Use of Automatic Chinese Character Decomposition and Human Gestures for Chinese Calligraphy Robots

Fei Chao, *Member, IEEE,* Yuxuan Huang, Chih-Min Lin, *Fellow, IEEE,* Longzhi Yang, *Senior Member, IEEE,* Huosheng Hu, *Senior Member, IEEE,* and Changle Zhou

***Abstract*—Conventional Chinese calligraphy robots often suffer from the limited sizes of predefined font databases, which prevent the robots from writing new characters. This paper presents a robotic handwriting system to address such limitations, which extracts Chinese characters from textbooks and uses a robot's manipulator to write the characters in a different style. The key technologies of the proposed approach include the following: (1) automatically decomposing Chinese characters into strokes using Harris corner detection technology and (2) matching the decomposed strokes to robotic writing trajectories learned from human gestures. Briefly, the system first decomposes a given Chinese character into a set of strokes and obtains the stroke trajectory writing ability by following the gestures performed by a human demonstrator. Then, it applies a stroke classification method that recognizes the decomposed strokes as robotic writing trajectories. Finally, the robot arm is driven to follow the trajectories and thus write the Chinese character. Seven common Chinese characters have been used in an experiment for system validation and evaluation. The experimental results demonstrate the power of the proposed system, given that the robot successfully wrote all the testing characters in the given Chinese calligraphic style.**

***Index Terms*—Robotic calligraphy, human-robot interactions, Chinese character decomposition.**

I. Introduction

APPLICATIONS of robotics in promoting human culture and civilization, such as robotic writing and drawing, represent a major topic that has often been ignored by traditional robotics research. Robotic writing, a research field that appeals to many robotic scientists [1]–[4], focuses on how to design control algorithms to drive robotic end-effectors to write complex characters or letters [5], [6]. Handwriting, a typical human motion, is a highly demanding task that requires kinematics and dynamics, and complex characters, such as Chinese characters containing human-like handwriting, are very challenging for robots to draw [7]–[11]. Indeed, Chinese character writing is much more complicated than English writing because one Chinese character may consist of many strokes that must be placed in particular positions.

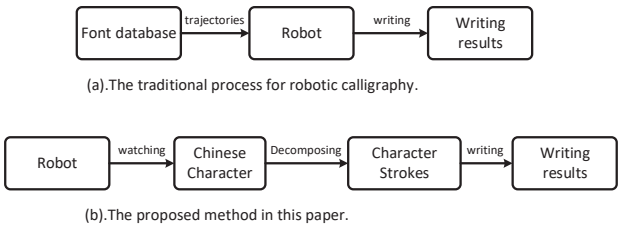


Fig. 1. (a) The main process of traditional calligraphy robots and (b) the proposed approach for the calligraphy robot.

Accordingly, a successful Chinese character-writing robot must be equipped with both stroke writing ability and space operation ability. A robot system capable of addressing such a challenging task can be widely applied in many other real-world exercises, such as industrial painting and assembly work. However, as shown in Fig. 1-(a), existing calligraphy robots can only write those characters included in predefined font databases. It is difficult, if not impossible, for the existing calligraphy robots to find a solution for writing a new Chinese character if the new character's font information is not embedded in the database. This observation leads to our belief that the robotic writing of Chinese characters faces the two major challenges detailed below.

The first major challenge is the requirement of extensive labor work in the construction of large font stroke trajectory databases with good coverage. A number of recent studies applied direct programming methods to a robot's control system with the support of a font database [12], [13]. Such a method requires complicated human work to convert font information into trajectories of manipulators [1]. In addition, this type of robot is only able to follow the font trajectories predefined in the font databases to write characters. Such robots work similar to printers or typewriters that repeatedly produce characters in exactly the same way as specified in the font libraries. Other scientists applied the follow-up ability of manipulators to obtain font information [14]. This method successfully imparts human calligraphic styles to robots, but it still entails much human work to enable manipulators to produce enough fonts.

The second major challenge is the high computational

F. Chao, Y. Huang, C.-M. Lin, H. Hu, and C. Zhou are with the Cognitive Science Department, School of Information Science and Engineering, Xiamen University, China e-mail: (fchao@xmu.edu.cn). L. Yang is with the Department of Computer and Information Sciences, Northumbria University, UK. C.-M. Lin is with the Department of Electrical Engineering, Yuan Ze University, Taiwan. H. Hu is with the School of Computer Science and Electronic Engineering, University of Essex, UK.

This work was supported by the National Natural Science Foundation of China (No. 61673322, 61673326, and 91746103), the Fundamental Research Funds for the Central Universities (No. 20720160126), the Natural Science Foundation of Fujian Province of China (No. 2017J01128 and 2017J01129), and the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 663830.

complexity in the establishment of stroke writing ability by character or stroke matching. Such an ability is usually built by adapting engineering methods in existing work. For instance, Garrido et al. [15] and Droniou et al. [16] applied a hidden Markov model and deep learning neural network to generate writing actions. Although very promising results were produced, these pieces of work involve an iterative training process or complex programming. Therefore, a more convenient and natural method is appealing for robotic stroke writing ability development.

To address the above challenges, this paper proposes an automatic Chinese character decomposition system that enables a robot to produce handwritten characters after "watching" the textbook version of Chinese characters. As shown in Fig. 1-(b), the robot system first captures a Chinese character from a regular Chinese calligraphy textbook and then automatically decomposes the captured character into a set of strokes. From this, each decomposed stroke is matched to a robotic stroke trajectory that is learned from the human demonstrator. After matching, the robot uses its own writing style to write the captured character stroke by stroke.

This method resolves the first challenge, as it does not require a font database to retain font information or complex programming to build the robotic writing ability. The automatic decomposition system can disassemble any Chinese character into a set of strokes, regardless of the previous experience in writing such a character. In terms of the second challenge, the proposed method does not require complex programming because it does not use human-robot interactions to establish the basic stroke writing ability. Learning via human gesture imitation generally reduces the high inherent complexity in the robot control algorithm [17]–[28]. In this work, the trajectories of human hand movements have two functions: (1) stroke shapes presentation and (2) decomposed stroke matching.

The main contributions of this work are summarized as follows:

1) An automatic stroke decomposition algorithm (detailed in Section II-C) is created to decompose a Chinese character into strokes to support the calligraphy robot.
2) A human-robot interaction method and a gesture classifier (in Section II-D) are established to enable robot learning in Chinese stroke writing.

The rest of this paper is organized as follows: Section II describes the proposed method via which the robot arm can write individual Chinese character automatically. Section III presents and discusses the experimental results. Section IV concludes the paper and discusses future research directions.

II. The Proposed Approach

A. Robotic System

The hardware setup for stroke learning is shown in Fig. 2-a. The hardware includes a motion-sensing input device, "Kinect," and a writing board. A human demonstrator stands in front of the Kinect within its detection range. The demonstrator performs the writing motions using their right arm, and the Kinect device captures the trajectories of the arm movement

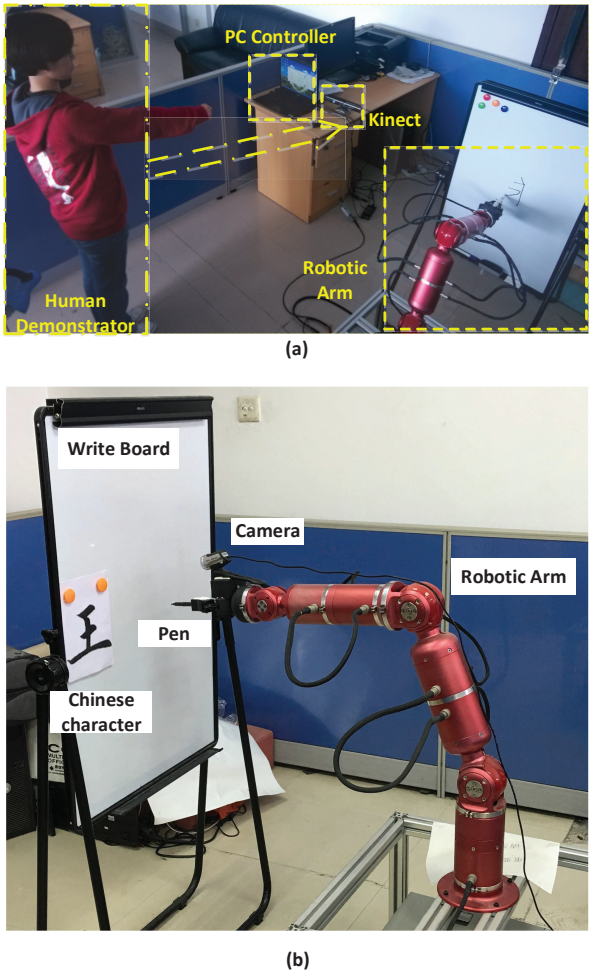


Fig. 2. The robotic writing system. (a) The hardware setup for stroke learning. (b) The robotic hardware setup for automatic writing.

to inform the robot. The trajectory recording method was developed in our previous work, as documented in Ref [11]. The stroke trajectory information is presented by the 2-dimensional trajectories of the human demonstrator's right hand. When the human performs stroke trajectories, only straight-arm gestures are accepted and processed to generate the trajectories, which are then converted into a set of robotic joint values.

The robotic hardware system is illustrated in Fig. 2-b, which includes a 5-DOF industrial robotic arm, a camera, and a writing board with Chinese character pictures. The relative positions of the robotic arm and the writing board are fixed. Four of the five arm joints are applied to perform a writing task. A soft pen is mounted at the tip of the arm, and the writing is placed within the arm's working range. The picture of an input character is placed in a fixed position on the writing board. The camera, mounted above the soft pen, captures the input Chinese character picture taken from Chinese calligraphic textbooks. When the input character is processed, the robot starts to write the character.

B. System Architecture

Fig. 3 shows the entire flow chart of the proposed approach, which contains the following four modules:

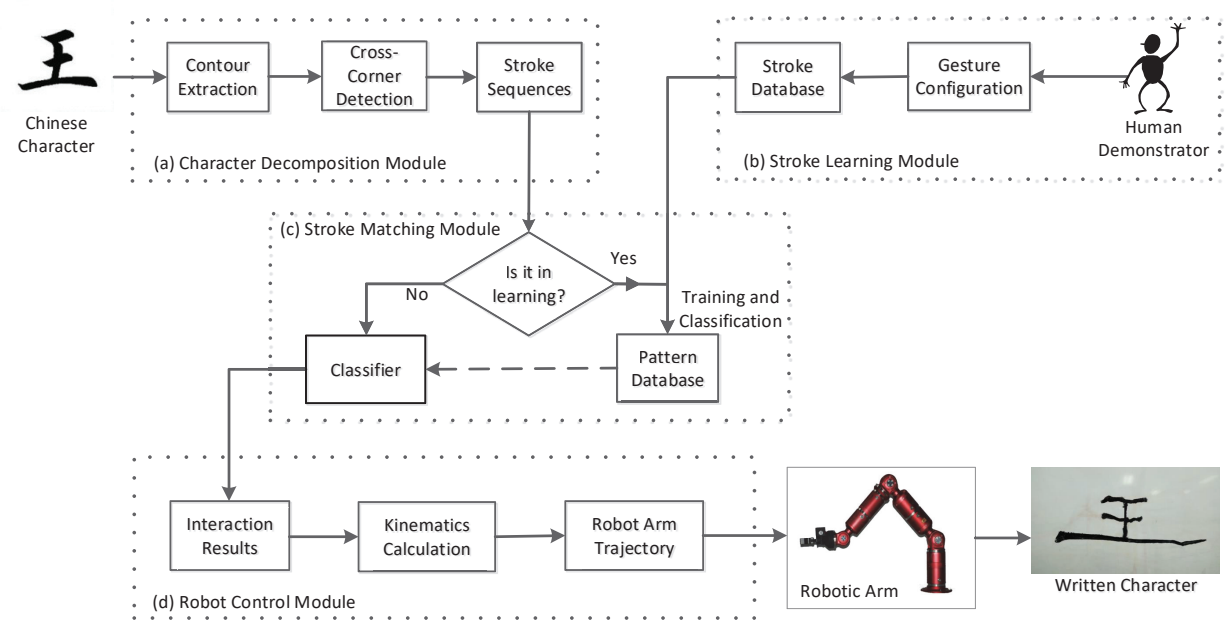


Fig. 3. The architecture of the proposed approach for robotic handwriting. The approach contains four modules: (a) character decomposition module, (b) stroke learning module, (c) stroke matching module, and (d) robotic control module.

1) **Character decomposition module**. The character decomposition module takes a single Chinese character image as input and produces a sequence of strokes by decomposing the character using a corner detection algorithm and a stroke decomposition algorithm [29], [30].
2) **Stroke learning module**. The stroke learning module establishes a robot stroke database. Each stroke in the database is generated by capturing the movement trajectories from the human demonstrator's right hand when the demonstrator performs the artificial writing in front of the Kinect device. Five primitive strokes are retained in the database, which can be used to assemble more complex character strokes through various combinations but cannot be decomposed any more.
3) **Stroke matching module**. The stroke matching is performed in two phases. In the first phase, the module requires labeled data to learn the stroke classification. Each labeled data instance consists of stroke trajectories and a stroke type. The stroke trajectories are generated by the character decomposition module or the stroke learning module. In order to rapidly implement the approach, a support vector machine (SVM) is employed as a classifier to learn the labeled data due to its high classification accuracy and wide availability. Specifically, a third-party version of SVM, "libsvm" [31], is used in this work. When the SVM's training is completed, the classifier is ready to perform classification tasks in the second phase. The module's input is the stroke trajectories generated from the character decomposition module, and the output is the stroke's trajectories selected from the robot stroke database.
4) **Robot control module**. The robot control module converts the stroke trajectories into the joint values of the manipulator. The module's input is the stroke trajectories resulting from the stroke matching module. Note that the stoke trajectory information is presented in a planar space due to the difficulty of capturing the pen pressure information. Therefore, a third coordinate with a fixed height value is introduced to enable calculation of the joint values of the robot arm. This step is implemented via an inverse kinematic approach, and the resulting joint values are used to drive the robots to write the character.

The technical details of the four modules are provided in the subsections below.

C. Character Decomposition Module

In Chinese calligraphy, character strokes are regarded as primitive units. Calligraphy practitioners write Chinese characters by painting strokes in a correct sequence in the right positions. Based on the observation that human calligraphy learners are able to correctly decompose Chinese characters into strokes, it is therefore crucial to establish a Chinese character decomposition method for the calligraphy robot. Yao et al. [32]–[34] first proposed 28 types of basic strokes for their calligraphy robot to write Chinese characters.

The stroke decomposition method reported herein is inspired by the work of Ao et al., Cao et al. [30] and Sun et al. [35]. In the work of Ao et al. [29] and Cao et al., candidate corner points of a Chinese character are first obtained based on the "Harris method", and then, strokes are decomposed using the filtered corner points. Their algorithm has been successfully applied to character decomposition of "Qi Gong" Calligraphy [30]. In contrast, Sun et al. [35] proposed a contour-based stroke decomposition algorithm by effectively using the corner point detection and stroke decomposition algorithms. The work of Sun et al. generally achieved better

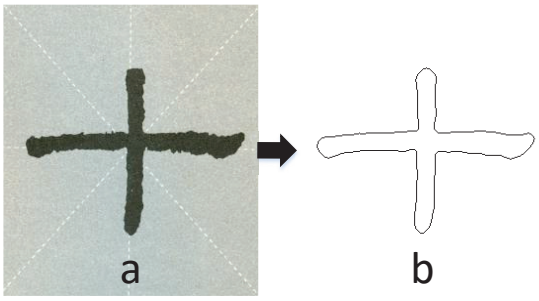


Fig. 4. An example of contour extraction of a Chinese character. (a) An individual Chinese character image captured by the robot's camera. (b) The binary image of the character's contour.

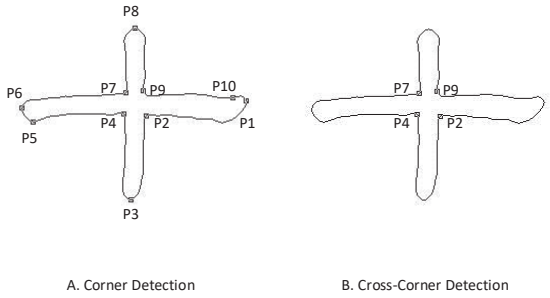


Fig. 5. The detection result of real cross-corners. The Chinese character "Ten" is applied as an example. (a) The detection result of the first process. The cross-corners are labeled $P1$ to $P10$. (b) The detection result of the second process. Only Points $P2$, $P4$, $P7$, and $P9$ are real cross-corners.

accuracy when decomposing Chinese characters and thus is adapted and further developed in this work. In particular, a stroke skeleton extraction function is introduced into the method of Sun et al. to convert a simple Chinese character into a number of writing trajectories that the robot arm can follow.

The character decomposition algorithm decomposes Chinese characters into a set of strokes to support robotic writing, and the algorithm used in this work consists of three steps: (1) contour extraction, (2) cross-corner detection, and (3) stroke decomposition. The input character of the proposed system is a certain type of Chinese regular script, called "Kai" style. The contour extraction step applies image processing technologies to generate the character contours. The cross-corner detection step detects the cross-corner points to find the real crossed positions between character strokes. If the overlapping strokes are simply segmented from the contours, the remaining strokes might be segmented into incomplete parts. Therefore, the stroke decomposition step must make up the missing parts to obtain complete strokes. The implementation of these three steps are described as follows:

1) Contour Extraction: In the contour extraction step, the robot's camera first captures images of individual Chinese characters. The size of each captured image is set to 160×160 pixels. A general image binarization process is first applied to reduce the contour extraction complexity. Then, an edge detection algorithm using eight directions extracts the character's contour [36]. From this, the extracted contour positions are retained in a contour matrix, which is executed using an array data structure. The sequence of the contour positions is clockwise. Fig. 4 shows the input (Picture a) and output (Picture b) of the contour extraction step using an example Chinese character.

2) Cross-corner Detection: Several ideas from the work of Ao et al. [29] and Cao et al. [30] are applied in the cross-corner detection step. The cross-corners are used to segment a Chinese character to a set of individual strokes. The key technology in this step is to determine cross-corners along character contours. However, it is difficult to correctly detect all the corners using a general corner detection algorithm, and some of the detected corners cannot be used to segment strokes. Therefore, a two-step cross-corner detection algorithm is proposed in this work:

(I) The first step finds all of the cross-corners using a general corner detection algorithm, including some false corners in the current research context. In particular, the Harris corner detection algorithm [37] is applied in this work. Then, the corner point positions are retained clockwise in a cross-corner candidate matrix ($\zeta_{candiate}$) for further processing in step 2. For the running example, ten cross-corners result, as shown in Fig. 5-a.

(II) The second step determines those cross-corners that are really needed for stroke decomposition. Regarding the Chinese character's composition and structure, the tangent lines of real cross-corners fall inside the contour [29]. Thus, how to obtain the cross-corners depends on the intersection point positions of the tangent lines. The following four-step algorithm is developed for the calculation of intersection point positions.

a) Obtain the tangent lines. For each corner, p_i, in the corner candidate matrix, $\zeta_{candiate}$, use p_i's previous and following points with a distance ι in the contour to generate p_i's tangent lines; here, $\iota = 5\ pixels$. Note that all "distances" in the paper refer to the two-dimensional Euclidean distance.

b) Compute the tangent line segment using the tangent line equations and a line length, δ, to define a line segment, $\overline{pk}$, in the tangent line; here, δ is also set to $5\ pixels$.

c) Calculate the end positions of $\overline{pk}$. Denote the end positions of $\overline{pk}$ as $k^i_{previous}$ and $k^i_{following}$. These end positions are used to determine if the tangent is inside the character's contour.

d) Determine the real cross-corners. If both $k^i_{previous}$ and $k^i_{following}$ are inside the character contour, p_i is determined to be a real cross-corner, and otherwise not. Then, store the position of p_i in a real cross-corner position matrix (ζ).

Fig. 5-b shows the processed result of the real cross-corners. Only Points $P2$, $P4$, $P7$, and $P9$ are determined to be real cross-corners; the remaining points are filtered out by the above algorithm.

3) Stroke Decomposition: When the real cross-corners of a character have been determined, they will be used by the stroke decomposition step to segment the strokes of the character.

There are two situations to be considered by the decomposition method. The first situation is that a cross-corner has only one adjacent corner. In this case, the decomposition method simply uses the line between the cross-corner and the only adjacent corner to segment the character.

In the second situation, a cross-corner has more than one adjacent corners, and the stroke decomposition step must identify which adjacent corner is the correct one. Taking Fig. 5-b as an example, if Point $P2$ is used as a starting cross-corner, Points $P4$, $P7$, and $P9$ are candidates of $P2$'s adjacent corner. Incorrect selection of the adjacent corner can lead to incorrect decomposed strokes. Therefore, the stroke decomposition method is created to determine which two cross-corners can be linked to each other.

The stroke decomposition algorithm works in 13 steps:

Step 1 Initialize the algorithm:
- ζ: Real cross-corner position matrix generated by the Cross-corner Detection step;
- χ: Adjacent corner candidates of current cross-corner;
- Ψ: Segmented stroke base;
- Ξ: Lookup table of connectable cross-corners;

Step 2 Set a starting point in each character's contour, which is the rightmost point in the contour. Then, travel the contour points clockwise and assign each cross-corner an index number starting from 0 and incremented by 1 for the next cross-corner.

Step 3 For each cross-corner, cp_j, in the cross-corner matrix, ζ, define a distance threshold to select cp_j's candidates of adjacent corner. The threshold is set to 25 $pixels$. The candidates are retained in χ.

Step 4 Check the size of χ through the following steps:
- Case 1: If the size equals 0, sequentially retain the remaining contours in Ψ. Then, the procedure terminates.
- Case 2: If the size of χ is greater than 4, segment the character by linking the cross-corner with the smallest index number and the corner with the largest index number. Retain the segmented contour in the stroke base Ψ. Remove the cross-corner point with the largest index number from χ and ζ. Return to Step 2.
- Case 3: If the size of χ equals 2, segment the character by linking the only two cross-corners, and retain the segmented contour in the stroke base Ψ. Remove the two cross-corners from χ and ζ. Return to Step 2.

Step 5 Otherwise, the size of χ is 4. Defines cp_j's previous and following contour points, $cp_j^{previous}$ and $cp_j^{following}$. The distances from $cp_{previous}^j$ to cp_j are empirically set to 5 $pixels$.

Step 6 Define two lines, $L_j^{previous}$ and $L_j^{following}$. The starting and end points of $L_j^{previous}$ are $cp_j^{previous}$ and cp_j, and those of $L_j^{following}$ are cp_j and $cp_j^{following}$. Then, calculate the slope values $\alpha_j^{previous}$ and $\alpha_j^{following}$ of the two lines. Each slope value is calculated as follows:

$$sl = \frac{|y_1 - y_2|}{|x_1 - x_2|}, \tag{1}$$

where (x_1, y_1) and (x_2, y_2) denote the two-dimensional positions of the two points defining a line.

Step 7 For each of the remaining cross-corners in χ, define a line, L_j^l, by linking cp_j to the lth cross-corner. l denotes the index number of the remaining corners in χ. Calculate the slope value κ_j^l of each L_j^l.

Step 8 Find a cross-corner, $cp_j^{candiate}$, in χ whose slop value, κ_j^l, is the closest to $\alpha_j^{previous}$ or $\alpha_j^{previous}$ and regard this corner, $cp_j^{candiate}$, as the connectable adjacent corner. Then, include $cp_j^{closest}$ and cp_j as a cross-corner pair in the cross-corner lookup table, Ξ, as indicated in Table I.

Step 9 Check the remaining elements in χ. If no element exists, save the lookup table, Ξ, for the next step. Otherwise, return to Step 6 to start a new iteration for the next cross-corner.

TABLE I
THE CROSS-CORNER PAIR LOOKUP TABLE Ξ.

Line	Connectable Point	Line	Connectable Point
$L_{P2}^{previous}$	P_4	$L_{P2}^{following}$	P_9
$L_{P4}^{previous}$	P_7	$L_{P4}^{following}$	P_2
$L_{P7}^{previous}$	P_9	$L_{P10}^{following}$	P_4
$L_{P9}^{previous}$	P_2	$L_9^{following}$	P_7

Step 10 Travel the contour points from the rightmost contour point. For the first cross-corner while traveling, find the line defined by its previous point, and directly link its connectable point according to Table I. Then, remove the cross-corner from Table I.

Step 11 Check whether the current traveling point is the starting point. If so, move to the next step; otherwise, return to Step 10.

Step 12 Retain the traveled contour points generated in Step 10 in the stroke base Ψ. Remove the traversed points from the character contour matrix ζ. However, this removal process breaks the overlapping strokes. Fig. 6 shows a typical incomplete segment result after the removal processes. In this figure, Stroke 1 is a complete stroke that is retained in the stroke base; however, Stroke 2 has a gap in the middle part.

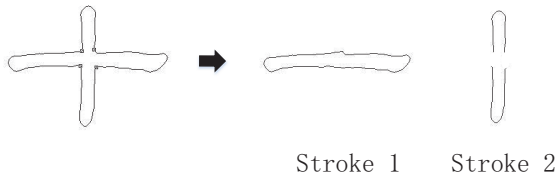


Fig. 6. The incomplete stroke segmentary result.

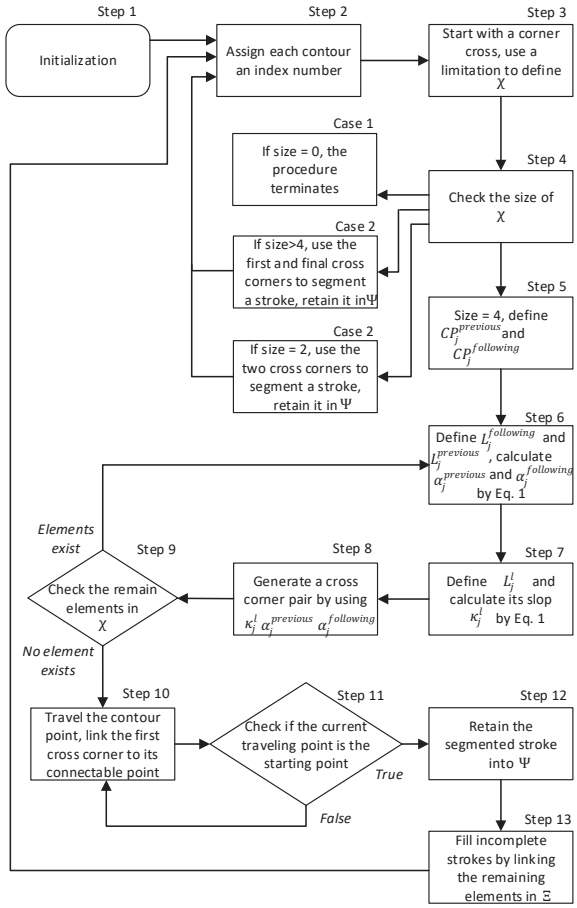


Fig. 7. The flowchart of the stroke decomposition procedure.

Step 13 Check the remaining elements in Table I and link each line's starting pointing to its connectable point. Then, return to Step 2 to start a new iteration.

In order to clearly show the flowchart, the entire procedure is summarized in Fig. 7.

4) Stroke Skeleton Extraction: The output of the stroke decomposition is stroke contours, but the calligraphy robot system cannot use the contours to write. Instead, the robot requires the skeleton trajectories of the stroke contours. Therefore, a contour filling and a skeleton distilling processes are involved to generate the skeletons of character strokes. One of the classical contour filling algorithms, "Flood Fill", in image processing technologies [38] is applied in this work to implement the stroke filling processes.

The skeleton extraction method proposed by Hou et al. successfully converts individual Chinese characters into skeleton trajectories [39]. It is generally simpler to perform stroke skeleton extraction than to perform the entire character skeleton extraction. Therefore, the extraction method proposed by Hou et al. is adopted in this work. The scanning sequence of stroke skeleton extraction must be predefined. First, the algorithm calculates the height-to-width ratio ρ of each character. If a stroke's $\rho \leq 10$, the scanning sequence is from left to right; otherwise, the sequence is from top to bottom.

D. Stroke Learning Module

The stroke learning module is responsible for the learning of the stroke writing skills for the robot. This module is developed based on previous work regarding learning robotic writing ability though human gesture analysis [22]. In order to perform Chinese character writing, five emblematic command gestures are chosen to represent five primitive/elementary Chinese strokes. The five gestures are as follows:

1) **Horizontal stroke.** Raise the forearm to approximately shoulder height; then, perform a horizontal waving motion.
2) **Vertical stroke.** Raise the arm to head height; then, move the arm down vertically.
3) **Left falling down stroke.** Raise the forearm towards head height; then, push the hand downwards to the left side of the body.
4) **Right falling down stroke.** Raise the forearm towards head height; then, push the hand downwards to the right side of the body.
5) **Point stroke.** This gesture is similar to the horizontal and right falling down strokes, but it is much shorter.

Since the stroke skeleton extraction algorithm cannot extract the pen pressure information from the stroke image, the stroke trajectory is only described as a planar motion. However, a third coordinate is required by the robot arm; therefore, a fixed height value is sent to the robot during the stroke writing stage.

Each stroke in this work is represented by fifteen equidistant position points. To build a robot stroke database, the robot requires the starting point position (x_{start}, y_{start}) of each stroke. Each of the remaining fourteen points are referenced back to its previous point. In other words, except for the first position, the remaining positions are relative values $(\Delta x_n, \Delta y_n)$ between the current position and the previous one. Therefore, one stroke's data structure can be expressed as follows:

$$T_s^{robot} : [(x_{start}, y_{start}), (\Delta x_1, \Delta y_1), (\Delta x_2, \Delta y_2), \cdots, (\Delta x_{14}, \Delta y_{14})], \quad (2)$$

where T_s^{robot} is the stroke type used as the search index.

Chinese character writing has a unique feature: one Chinese stroke has many variations. Usually, high-quality Chinese writing cannot be achieved simply by using only monotonous stroke shapes. Therefore, the five elementary strokes are designed to possess a scale function to construct various shapes of one type of stroke. The input of the database must include the starting position (x_{start}, y_{start}) and the ending position (x_{end}, y_{end}) for each stroke. Therefore, the stroke data structure in Equation 2 is extended as follows:

$$T_s^{robot} : [(x_{start}, y_{start}), (\alpha\Delta x_1, \beta\Delta y_1), (\alpha\Delta x_2, \beta\Delta y_2), \cdots, (\alpha\Delta x_{14}, \beta\Delta y_{14})], \quad (3)$$

where α and β are scaling parameters of each point, which are in turn defined by

$$\begin{cases} \alpha = \dfrac{\sqrt{(x'_{start})^2 - (x'_{end})^2}}{\sqrt{(x_{start})^2 - (x_{end})^2}} \\ \beta = \dfrac{\sqrt{(y'_{start})^2 - (y'_{end})^2}}{\sqrt{(y_{start})^2 - (y_{end})^2}}, \end{cases} \tag{4}$$

where x'_{start}, y'_{start} and x'_{end}, y'_{end} are the starting and ending positions of the new input stroke. As the human demonstrator uses different stroke's starting and ending positions to control a stroke's shape, the robot learns different types of strokes. The output of this module is the robot stroke database.

E. Stroke Matching Module

This module works in two modes: (1) the training mode and (2) the performance mode. In the training mode, the decomposed strokes are labeled by stroke types and used as a training sample for the stroke classifier. In the performance mode, a set of strokes is sent to the classifier as input, and then the classifier generates the type labels for the strokes.

Because the classifier cannot directly recognize the decomposed stroke contours, it is necessary to convert the contours into skeletonizing trajectories. Thus, using the stroke skeleton extraction module in Section II-C4, fifteen equidistant position points are selected from the skeleton trajectories. After the conversions, the stroke's trajectories can be expressed as follows:

$$\begin{aligned} T_s^{input} : [(X_{start}, Y_{start}), (\Delta X_1, \Delta Y_1), (\Delta X_2, \Delta Y_2), \cdots, \\ (\Delta X_{14}, \Delta Y_{14})], \end{aligned} \tag{5}$$

where T_s^{input} is the input stroke type used as the search index and (X_{start}, Y_{start}) is the starting position of the input stroke.

In the training mode, the decomposed strokes generated from Equation 5 and the learned strokes in the robot's database are used as the training samples. The decomposed strokes are labeled manually. A varied distribution of SVM, named "C-support vector classification (C-SVC)", is employed as the classifier herein [31]. The kernel function of the employed SVM is the radial basis function (RBF). Two key parameters, $\mathcal{C}$ and γ, are set to 1 and 0.04, respectively. The grid search [31] implemented by "libsvm" generates the best combinations of the two key parameters.

When the training phase is completed, the SVM is ready to perform the classification tasks. In the performance mode, the input of this module is the stroke contours, and the output is the type of the stroke in reference to the stroke types in the robotic stroke database. Then, the robot searches the trajectories of the identified stroke type to perform the writing, as detailed in the next subsection.

F. Robot Control Module

The robot control module takes the classified results to obtain the writing trajectory position through inverse kinematics calculations and drives the manipulator to finally write the characters. Therefore, this module includes stroke trajectory conversion and kinematics calculations, which are described as follows:

1) Stroke Trajectory Conversion: This step converts the stroke contours to the the trajectories of the robot manipulator. Two pieces of information are required in performing the conversion: (1) the relative position of each stroke of the Chinese character that is detected by the robot and (2) the stroke trajectories of the robot that are obtained from the robot stroke database.

Thus, when the detected character has been decomposed into a set of strokes, the starting and ending positions of each decomposed stroke must also be detected. Combining the starting and ending positions and the robot's stroke trajectories, Equation 5 can thus be reexpressed as

$$\begin{cases} \Delta X_i = \dfrac{\Delta x_i \cdot (X_{end} - X_{start})}{\sum_{j=1}^{14} \Delta x_j} \cdot \gamma \\ \Delta Y_i = \dfrac{\Delta y_i \cdot (Y_{end} - Y_{start})}{\sum_{j=1}^{14} \Delta y_j} \cdot \varphi, \end{cases} , i = 1, \cdots, 14 \tag{6}$$

where X_{start}, Y_{start} and X_{end}, Y_{end} denote the starting and ending positions of the stroke, respectively.

The type of the character provided on the writing board and detected by the robot is the "Kai" style, but the robot aims to write in another style, and the "Li" style is specifically used in this work for demonstration. The most significant difference between these two styles is that the width of the "Li" style is much wider than that of the "Kai" style. Therefore, γ and φ are the scaling factors for changing each stroke's shape, and the stroke conversion method is defined as follows:

$$\begin{aligned} T_s^{robot} : [(X_{start}, Y_{start}), \quad (X_{start} + \Delta X_1, Y_{start} + \Delta Y_1), \\ (X_{start} + \Delta X_1 + \Delta X_2, Y_{start} + \Delta Y_1 + \Delta Y_2), \\ \cdots, \quad (X_{start} + \sum_{i=1}^{14} \Delta X_i, Y_{start} + \sum_{i=1}^{14} \Delta Y_i)]. \end{aligned} \tag{7}$$

2) Kinematic Calculation: As only the joint angle values control the electrical motor of the robot arm, the kinematic calculation step must convert the stroke trajectories into the robot's joint values.

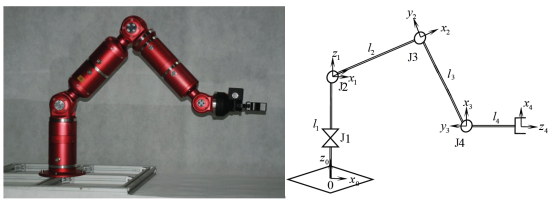


Fig. 8. The configuration of the robot arm.

The configuration of the robotic arm is illustrated in Fig. 8, which includes each joint's coordinate frame and the setup

of the arm's joints and links. The robotic arm has four linked parts, l_1, l_2, l_3, and l_4, with lengths of 150 mm, 375 mm, 354 mm, and 175 mm, respectively. The robotic arm's origin coordinate frame is based on the first joint. In this setup, the x_0 axis is vertical relative to the writing board; the z_0 axis is vertical relative to the ground; and the y_0 is vertical relative to the plane that is defined by the axes x_0 and z_0.

To conveniently describe the position of each joint and control the robotic arm, the Denavit and Hartenberg (D-H) convention is used to analyze the forward and inverse kinematics of the robot's manipulator. The D-H parameters are listed in Table II.

TABLE II
D-H PARAMETER TABLE.

Joint No.	θ	d	a	α	Range
1	θ_1	0	150	-90	[-120°,120°]
2	θ_2	0	375	0	[-90°,90°]
3	θ_3	0	354	0	[-90°,90°]
4	θ_4	0	175	0	[-45°,45°]

In this table, θ_i denotes the rotation angle from the x_{i-1} to the x_i axis along the z_{i-1} axis; d represents the distance from the origin of the *(i-1)* coordinate system to the intersection of the z_{i-1} axis and the x_i axis about the z_{i-1} axis; a indicates the distance from the intersection of the z_{i-1} axis and the x_i axis to the origin of the ith coordinate system about the x_i axis; and α is the rotation angle from the z_{i-1} axis to the z_i axis along the x_i axis.

The inverse kinematics analysis of the robot arm is obtained from the forward kinematics. Thus, if the positions of the manipulator are obtained, the four joints of the manipulator are calculated using the following equations:

$$\theta_1 = arctan\frac{p_x}{p_y}, \tag{8}$$

$$\theta_2 = arcsin- \frac{a_2 p_z + a_3 cos\theta_3 p_z + a_3 sin\theta_3 \sqrt{a_2^2 + a_3^2 + 2a_2 a_3 cos\theta_3 - p_z^2}}{a_2^2 + a_3^2 + 2a_2 a_3 cos\theta_3}, \tag{9}$$

$$\theta_3 = arccos\frac{(p_x cos\theta_1 + p_y sin\theta_1 - a_4)^2 + p_z^2 - a_2^2 - a_3^2}{2a_2 a_3}, \tag{10}$$

$$\theta_4 = -\theta_2 - \theta_3. \tag{11}$$

3) Software Architecture: The software implementation is illustrated in Fig. 9. The implementation consists of two components: (1) an algorithm computer and (2) a hardware controller. The algorithm computer handles high-level programs, such as the stroke learning, image processing, Chinese character decomposition, stroke matching and classification, and inverse kinematics calculations. The hardware controller receives joint values from the algorithm computer and converts the commands into the values that can be accepted by the motors of the robotic arm.

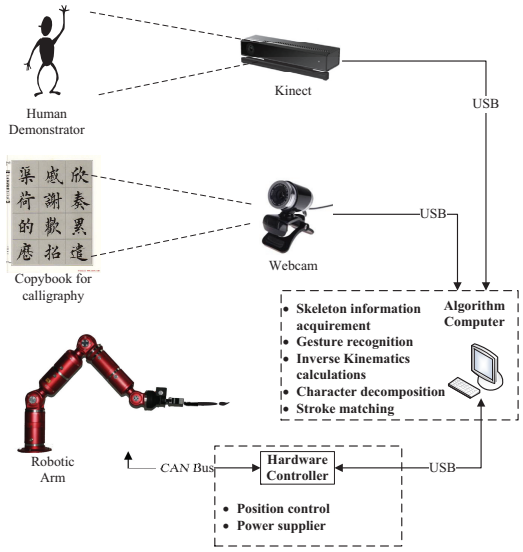


Fig. 9. The software implementation of the proposed approach.

The configuration of the algorithm computer is set as follows: the CPU and the operating system of the development computer are Intel Core i5-4200U CPU@2.30 GHz and Windows 7 Professional. The implementation language of the software is "C++" in the environment of "Microsoft Visual Studio" and the "Kinect SDK 1.5". In addition, the manipulator used in the experiment is a 5-DOF industrial robotic arm designed by the Biomimetic and Intelligent Robotics Lab, Guangdong University of Technology, China.

III. EXPERIMENTATION

As shown in Fig. 3, the proposed approach consists of four modules; therefore, the experiment is also divided into four parts: (1) character decomposition, (2) stroke learning, (3) classifier training and classification, and (4) robot writing.

The decomposed strokes of each type (generated in Part 1) are also used for training the SVM classifier in Part 3. In particular, seven simple and common Chinese characters are selected for system testing. The obtained strokes are used to match the robot strokes included in the stroke database. The second part, i.e., the stroke learning, was implemented based on our previous work; therefore, the details of this part are omitted from this paper for concise presentation. Finally, in Part 4, the results of the seven written Chinese characters demonstrate the effectiveness of the proposed method.

A. Character Decomposition

The entire cross-corner detection process is shown in Fig. 10. Take the Chinese character "Big" as an example. Fig. 10-(a) shows the original character image that the robot read. Fig. 10-(b) shows the character's contour obtained by using the contour extraction algorithm. Fig. 10-(c) illustrates all the potential corners obtained by the Harris corner detection algorithm. Eleven corners, highlighted by black points, are

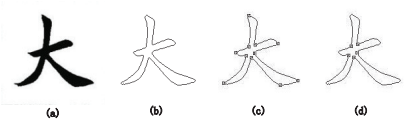


Fig. 10. The entire process of the cross-corner detection.

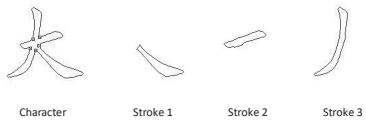


Fig. 11. The stroke contour sequence of the character "Big".

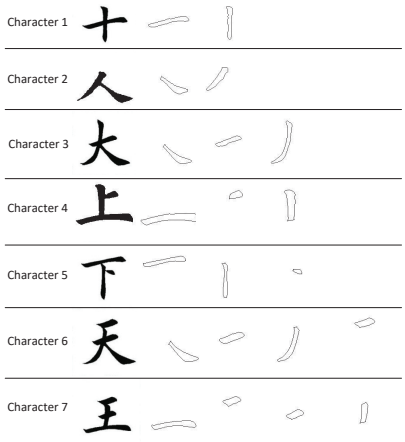


Fig. 12. The decomposed results of the seven characters selected.

depicted in this figure; however, only a subset of these corners are genuine cross-corners. Fig. 10-(d) presents the five correct cross-corners obtained by the proposed algorithm. This figure shows that our proposed method successfully found all the correct cross-corners, which were used for character decomposition. Fig 11 shows the four decomposed strokes obtained using the stroke decomposition algorithm; it is clear that none of the decomposed strokes is broken.

To further investigate the effectiveness of the character decomposition module, seven other Chinese characters were selected to test the module. Fig. 12 shows the decomposed results of the seven characters. All the strokes were successfully generated. Additionally, the algorithm guarantees the completion of each stroke. However, our character decomposition module cannot treat Chinese characters that contain more than five cross-corners. If so, the module may mistakenly segment a correct stroke into two parts. In addition, the module is sensitive to digital noise. If the boundary shapes of a character are not smooth, the performance of the character decomposition module will be compromised. More investigation is required to address such difficult situations.

B. Stroke Training and Classification

This work applies five types of strokes to write Chinese characters. In the stroke training module, each type of stroke consists of twenty differently shaped samples; therefore, the entire training database contains 100 samples. These training samples were used to train the SVM. In addition, ten additional samples for each stroke type were used for testing; thus, the testing database contains 50 instances. After testing, the classification accuracy was 98%. The amount of training samples, testing samples and the class labels are summarized in Table III.

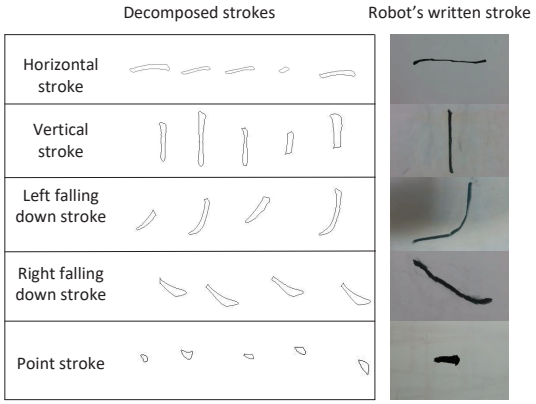


Fig. 13. The recognize results of the strokes.

TABLE III
DETAILS OF THE TRAINING AND TESTING DATASETS.

Stroke Type:	**Training Sample Amount:**	**Testing Sample Amount:**	**Class Label (T_s):**
Horizontal stroke	20	10	1
Vertical stroke	20	10	2
Left falling down stroke	20	10	3
Right falling down stroke	20	10	4
Point stroke	20	10	5

In the performance mode, the SVM must recognize each input stroke contour to generate its corresponding stroke type and then use the stroke type to search the robot stroke database to find the corresponding manipulator trajectories. Fig. 13 shows the recognition results of the SVM. The contours in the left column are various decomposed strokes obtained from the stroke decomposition module. The trajectories in the right column are the corresponding robot trajectories.

C. Robot Writing

The writing results for the seven Chinese characters are shown in Fig. 14. The left column shows the "Kai" style characters detected by the robot's camera; the right column shows the writing results of the robot manipulator. In particular, the writing style in the rightmost column is totally different from that of the input characters. This style difference reflects the important contribution of this work, that the robot uses its own writing style rather than redrawing the identical shapes from the input characters. Fig. 14 also shows that the robot successfully wrote all the characters using its learned writing skills.

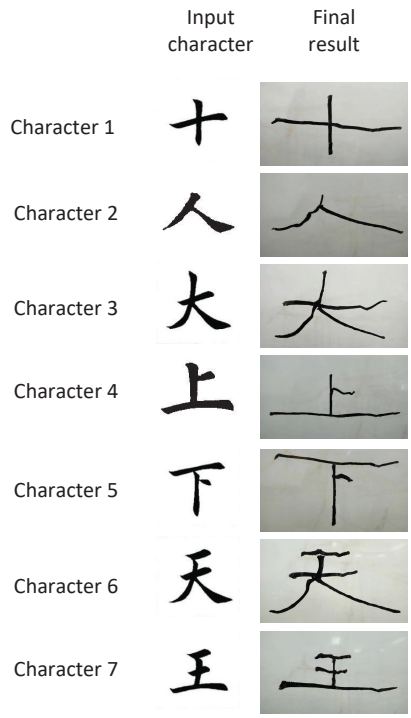


Fig. 14. The writing process and results of the selected seven characters.

Fig. 15. The recognize result of the strokes.

TABLE IV
SUMMARY OF THE COMPARISON WITH THE CONVENTIONAL APPROACHES

Options:	**Conventional approaches:**	**Our approach:**
Font database implementation methods:	almost all existing research applied predefined font databases, which were created by computational programming [12], [13] or by follow-up mechanisms [14], [22];	no font database is implemented in this work; alternatively, our robot system contains a stroke database and merely requires to receive font images from Chinese calligraphic copy books.
Support to write new Chinese characters:	few current research contains this feature, current calligraphic robots can the characters that are embedded in databases;	by using the character decomposition algorithm, our robot is able to treat new Chinese characters with simple structures.
Support to write various font styles:	current research depends on the font databases, several robots can write different font styles, which must be embedded in the font databases; however, it is difficult to add new styles to their font database;	our approach is able to treat new writing styles by following human demonstrator's gestures that perform different stroke styles.
Human-robot interactions	only a small number of models exhibit this feature [40];	our approach applies human-robot interactions to establish the elementary stroke database.

To further demonstrate the advantages of the proposed approach, various font types are introduced to the calligraphic robot by following different human demonstrators. Fig. 15 shows three different stroke styles of Character 7 shown in Fig. 14; the stroke types of Styles 2 and 3 are generated from two other human demonstrators. For each stroke style, the two demonstrators must use different gesture trajectories to perform the writing. Although the input character was not changed, the shapes of the writing results were significantly different, which clearly demonstrates the ability of the proposed system in supporting various font styles.

D. Discussion and Comparison

Based on the above experiments, we believe that the proposed human-robot approach is successful in controlling the robot to write Chinese characters. This work significantly differs from the existing pieces in that the proposed calligraphy robot uses a Chinese character decomposition method to convert a Chinese character into a series of elementary strokes. The robot obtains the stroke writing ability through human-robot interactions; in addition, the stroke matching can build the relationship between the recognized strokes and robot's written strokes. To further reflect the strengths of this research, a comparison with conventional calligraphic robot approaches is presented in Table IV. The comparison particularly focuses on the following four important features: 1) font database implementation methods, 2) ability to write new characters, 3) ability to write various font styles, and 4) human-robot interactions.

Many conventional approaches prefer to apply direct computational programming or a follow-up mechanism to create embedded font databases. However, this work does not require such a font database; alternatively, the proposed robot system contains a stroke database and only requires to receive font images from Chinese calligraphic text books. Therefore, our approach can greatly reduce the amount of work in font database implementations. Additionally, because of the capacity limitation of font database, few existing systems support writing new Chinese characters that are not embedded in their font databases. In contrast, our robot is able to address new Chinese characters with simple structures by using the character decomposition algorithm.

Only a few conventional approaches tend to focus on building various font styles of robotic writing. However, it is still very difficult to add new writing styles to their font databases. These approaches can thus only make use of the embedded font information to support limited font styles. In our work, the proposed robot is able to treat new writing styles by following the gestures of a human demonstrator in performing different stroke styles; in other words, the human-robot interactions allow our robot conveniently to learn writing

new strokes. Such a property is not often exhibited by the existing approaches.

IV. Conclusion

This paper proposed a new approach to implement robotic writing. The proposed robot system is capable of learning different Chinese strokes using human gestures and thus automatically writing individual Chinese characters. After learning stroke writing from a human demonstrator, the robot system can automatically decompose a character into strokes, then match the decomposed strokes to the corresponding robot trajectories, and finally w rite t he c haracter. T he s troke decomposition method is implemented mainly using the corner detection algorithm. To enable a robot to learn writing skills, a motion-sensing device is used to detect arm gestures of a human. Seven common Chinese characters are utilized in the experiments, which demonstrate that the proposed method can correctly decompose these characters into strokes and that the robot arm can successfully write Chinese characters in different styles on a writing board.

While the proposed approach is promising, there is room for improvement. In the present work, the stroke decomposing method is limited to an area with five c ross-corners, a nd the stroke decomposing method might not work properly if the contour shapes of the character are not smooth; in addition, the proposed method cannot treat shapes with holes or loops. Therefore, it is necessary to extend the method to support any number of cross-corners and to address the noise-sensitivity issues in the future. Additionally, stroke writing sequence is another crucial issue in robotic writing that represents an important factor in the quality of written Chinese characters [41]; further study is therefore required to determine the writing sequence for the system. Finally, the stroke trajectories are considered as planar movements during the trajectory learning stage; it would be very interesting to investigate the extraction of pen pressure to directly support the calligraphic robot.

Acknowledgement

The authors are very grateful to the anonymous reviewers for their constructive comments, which have helped significantly in revising this work.

References

[1] H. Zeng, Y. Huang, F. Chao, and C. Zhou, "Survey of robotic calligraphy research," *CAAI Transactions on Intelligent Systems*, vol. 11, no. 1, pp. 15–26, 2016.

[2] C. Yang, S. Chang, P. Liang, Z. Li, and C. Y. Su, "Teleoperated robot writing using emg signals," in *2015 IEEE International Conference on Information and Automation*, Aug 2015, pp. 2264–2269.

[3] Z. Ma and J. Su, "Aesthetics evaluation for robotic Chinese calligraphy," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 1, pp. 80–90, March 2017.

[4] F. Chao, J. Lv, D. Zhou, L. Yang, C. Lin, C. Shang, and C. Zhou, "Generative adversarial nets in robotic Chinese calligraphy," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1104–1110.

[5] N. Huebel, E. Mueggler, M. Waibel, and R. D'Andrea, "Towards robotic calligraphy," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vilamoura: IEEE Press, 2012, pp. 5165–5166.

[6] S. Mueller, N. Huebel, M. Waibel, and R. D'Andrea, "Robotic calligraphy - Learning how to write single strokes of Chinese and Japanese characters," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 1734–1739.

[7] V. Potkonjak, "Robot handwriting: why and how," in *Interdisciplinary Applications of Kinematics*, A. Müller, A. Kecskeméthy, and V. Potkonjak, Eds. Springer, Netherlands, 2011, pp. 19–35.

[8] F. Chao, Z. Wang, C. Shang, Q. Meng, M. Jiang, C. Zhou, and Q. Shen, "A developmental approach to robotic pointing via human-robot interaction," *Information Sciences*, vol. 283, pp. 288–303, 2014.

[9] F. Chao, F. Chen, Y. Shen, W. He, Y. Sun, Z. Wang, C. Zhou, and M. Jiang, "Robotic free writing of Chinese characters via human robot interactions," *International Journal of Humanoid Robotics*, vol. 11, no. 1, pp. 1 450 007–1–26, March 2014.

[10] K. Endo, M. Kanoh, and T. Nakamura, "Teaching handwriting using robot and onomatopoeia," in *2015 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, Nov 2015, pp. 484–490.

[11] F. Chao, Y. Huang, X. Zhang, C. Shang, L. Yang, C. Zhou, H. Hu, and C.-M. Lin, "A robot calligraphy system: From simple to complex writing by human gestures," *Engineering Applications of Artificial Intelligence*, vol. 59, pp. 1–14, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197616302329

[12] X. Ma, Q. Kong, W. Ma, and X. Zhang, "4-DOF lettering robots trajectory planning," in *Mechanical Engineering and Automation*, 2010, vol. 165, no. 5, pp. 161–163.

[13] Y. Man, C. Bian, H. Zhao, C. Xu, and S. Ren, "A kind of calligraphy robot," in *IEEE International Conference on Information Sciences and Interaction Sciences*, China, 2010, pp. 635–638.

[14] Y. Sun, H. Qian, and Y. Xu, "Robot learns Chinese calligraphy from demonstrations," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, pp. 4408–4413.

[15] J. Garrido, W. Yu, and A. Soria, "Human behavior learning for robot in joint space," *Neurocomputing*, vol. 155, pp. 22–31, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231214017329

[16] A. Droniou, S. Ivaldi, and O. Sigaud, "Learning a repertoire of actions with deep neural networks," in *4th International Conference on Development and Learning and on Epigenetic Robotics*, Oct 2014, pp. 229–234.

[17] F. Chersi, "Learning through imitation: a biological approach to robotics," in *IEEE Transactions on Autonomous Mental Development*, 2012, vol. 4, no. 3, pp. 204–214.

[18] P. M. Yanik, J. Manganelli, J. Merino, A. L. Threatt, J. O. Brooks, K. E. Green, and I. D. Walker, "A gesture learning interface for simulated robot path shaping with a human teacher," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 1, pp. 41–54, Feb 2014.

[19] D. Chen, G. Li, Y. Sun, J. Kong, G. Jiang, H. Tang, Z. Ju, H. Yu, and H. Liu, "An interactive image segmentation method in hand gesture recognition," *Sensors*, vol. 17, no. 2, pp. 253:1 – 253:17, 2017. [Online]. Available: http://www.mdpi.com/1424-8220/17/2/253

[20] J. P. Bandera, J. A. Rodríguez, L. Molina-Tanco, and A. Bandera, "A survey of vision-based architectures for robot learning by imitation," in *International Journal of Humanoid Robotics*, 2012, vol. 9, no. 1, pp. 1 250 006–1–1 250 006–40.

[21] Z. Lu, X. Chen, X. Z. Q. Li, and P. Zhou, "A hand gesture recognition framework and wearable gesture-based interaction prototype for mobile devices," in *IEEE Transactions on Human-Machine Systems*, 2014, vol. 44, no. 2, pp. 293–299.

[22] F. Chao, Y. Sun, Z. Wang, G. Yao, Z. Zhu, C. Zhou, Q. Meng, and M. Jiang, "A reduced classifier ensemble approach to human gesture classification for robotic Chinese handwriting," in *IEEE Conference on Fuzzy Systems (FUZZ-IEEE)*, 2014, pp. 1720–1727.

[23] P. Liang, C. Yang, Z. Li, and R. Li, "Writing skills transfer from human to robot using stiffness extracted from sEMG," in *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, June 2015, pp. 19–24.

[24] J. Liu, Y. Luo, and Z. Ju, "An interactive astronaut-robot system with gesture control," *Computational Intelligence and Neuroscience*, vol. 2016, pp. 7 845 102:1–7 845 102:11, 2016. [Online]. Available: https://doi.org/10.1155/2016/7845102

[25] R. B. Warrier and S. Devasia, "Iterative learning from novice human demonstrations for output tracking," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 4, pp. 510–521, Aug 2016.

[26] J. Saunders, D. S. Syrdal, K. L. Koay, N. Burke, and K. Dautenhahn, ""Teach me - Show me" – End-user personalization of a smart home and companion robot," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, pp. 27–40, Feb 2016.

[27] Z. Ju, X. Ji, J. Li, and H. Liu, "An integrative framework of human hand gesture segmentation for human-robot interaction," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1326–1336, Sept 2017.

[28] C. Yang, J. Chen, Z. Ju, and A. S. K. Annamalai, "Visual servoing of humanoid dual-arm robot with neural learning enhanced skill transferring control," *International Journal of Humanoid Robotics*, vol. To appear, p. 1750023. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0219843617500232

[29] X. Ao, Z. Wu, M. Zhou, R. Song, and Y. Wang, "The stroke extraction algorithm of brush strokes based on corner detection," in *The Third Session of the Conference on Image and Graphics Technology and Application*, BeiJing, China, 11 2008, pp. 173–176.

[30] F. Cao, Z. Wu, X. Ao, and M. Zhou, "Vectorization of Qi Gong calligraphy," in *Journal of Chinese Information Processing*, 2010, vol. 24, no. 6, pp. 97–102.

[31] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[32] F. Yao, G. Shao, and J. Yi, "Extracting the trajectory of writing brush in Chinese character calligraphy," in *Engineering Applications of Artificial Intelligence*, 2004, vol. 17, no. 6, pp. 631–644.

[33] ——, "Trajectory generation of the writingcbrush for a robot arm to inherit blockcstyle Chinese character calligraphy techniques," in *International Journal of Advanced Robotics*, 2004, vol. 18, no. 3, pp. 331–356.

[34] F. Yao and G. Shao, "Modeling of ancient-style Chinese character and its application to CCC robot," in *IEEE International Conference on Networking, Sensing and Control*, Piscataway, USA, 2006, pp. 72–77.

[35] Y. Sun, H. Qian, and Y. Xu, "A geometric approach to stroke extraction for the Chinese calligraphy robot," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3207–3212.

[36] C. Raju, D. N. Babu, and S. A. Babu, "Novel edge detection algorithm in eight different directions," *International Journal of Engineering Research and Applications*, vol. 2, no. 4, pp. 354–361, 2012.

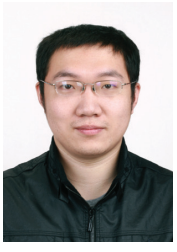
[37] R. Chen, "The corner detection of image based on harris algorithm," in *International Conference on Intelligent Systems Research and Mechatronics Engineering (ISRME 2015)*. Atlantis Press, 2015, pp. 1040–1043.

[38] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis, and machine vision*, 4th ed. Cengage Learning, 2014.

[39] L. Hou, J. Zhang, and L. Huo, "An improved algorithm for extracting the skeletons of the Chinese calligraphy," *Microcomputer & Its Applications*, vol. 17, p. 025, 2011.

[40] F. Chao, F. Chen, Y. Shen, W. He, Y. Sun, Z. Wang, C. Zhou, and M. Jiang, "Robotic free writing of Chinese characters via human robot interactions," in *International Journal of Humanoid Robotics*, 2014, vol. 11, no. 1, pp. 1 450 007–1–26.

[41] H. I. Lin and Y. C. Huang, "Visual matching of stroke order in robotic calligraphy," in *Advanced Robotics (ICAR), 2015 International Conference on*, July 2015, pp. 459–464.



Fei Chao (M'11) received a B.Sc. degree in Mechanical Engineering from Fuzhou University, China and his M.Sc. degree with distinction in computer science from the University of Wales, Aberystwyth, U.K. in 2004 and 2005, respectively, and a Ph.D. degree in robotics from Aberystwyty University, Wales, U.K. in 2009. He is currently an Associate Professor with the Cognitive Science Department, Xiamen University, China. Dr Chao has published more than 30 peer-reviewed journal and conference papers. His research interests include developmental robotics, machine learning, and optimization algorithms.

Yuxuan Huang received his B.Sc. and MEng degrees in Cognitive Science and Technology from the Xiamen University, China in 2014 and 2017, respectively. He is further continuing his study at Xiamen University. His research interests include Chinese calligraphic robots and developmental learning algorithms.

Chih-Min Lin (M'87SM'99F'10) was born in Taiwan in 1959. He received his B.S. and M.S. degrees from the Department of Control Engineering and a Ph.D. degree from Institute of Electronics Engineering, National Chiao Tung University, Hsinchu, Taiwan, in 1981, 1983 and 1986, respectively. He is currently a Chair Professor and the Vice President of Yuan Ze University, Chung-Li, Taiwan. His current research interests include fuzzy neural networks, cerebellar model articulation controllers, intelligent control systems and signal processing. He has published more than 170 journal papers. Professor Lin is an IEEE Fellow.

Longzhi Yang (M'12-SM'18) is currently a Programme Leader and a Senior Lecturer with Northumbria University, Newcastle upon Tyne, U.K. His research interests include computational intelligence, machine learning, big data, computer vision, intelligent control systems, and the application of such techniques in real-world uncertain environments. He is the Founding Chair of the IEEE Special Interest Group on Big Data for Cyber Security and Privacy. Dr. Yang received the Best Student Paper Award from the 2010 IEEE International Conference on Fuzzy Systems.

Huosheng Hu (M'94-SM'01) received his M.Sc. degree in industrial automation from the Central South University, Changsha, China, in 1982 and a Ph.D. degree in robotics from the University of Oxford, Oxford, U.K. in 1993. He is currently a Professor with the School of Computer Science and Electronic Engineering, University of Essex, Colchester, U.K., leading the Robotics Research Group. He has authored approximately 450 papers in journals, books, and conferences in these areas. His current research interests include behavior-based robotics, human-robot interaction, service robots, embedded systems, data fusion, learning algorithms, mechatronics, and pervasive computing.

Changle Zhou Chang-Le Zhou received his PhD from Peking University in 1990. Currently, he is a professor in the Cognitive Science Department at Xiamen University. He is also an affiliated professor of linguistics and applied linguistics in the Humanity College at Zhejiang University and an affiliated professor of the Philosophy Department of Xiamen University. His research interests lie in the areas of artificial intelligence. His scientific contribution to AI has more to do with machine consciousness and the logic of mental self-reflection.