Proceeding of the 2006 IEEE
International Conference on Automation Science and Engineering
Shanghai, China, October 7-10, 2006

Evolutionary Replication of Calligraphic Characters By A Robot Drawing Platform Using Experimentally Acquired Brush Footprint*

Ka Wai Kwok, Ka Wah Lo, Sheung Man Wong and Yeung Yam, *Senior Member, IEEE*

Dept. of Automation & Computer-Aided Engineering.
The Chinese University of Hong Kong
Shatin, NT, Hong Kong
kwkwok, kwlo, smwong, yyam@acae.cuhk.edu.hk

***Abstract* – In this paper, the techniques of an experiment-based brush footprint acquisition method and a Genetic Algorithm (GA)-based brush stroke generation scheme are combined. The combined method is applied to replicating a given Chinese character. Additionally, a new method to come up with the brush stroke skeleton to facilitate Bézier representation of the brush stroke, rather than the Thinning algorithm in previous works, is also incorporated. Actual experiments were conducted to demonstrate the performance of combined method and to compare with the original character. The present paper is part of an endeavour to attempt replicating the works of famous past calligraphers.**

***Index Terms* –Bézier curve, brush stroke, calligraphy painting, control points, robot platform.**

I. Introduction

Numerous works on virtual and robot painting of Chinese calligraphies and paintings have been reported in recent years. Among them, Chu, et la. [1] proposed a physically-based 3D brush model with spreading bristles and rendering strokes in real time to enable users to create highly aesthetic brushworks. Mano [2] proposed a method to generate brush-style writing of Japanese Hiragana characters, but the method requires information on pen-position, pressure and speed on human writing. Also, Zhang, et la. [3] developed a calligraphic robot to paint calligraphic stroke by controlling the brush width and orientation through a multi-sensor fusion system. The work, however, involves no actual brush manipulation. Rather, it utilizes only calligraphy image input to create artistic brushwork.

We have constructed a robot drawing platform for the actual generation and replication of Chinese calligraphies and drawings in our laboratory. The platform serves as a testbed to experiment and validate our proposed calligraphic schemes and methodologies. The platform, which constitutes the top part of the setup shown in Fig.1, is equipped with an automated manipulation system capable of controlling pen/brush movement. The robot manipulator supports four degrees of freedom DOFs (*x*, *y*, *z* and *z*-rotation) of a brush pen motion to emulate the needed movement in calligraphic movements. Interested readers may refer to [4] for more detailed description.

In a previous work [5], we have proposed a brush stroke generation scheme based on Genetic Algorithms (GA). The work generates a painting scheme through evolutionary computing process via the minimization of a properly defined objective function. Under the proposed scheme, brush stroke trajectories are described by Bézier curves. And the Bézier control points, which include the coordinates (*x, y*) and brush painting *z*-depth, are taken as chromosome. The work, however, is applied to replicating given Chinese characters using only fictitious footprint models such as the water drop or tear-shaped templates.

The present work differs from that of [5] in the following ways:

1. The brush artificial template model is actually captured using a transparent setup with vision system.
2. The control points are automatically generated via the CAT algorithm, instead of the Thinning algorithm as in past works
3. The corresponding control points can be distributed unevenly inside the stroke by the non-linear functions we assigned
4. The GA stroke simulation painting scheme is executed in the transparent setup, allowing an actual comparison of the character would-be-executed by the robot platform with the original character.

The present paper is organized as follows. The next section describes the transparent system to acquire the actual brush stroke footprint. This is followed by a description of trajectory representation using Bézier curves in Section III. The GA-based brush stroke generation scheme is presented in Section IV. Section V gives an experimental comparison of the replication results using the robot on the transparent system. Finally, Section VI gives the conclusions.

II. Footprint Acquisition by Visual System

The basic idea of the visual system is shown in Fig.2. As the brush moves on the transparent glass plate upon certain command, robot, its brush footprint is captured by a video camera system placed below the plate and looking upwards. The captured footprints are then to be correlated

* This work was supported by the Hong Kong Research Grant Council under CUHK Direct Grant 2050342.

with the input commands for a non-parametric characterization of their relations. Fig.3(a) shows the actual transparent setup and the installed camera system which are housed in the lower part of the setup shown in Fig.1. The system uses a Sony EVI-D30/D31 Pan/Tilt/Zoom Colour Video camera with a resolution of 640× 480 pixels and the ability to capture 20-30 frames per second. These video frames are indexed with the corresponding time and commands for ensuing analysis. The transparent plate in our case is actually in the form of a container. At times, blue liquid are put in the container to enhance the effect of the footprint capturing, as the case shown in Fig.3(b). The blue liquid acts as the background of the video captured, providing a good contrast to the yellow brush tuft of the footprint.

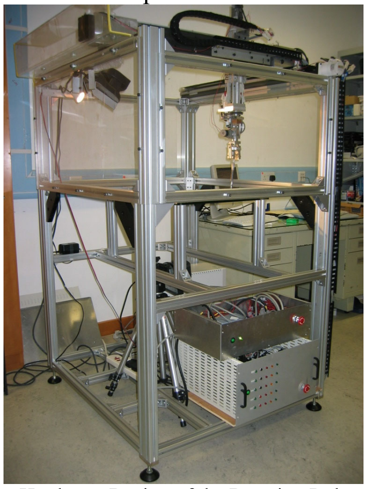


Fig.1. Hardware Design of the Drawing Robot

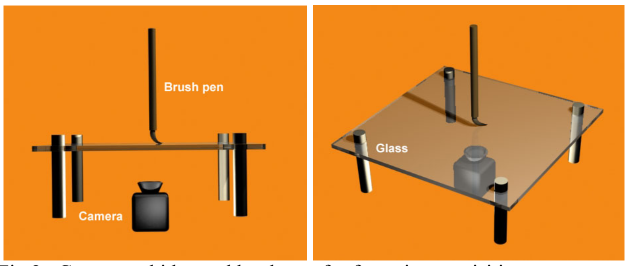
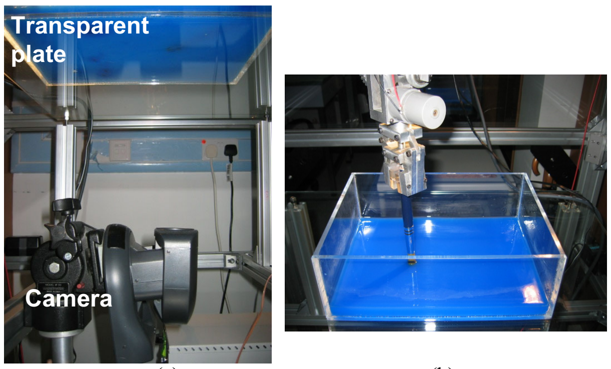


Fig.2. Conceptual idea and hardware for footprint acquisition



(a) (b)
Fig.3. Actual hardware for footprint acquisition: (a) Video camera, (b) the transparent drawing setup

The captured video needs to be transformed into a full plane view as if observed directly from below the transparent plate. This Projective Rectification process [6] serves to remove the projective distortion in the perspective video image. This process allows the captured image to be rectified in a full plane view with proper scaling so that the rectified image has the same pixel resolution as the original stroke image and subsequent stroke quality measurements can be based under the same pixel dimensions. The strict requirement of having the camera system looking up at 90° may also be relaxed.

Fig.4 shows the sampled footprint of one of the executed strokes with time indexing. The *z*-axis painting depth in the case is *z*= - 4.5mm. We can actually take the union of these footprints along the video frames to yield the would-be-executed brush stroke, as shown in Fig.5.

Fig.4. Captured footprint with time indexing for *z*=-4.5mm

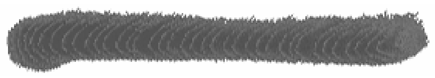


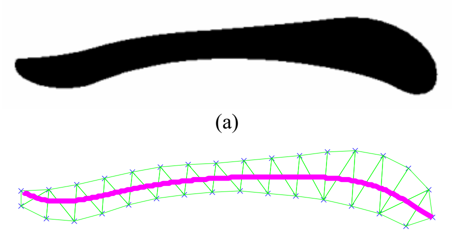
Fig.5. Would-be brush stroke achieved by taking the union of the captured footprint

III. Brush Stroke Trajectory Representation

The study of Bézier curve falls under the general topic of curve fitting. Its true value lies actually not in scientific studies but in artistic purposes. Originally used by car designers to create pleasant looking curves, the Bézier curve is now used by graphic artists in many fields where the generation of curved shapes is necessary. In our case, it provides us with a simple model for representing stroke painting trajectory. The advantages of choosing Bézier curve to represent the painting trajectory rather than B-spline or spline curve are mentioned in [5]. Mathematically, Bézier curve is expressed in the following form:

$$B(u)=\sum_{k=0}^{n-1} P_k \binom{n-1}{k} u^k (1-u)^{n-1-k} \quad , \quad \text{for} \quad 0 \geq u \geq 1 \; , \qquad (1)$$

where $P_{k=0\ldots n-1}$ is the sequence of control point position.


(a)

(b)

Fig.6. (a) The Chinese character "一" in Running Script style "行書"; (b) the stroke mesh obtained by CAT

A. *Stroke Skeleton Extraction*

Proper Bézier control point selection will have a strong effect on generating stroke trajectory with artistic appearance. Here, we base the control point selection on first obtaining the stroke skeleton, which run more or less along the middle of the stroke segment. Thinning Algorithm is one of the most common pre-processing steps for such feature extraction on many pattern recognition systems and has been used in our previous works [5]-[6]. The algorithm results in on-pixel-wide skeleton of the image foreground. However, for some thick and high resolution stroke images, it usually generates spurious skeletons such as splitting some forked line segment at the stroke terminals. These defects will usually result in the failure of subsequent control point estimation.

In this work, we adopt another morphological transform called the Chordal Axis Transform (CAT) [7] which serves to obtain the skeleton from polygon shape. Firstly, we divide the boundary of each stroke into equal edge unit length. The optimal edge unit length a given boundary can be determined using the Constrained Delaunay Triangulation (CDT) of polygons. In order to have optimal edge unit length, which highly affects the performance of CDT process, we used an iterative search algorithm with pre-defined initial edge unit length. In our case, we adopted 30 pixels as the initial value. Selection is conducted based on maximized coverage using the triangular elements but with minimized overlapping of the inner stroke. Once the desired edge unit length is determined, the appropriate inner edges are thus generated accordingly. Fig.3(a) depicts the character "一" in Running Script style "行書", one of the major categories of Chinese calligraphy. Fig.3(b) shows the stroke triangular mesh as generated by CAT. The Bézier curve is then determined by the middle points of the internal edges. This gives rise to the corresponding skeleton as shown.

B. *Control Point Sequence Assignment*

In human calligraphy and painting, complicated brush manipulation is required usually at the beginning and ending region of a stroke. To increase the flexibility of trajectory forming, denser control points should appear at these stroke regions. Thus, we would like to adopt a non-linear sigmoid function

$$sig(x) = \frac{1}{1+e^{-x}} \tag{2}$$

for uneven control point sequence distribution. Since the stroke skeleton is represented by a Bézier curve $B(u)$ $u \in [0,1]$ as well, we can assign n set of perpendicular control point line sequences at individual $u_{i=0,..,n-1}$. The intervals u_i are computed as follows:

$$u_i = \frac{sig\left(r_0 + \dfrac{i(r_1 - r_0)}{n-1}\right) - sig(r_0)}{sig(r_1) - sig(r_0)}, \quad \text{for } i=0\ldots n-1 \tag{3}$$

where $u_0 = 0$, $u_{n-1} = 1$, $r = [r_0, r_1]$ is the tuning factor of linearity with $r_0 < 0$ and $r_1 > 0$. For $(r_1 - r_0)$ small, the distribution of control point sequences become even. Otherwise, more control points will cluster at two ends than the middle of the stroke. Furthermore, for $|r_0| > r_1$, denser control point sequences are placed at the start than at the end and the middle region. Similarly, denser point sequences are at the end region for $|r_0| < r_1$. Fig.7(a) shows the example with 17 perpendicular control point line sequences formed even along the skeleton curve. Moreover, Fig.7(b) and (c) demonstrate two case of uneven distribution of the control line sequences achieved using the r values. In some complicated stroke examples, a turning brush motion is usually required in other to form a artistic curving contour at the start or termination region. Thus, the corresponding control point sequences at this region should contribute to such turning effect in the resulting Bézier trajectory. In Fig.8, numerous painting trajectories with such curly effects in both ends are fitted by different control points from the sequence index with the possibility of repeating: 4→ 3→ 2→ 1→ 0→ 1→ 2→ 3→…→ 16→ 17→ 16→ 15→ 14→ 13. Counting all the combination of possible selections in these 25 groups of control points, there is astronomical sum of 1.7879×10^{16} of 2D trajectories that can be generated for the example. We will apply GA-based evolution methods to efficiently come up with a good selection.

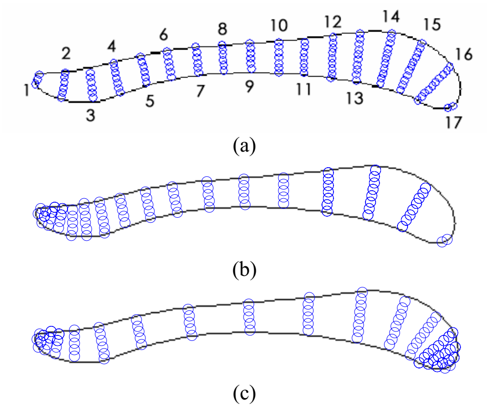


Fig.7. 17 control point line sequences are distributed inside the stroke, (a) r =[-0.1, 0.1]; (b) r =[-3.5, 0.5]; (c) r =[-4.5, 4.5]

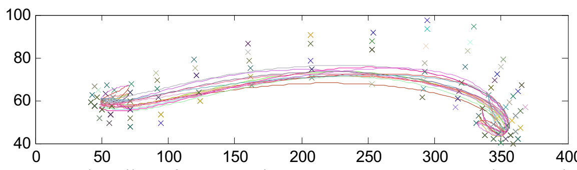


Fig.8. A bundle of 15 Bézier curves represent various painting trajectories

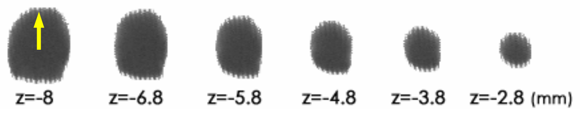


Fig.9. A set of brush footprints captured in different painting depths with motion direction defined. The direction is shown by the arrow.

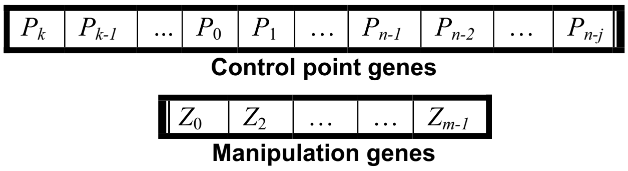
IV. GA-BASED BRUSH STROKE GENERATION

A realistic brush stroke model is another vital factor in order to achieve good trajectory result. In this work, rather than using the artificial brush templates in [5] and [8], we are to obtain the experimental footprints and use them as the brush model in our GA-based stroke generation. Here, the simple kinematics model of the brush tuft is utilized as starting point for rendering the artistic stroke. Fig.9 shows some real footprints of different sizes due to different applied painting depths along their straight running trajectories. In the simulation process, they will be "stamped" tangentially and also centered along the Bézier curve as according to the applied painting depths.

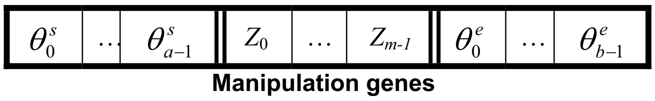
The question remains is that there is no standardized method for determining the stroke trajectory. In what follows, we use GA to determine the (x, y) trajectory and z-depth altogether upon a given brush stroke. GA serves to effectuate a simple search process rather than the need to go into detail studies of the brush geometry.

A. GA Chromosome for Stroke Painting

The GA chromosome C_i can be regarded as the kinematics brush execution that uses $P_{j=0,\ldots,n-1}$ as the Bézier curve control point genes and $Z_{j=0,\ldots,m-1}$ as the z-depth (in direction perpendicular to the paper) manipulation genes, see Fig.10, where n is the number of control point group and m is the number of z-depth quantization. For control point genes, as we mentioned there would be k and j groups with $k+j<$n, respectively, repeated at the start and the end of the stroke.


P_k | P_{k-1} | ... | P_0 | P_1 | … | P_{n-1} | P_{n-2} | … | P_{n-j}
Control point genes
Z_0 | Z_2 | … | … | Z_{m-1}
Manipulation genes
Fig.10. Chromosome structure of brush stroke represented by Bézier curve control point genes and manipulation genes

The chromosome structure above allows future inclusion into the manipulation genes additional DOFs to be constructed into the robot hardware. For example, as Fig.11 indicates, the manipulation genes can be expanded to include the tilted motion angles $\theta^s_{i=0..a-1}$ and $\theta^e_{i=0..b-1}$, respectively, at the start and end region of painting trajectory, with $a+b<m$.


θ^s_0 | … | θ^s_{a-1} | Z_0 | … | Z_{m-1} | θ^e_0 | … | θ^e_{b-1}
Manipulation genes
Fig.11. Manipulation genes consist of brush painting depth and title angle

B. Objective and Fitness Evaluation

The objective function is defined to measure how the individuals performed in stroke generation. In the case of a minimization problem, the most fitted individual will have the lowest numerical value of the associated objective function. The section function is dependent on the specific problem at hand. Consider the sample stroke in Fig.6(a) depicting the character "—". Under our criteria, the stroke to be generated should resemble closely the stroke as shown. We hence design the overall objective function to include the following three sub-objectives.

Sub-objective 1: Painting to be within stroke boundary.

Thinning Algorithm is a procedure to iteratively remove boundary pixels from a given sample image. With repeated applications of iterative removal, the thinner the remaining image would become. Using such procedure, one can divide the sample stroke into two regions. Region 1, as shown in Fig.12(a), is the sample stroke thinned iteratively until the remaining number of pixel is about 35% of the original number of pixels. Region 2, as shown in Fig.12(b), is obtained by truncating region 1 from the given sample stroke. The sub-objective 1 of the cost function is thus to minimize:

$$ObjV_1 = w_1 p_1 + w_2 p_2\,, \tag{4}$$

where p_i is the number of unpainted pixels within the region i, and w_i is the corresponding cost weighting of the region i. As the goal is to have the painting inside the stroke boundary as much as possible, we usually set $w_1 > w_2$ during the GA evolution.

Sub-objective 2: No painting outside stroke boundary

On the other hand, dilation is a morphological operation to expand image object. The procedure effectively sticks a pixel "layer" on the image object. Fig.13(a) shows the sample stroke being dilated iteratively until the added "layer" constitutes greater than 85% area of the sample stroke. This added layer is labelled as region 3. The remaining region beyond region 3 as depicted in Fig.13(b) is labelled as region 4. Both region 3 and 4 are outside the stroke boundary. The sub-objective 2 of our cost function is thus to minimize:

$$ObjV_2 = w_3 p'_3 + w_4 p'_4 \tag{5}$$

where p'_i is the number of painted pixels within the white region i. Similarly, we set $w_4 > w_3$ to impose the desire of not painting in region 3 and even less so in region 4.

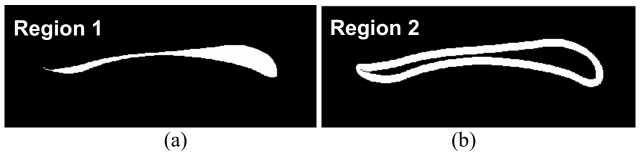


Fig.12. The sample stroke is divided into two regions in white: (a) inner region, (b) outer region, those are inside the stroke

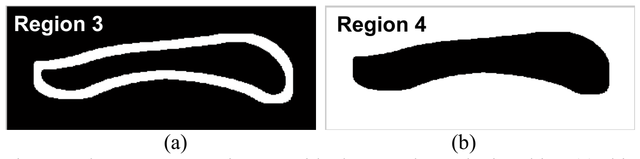


Fig.13. There are two regions outside the sample stroke in white: (a) skin region; (b) out-of-skin region

Sub-objective 3: Smooth painting depth change

The actual applied painting depth over the whole Bézier curve is the linear interpolation over z-depth values at m manipulation points distributed evenly along the curve. In real calligraphy painting, the force applied on the brush pen usually increases or decrease gently for smooth painting. It is thus necessary to have a penalty scheme for abrupt z-depth change along the trajectory. Sub-objective 3 is hence to minimize:

$$ObjV_3 = w_5 \sum_{i=0}^{m-2} \left| Z_{i+1} - Z_i \right|, \tag{6}$$

where Z_i is the z-depth values applied on the i^{th} control points.

The total objective cost function of the present study is the sum of three sub-objectives as introduced:

$$ObjV(C_i)=ObjV_1(C_i)+ObjV_2(C_i)+ObjV_3(C_i) \tag{7}$$

The fitness function serves to transform the objective function value into a measure of relative fitness. This mapping is necessary as the objective function is to be iteratively reduced in the quest of a "fitter" individual in the next generation. In this case, a non-linear fitness assignment is adopted to prevent premature convergence. Individuals are assigned fitness value according to their rankings in the population rather than their raw performance as,

Ranking: $x_i = ranking(ObjV(C_i))$, (8)

where x_i is the position in the ordered population of the i^{th} individual according to its objective cost in (7), $i=1,\ldots,N_{ind}$, and N_{ind} is a population size in each generation. Thus, the chromosome with lower objective cost implies higher integer ranking x_i in the population. The fitness of an individual in the population is calculated as,

Fitness: $F(x_i) = 2 - PRS + 2(PRS-1)\dfrac{x_i - 1}{N_{ind} - 1}$, (9)

where the parameter PRS is typically chosen within the interval [1.1, 2.0]. The fitness assignment ensures that each individual has a probability of reproducing according to its relative fitness.

C. Evolutionary computing parameter settings

The following genetic parameters and operations which are tested and tuned in many concrete brush stroke cases before are adopted for the present stroke generation.

1) Control point distribution parameters: $r=[r_0, r_1]$
2) No. of control points in group i: $n\_cp_{i=0\ldots n-1}$
3) Total no. of control points inside the stroke:
$$cpn = \sum_{i=0}^{n-1} n\_cp_i \tag{10}$$
4) No. of points interpolated over the curve: n_pt
5) No. of painting depth level applied: There are m motion points and the value usually is ranging from 0 to 0.85mm, i.e., z-depth value are defined: $[Z_{\min}, -Z_{\max}]=[0, 0.85]$
6) Stroke region division: the area of region 1, 2 and 3 occupy the sample stroke in percentage R_i=[35, 65, 85]%
7) Objective value weighting: $w_i=[w_1, w_2,\ldots, w_5]$=[3, 1, 1, 3, 2]
8) Chromosome length in binary coded:
$$l_{bit} = \sum_{i=1}^{k}\lceil \log_2(n\_cp_i)\rceil + \sum_{i=1}^{n}\lceil \log_2(n\_cp_i)\rceil + \sum_{i=n-j}^{n-1}\lceil \log_2(n\_cp_i)\rceil + m\lceil \log_2(Z_{\max}-Z_{\min}+1)\rceil \tag{11}$$
9) Crossover method: Multi-point crossover with probability p_c
10) Mutation method: New individuals are generated by taking the current population and mutating each element with probability $p_m=0.7/l_{bit}$
11) Selection method: Stochastic University Sampling with population selection p_s
12) Replacement: Fitness-base reinsertion to current population
13) Population size N_{ind} : between 40-100, depending on the length of the chromosome
14) Number of generation past: N_{gen} =151
15) Initial population: randomly generated within the groups of control points and range of painting depth

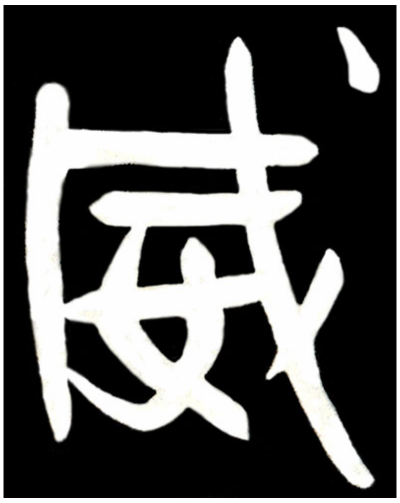


Fig.14. A 400×494 image calligraphic character "威" (source: [9])

V. Evolutionary Calligraphic Painting Results

Before the robot execution, the GA stroke generation is performed on a PC with AMD Athlon 64 1.81GHz CPU, 1GB RAM, using Windows XP. The computation time for generating one solution is roughly 15-25mins. This computation time (average by multiple runs) is for reference only since the program is running in the debug mode of MatLab environment. The actual speed should be much faster.

We conduct stroke generation for the Chinese character as depicted in Fig.14. The character "威" (means might and power) by a prestigious ancient calligrapher called Gan Shinkei (顏真卿) was taken from his masterpiece Hai Shōgun shi (裴將軍詩). The character is decomposed manually into the 9 strokes. Corresponding control points in groups are then distributed automatically along the individual strokes before subjecting to the GA-based evolution process. Fig.15 shows the re-assembled character formed by the 9 strokes obtained only after the first generation of the evolution process. The evaluated objective cost is 66,199. Upon 151 generations later, the objective cost is reduced to 7,688. The corresponding calligraphic stroke results and brush trajectories are shown in Fig.16, with "x" indicating the start point. Finally, using the GA-based simulation trajectory, the robot executed the painting scheme on the transparent setup. The executed character is formed by the union of captured footprints as shown in Fig.17.

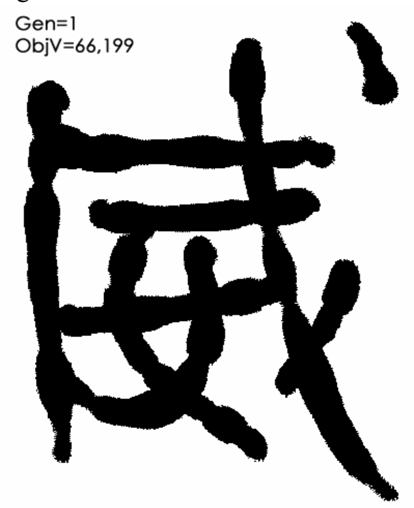


Fig.15. The character "威" recomposed with strokes generated from beginning of evolution process

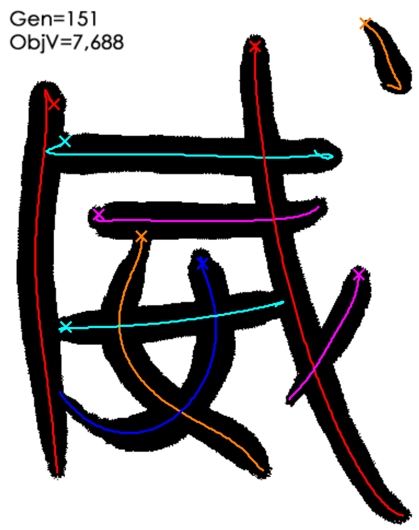


Fig.16. The resulting strokes and brush trajectories obtained after 151 generations of evolution process

VI. Conclusions

This paper combines the techniques of an experiment-based brush footprint acquisition method and a Genetic Algorithm (GA)-based brush stroke generation scheme. A CAT method to conduct triangulation of the given brush stroke and then selection of the best edge unit length is also implemented as a new way to determine the stroke skeleton, instead of the normal Thinning algorithm utilized in many of our previous works., is also incorporated. The overall method is applied to replicating a given Chinese character "威" via the replication of its various strokes. The hardware for footprint acquisition, which includes the transparent tank and the camera system looking upward, provides a setup to compare the would-be-executed stroke by the robot and the original character. The overall performance of the results is quite adequate given the rather complicated nature and the many strokes involved in the character "威". It can be observed that the replicated character captured via the transparent tank is quite close to the original version. For the next step of our work, we plan to conduct actual execution on real paper for comparison, where the full realistic effects of Chinese calligraphy will be taken into account.

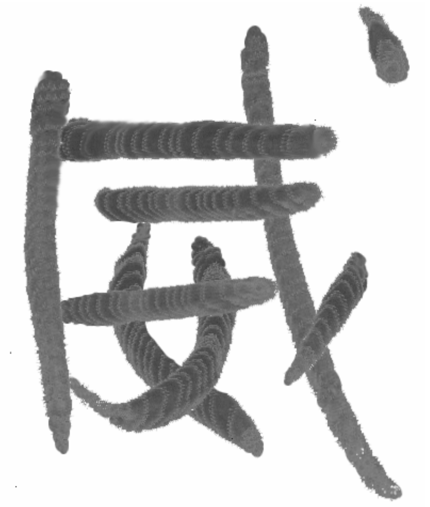


Fig.17. Robot execution character formed by the union of captured footprints

References

[1] Nelson S.-H. Chu and C.-L. Tai, "Real-time Painting with an Expressive Virtual Chinese Brush", *IEEE Computer Graphics and Applications*, Vol. 24, No. 5, Sept-Oct, 2004, pp. 76-85.

[2] Mano, J.; He, L.; Nakamura, T.; Enowaki, H.; Mutoh, A.; Itoh, H, "A method to generate writing-brush-style Japanese Hiragana character calligraphy," Proc. of 1999 *IEEE International Conference on Multimedia Computing and Systems*, Vol.1, 7-11 June 1999, pp.787 - 791

[3] Kejun Zhang and Jianbo Su, "On Sensor Management of Calligraphic Robot," Proc. of the *2005 IEEE International Conference on Robotics and Automation*, 18-22 April 2005, pp.3570 – 3575

[4] K.W. Lo, K.W. Kwok, and Y. Yam, "Automated Replication of Line Drawings By a Robot Drawing Platform", Proc. of *the 8th World Multi-Conference on Systemics, Cybernetics and Informatics*, Orlando, Florida, USA, July 18-21, 2004, pp. 80-85.

[5] K.W. Kwok, S.M. Wong, K.W. Lo, and Y. Yam. "GA-based Brush Stroke Generation for Replication of Chinese Calligraphic Character," Proc. of *IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, Vancouver, BC, Canada, July 16-21, 2006, pp.3802-3809.

[6] K.W. Kwok, Y. Yam and K.W. Lo, "Vision System and Projective Rectification for A Robot Drawing Platform", Proc. of the *2005 International Conference on Control and Automation (ICCA'05)*, Budapest, Hungary, June 27-29, 2005, pp.691-695.

[7] L. Prasad. Morphological analysis of shapes. CNLS Newsletter, July 1997, 139: pp.1-18.

[8] Xiaofeng Mi; Jie Xu; Min Tang; Jinxiang Dong, "The droplet virtual brush for Chinese calligraphic character modeling," Proc. of *the Sixth IEEE Workshop on Applications of Computer Vision, 2002. (WACV 2002),* 3-4 Dec. 2002, pp.330 – 334

[9] Shodō geijutsu -- Yan Zhenqing, Liu Gongquan / Nakata Yūjirō sekinin henshū, Tōkyō : Chūō Kōronsha, Shōwa 45-47 [1970-1972]