word list and found 590 pairs of connected characters, where 159 mostly used pairs covered 90% of total pairs, 109 pairs covered 80%, and 78 pairs covered 70%.

To design the writing task, we search for a concise and complete paragraph among world literature masterpieces that can (1) cover as many as common character pairs as possible, and (2) be not too cumbersome for users to write. The Chapter 12 of *The Little Prince* was chosen because it contains 159 words in this paragraph, including 164 character pairs which covered 82% of the above total pairs. The additional 15 words can increase the coverage to be about 90%.