2006 IEEE Congress on Evolutionary Computation
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

# Genetic Algorithm-Based Brush Stroke Generation for Replication of Chinese Calligraphic Character

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

***Abstract*—This paper presents a novel brush stroke generation scheme based on Genetic Algorithms (GA) and pre-defined brush template models. The work is part of an endeavor to attempt imitating master works of famous past calligraphers. The concept is to parametrize, in some computational sense, the writing styles and techniques of certain calligraphers and then executes the results in a robot drawing platform developed in our laboratory. The present work describes the algorithmic development, simulation studies and experimentation of the GA-based stroke generation scheme upon given certain calligraphic characters. To further study the effectiveness of calligraphic writing with the robot platform, the Cross-Entropy method of the Traveling Salesman problem is incorporated to determine the sequence of stroke execution.**

## I. Introduction

Chinese characters are made up of basic strokes. As such, the artistic values of a piece of Chinese calligraphy depend highly on the quality of the execution and control of its composing strokes. With due consideration of the thickness and absorptivity of the paper, and expert controlling of the ink concentration and brush flexibility, an artist has the freedom to produce infinite variety of styles and forms [1].Unlike other visual art techniques, Chinese calligraphic strokes are permanent and incorrigible once after execution. Careful planning and confident manipulation are hence the keys to the producing a work of high artistic level. Fig.1 highlights the stroke trajectory for executing a basic stroke in the formation of the Chinese character "中" (means center or middle).

In recent years, numerous works on simulation of Chinese calligraphies and paintings have been reported, e.g., Chu, et la. [2]-[3] developed a physically-based 3D brush model with spreading bristles and rendering strokes in real time. The users are able to create much aesthetic brushworks with this virtual brush. Also, Lee [4] and Chu [5] proposed a method to simulate real-time dynamic of ink diffusion in absorbent paper for art creation. Additionally, Xu, et la. [6] proposed an intelligent system to create calligraphy artwork of various styles through an interpolation process using input samples of trained font style as knowledge sources. The work, however, did not deal with brush manipulation to create artistic strokes.

In this paper, we propose a brush stroke generation scheme based on Genetic Algorithms (GA) and pre-defined brush template model. We describe the brush stroke trajectory by Bézier curve and take as chromosome the Bézier curve control points which are the coordinates $(x, y)$ and brush painting $z$-depth. A painting scheme is then generated through evolutionary computing process via the minimization of a properly defined objective function. Upon generation of the brush strokes, however, we still have to decide the sequence by which the strokes are executed. In this work, we attempt the use of the Traveling Salesman problem (TSP) solver, specifically, the Cross-Entropy (CE) method, to automatically determine the sequence. Comparison of the TSP-CE method with the traditional way of assigning the sequence is also discussed.

(a) (b) (c)

Fig.1. (a) One of the stroke trajectories for the Chinese character "中"; (b) The resulting full stroke upon execution of the trajectory of (a); (c) The character "中" is completed when all stokes are executed.

While previous works on Chinese brush strokes cited above mainly confined themselves in simulation case studies only, our generated brush strokes are intended for robot execution. In this regards, we have designed and constructed in our laboratory a robot drawing platform [7]-[8] aimed at studying Chinese painting and calligraphy. Referring to Fig.2, the platform and the gripper together provides a total of 5 degrees of freedom (DOFs) (consists of x, y, and z translation, z-rotation, and pitch) for the brush pen to emulate the hand and wrist movement need in fine execution of brush strokes. The results of the present work on GA-based stroke generation and TSP-CE sequence assignment will be put to execution on the robot platform. The results will be reported

K.W. Kwok is with the Department of Automation and Computer-Aided Engineering, the Chinese University of Hong Kong, Shatin, Hong Kong (corresponding author to provide phone: 852-2609-8040; fax: 852-2603-6002; e-mail: kwkwok@acae.cuhk.edu.hk)

The authors are with the Department of Automation and Computer-Aided Engineering, the Chinese University of Hong Kong, Shatin, Hong Kong (e-mail: smwong@acae.cuhk.edu.hk, kwlo@vtc.edu.hk, yyam@acae.cuhk.edu.hk)

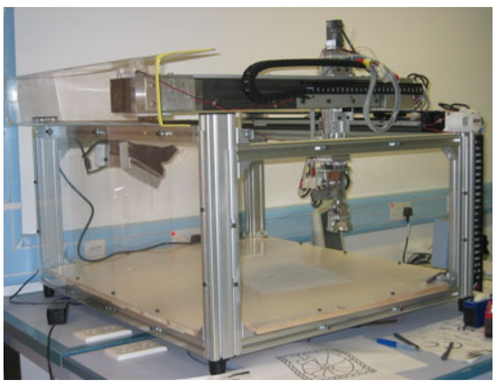
in a future work.

## II. Bézier Curve 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. Mathematically, Bézier curve is expressed in the following form:

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

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



(a)

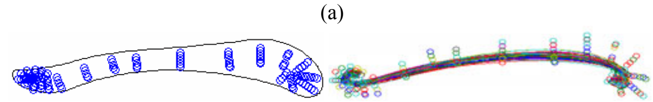
(b)

Fig.2. (a) The robot drawing platform with the Gripper; (b) The gripper holding a drawing pen.

The advantages of choosing Bézier curve to represent the painting trajectory rather than B-spline or spline curve are that: 1) It relaxes the stringent requirement on the painting trajectory – it does not pass though any of the control points except the first and the last; 2) We need not to input any "knots" as B-spline for determining the degree of curve polynomial. In Bézier curve, $degree(B(u))$ < the number of control points, $n$; 3) It allows easier generation of stroke with "better" appeal – multiple control points clustered at a region in space will add more weight to those points "pulling" the Bézier curve towards them.

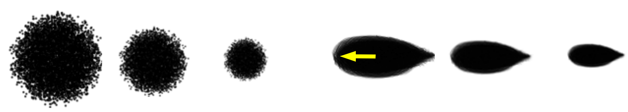
Fig.3(a) depicts the Chinese number character one "一" in Running Script style "行書", which is one of the major categories of Chinese calligraphy. Fig.3(b) depicts the 121 manually assigned control points gathered in 23 groups distributed along the length of the stroke. The selection of a control point from each of these 23 groups determines a Bézier spine curve, a few of which are shown in Fig.3(c). Counting all the combination of possible selections, there is an astronomical sum of $1.9508 \times 10^{16}$ of 2D trajectories that can be generated for the example. Obviously, the exhaustive search method to pick the best Bézier curve is not desirable. This motivates the use of GA-based technique.

(a)



(b) (c)

Fig.3. (a) The Chinese character "一" in Running Script style "行書"; (b) The 121 assigned control points in 23 groups along the length of the line segment; (c) A bundle of Bézier curves represent various painting trajectories.



(a) (b)

Fig.4. (a) Spray paint templates under different painting depth descending from left to right; (b) another brush templates in water drop shape with front direction defined which is shown by the arrow

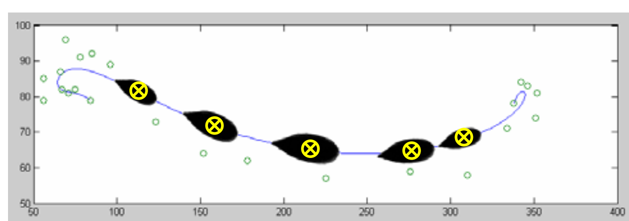



Fig.5. The brush templates chopped tangentially along and centered at the Bézier curve under different $z$-depth applied

## III. GA-based Brush Stroke Generation

Besides the trajectory, the brush stroke model is another vital factor to achieve good painting result. To illustrate our approach, we use the two simple, pre-defined 2D brush pen models as shown Fig.4, the spray paint template and the water drop template. Here, the simple kinematics model of the brush tuft is utilized as starting point for rendering the artistic stroke. For each, it is assumed that the footprint will be enlarged proportionally as the writing $z$-depth increases.

Fig.5 shows some brush templates in different sizes due to their applied $z$-depths along a known trajectory.

The question remains 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.

The models we attempted in the present work as depicted in Fig.4 are simple ones. In time, we may attempt more realistic 2D brush templates and study how they would affect the corresponding stroke generation and resulting artistic quality. In this regards, we have initiated a process to capture, model, and parametrize the resulting 2D brush stroke footprints during actual execution, which will be presented in later works.

### *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.6, where $n$ is the number of control point group and $m$ is the number of $z$-depth quantization.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $P_0$ | $P_2$ | … | … | $P_{n-1}$ | $Z_0$ | $Z_2$ | … | … | $Z_{m-1}$ |
| **Control point genes** | | | | | **Manipulation genes** | | | | |

Fig.6. 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 or virtual brush painter in [3]. For example, as Fig.7 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 $a+b<m$.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $\theta^s_0$ | … | $\theta^s_{a-1}$ | $Z_0$ | … | $Z_{m-1}$ | $\theta^e_0$ | … | $\theta^e_{b-1}$ |

**Manipulation genes**

Fig.7. 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 selection of objective function is dependent on the specific problem at hand. Consider the sample stroke in Fig.3(a) depicting the character "一". 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 [9]. 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.8(a), is the sample stroke thinned iteratively until the remaining number of pixels is about 35% of the original number of pixels. Region 2, as shown in Fig.8(b), is obtained by truncating region 1 from the given sample stroke. The sub-objective 1 of the cost function is thus to minimize:

$$ObjV_1 = w_1 p_1 + w_2 p_2\,, \tag{2}$$

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.

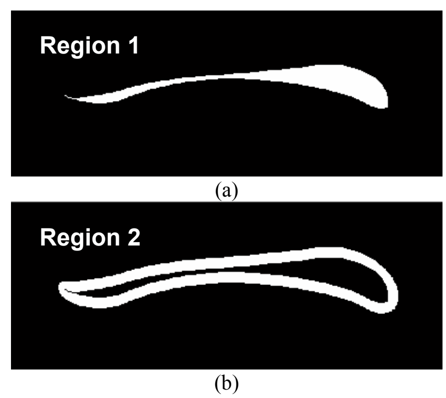



Fig.8. The sample stroke is divided into two regions in white: (a) inner region, (b) outer region, those are inside the stroke

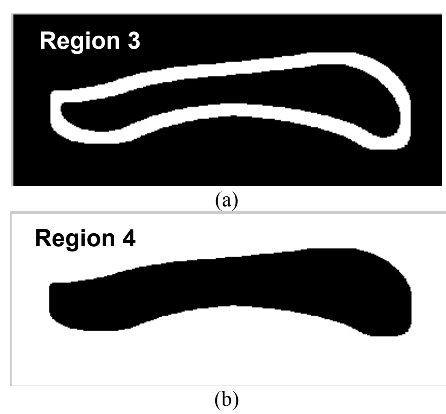



Fig.9. There are two regions outside the sample stroke in white: (a) skin region; (b) out-of-skin region

***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.9(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 labeled as region 3. The remaining region beyond region 3 as depicted in Fig.9(b) is labeled 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:

$$ObjV_2 = w_3 p'_3 + w_4 p'_4 , \quad (3)$$

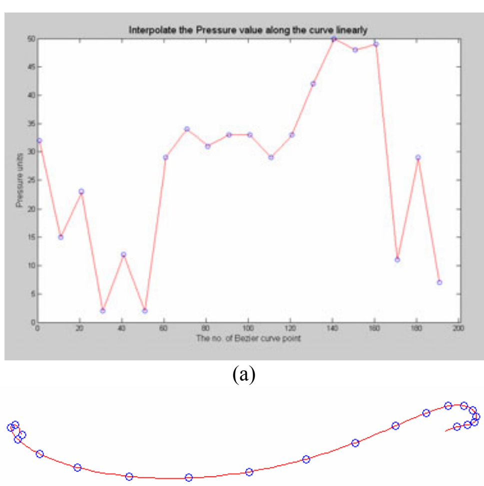
where $p'_i$ is the number of painted pixels within the white region *i*. Similarly, we set $w_4 > w_3$ to impose the desire of not painting in region 3 and even less so in region 4.

***Sub-objective 3:*** Smooth painting depth change

Fig.10(a) and (b) depicts one example of the *z*-depth values distributed along the trajectory. The actual applied depth over the whole Bézier curve is the linear interpolation over the *z*-depth values at the manipulation points. 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:

$$ObjV_3 = w_5 \sum_{i=0}^{m-2} \left| Z_{i+1} - Z_i \right| , \quad (4)$$

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




(a)

(b)

Fig.10. (a) Linear Interpolation of z-depth values along the control points; (b) 20 circles depicting the manipulation points distributed along the trajectory

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

$$ObjV(C_i) = ObjV_1(C_i) + ObjV_2(C_i) + ObjV_3(C_i) \quad (5)$$

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:** $x_i = ranking(ObjV(C_i))$, (6)

where $x_i$ is the position in the ordered population of the $i^{th}$ individual according to its objective cost in (5), $i=1,\ldots,N_{ind}$, 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:** $F(x_i) = 2 - PRS + 2(PRS-1)\dfrac{x_i - 1}{N_{ind} - 1}$. (7)

As suggested in [10], 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 stroke generation problem in the following.

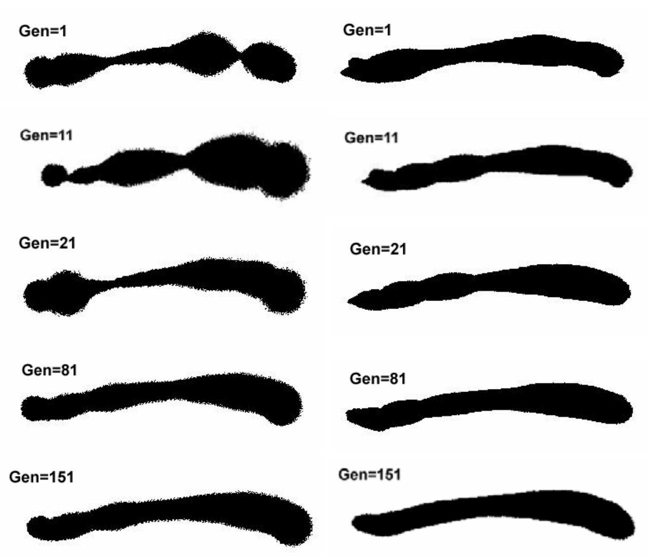
1) No. control point groups defined: *n*, the control points are grouped into line segment drawn inside the stroke manually. The control points are located on the line segment in every *inv* pixels. For example of Fig.3(b), *inv*=3 pixels.
2) No. of control points in group *i*: $n\_cp_{i=0\ldots n-1}$
3) Total no. of control points inside the stroke:
$$cpn = \sum_{i=0}^{n-1} n\_cp_i \quad (8)$$
4) No. of points interpolated over the curve: *n_pt*
5) No. of pressure level applied: There are *m* motion points and the value usually is ranging from 1 to 50, i.e., *z*-depth value are defined $[Z_{\min}, Z_{\max}]=[1, 50]$
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: $w_i=[w_1, w_2, \ldots, w_5]$
8) Chromosome length in binary coded:
$$l_{bit} = \sum_{i=1}^{m} \lceil \log_2(n\_cp_i) \rceil + n \lceil \log_2(Z_{\max} - Z_{\min} + 1) \rceil \quad (9)$$
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: $N_{gen}$
15) Initial population: randomly generated within the groups of control points and range of painting depth

## IV. Evolutionary Calligraphy Painting Results

The GA stroke generation is performed on a notebook PC with Centrino (1.6G) CPU, 512MB RAM, using Windows XP. The computation time for generating one solution is roughly 20-35mins. 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. The following results are chosen as the best upon a few trials.

The evolutionary painting results for the sample stroke of Fig.3(a) is shown in Fig.11. Fig.11(a) presents the results using the circular spray paint template of Fig.4(a). The parameters are set as: $inv$=3, $n\_pt$=200, $N_{ind}$=60, $N_{gen}$=151, $p_s$=70%, $l_{bit}$=182, $p_m$=3.8462e-3, $p_c$=0.7, $m$=23, $n$=20, $cpn$=121, [$Z_{\min}$ , $Z_{\max}$]=[1, 50], $w_i$=[3, 1, 1, 3, 3], and the computation time is 25mins 43sec. Fig.11(b) presents the same using the water drop templates of Fig.4(b). The corresponding objective cost as a function of the generation number is depicted in Fig.12. By comparing the two experimental results, it can be seen that the water drop template performed better. Some shapes of templates like the circular spray paint templates cannot paint the calligraphy stroke well in nature since they are much different from the painting footprint of real brush tuft. Fig.13 shows the best painting trajectory resulted using the water drop templates.




(a) (b)

Fig.11. Samples of results from stroke generations using GA: (a) using circular spray paint templates of Fig.4(a); (b) using water drop templates of Fig.4(b)

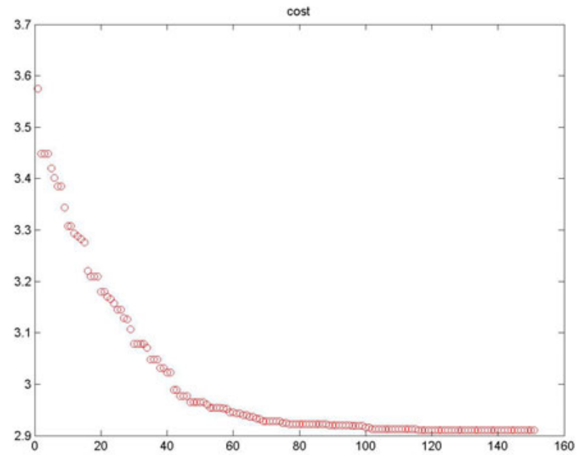



Fig.12. The objective cost (in *log* scale) of the best individual vs the number of generation past

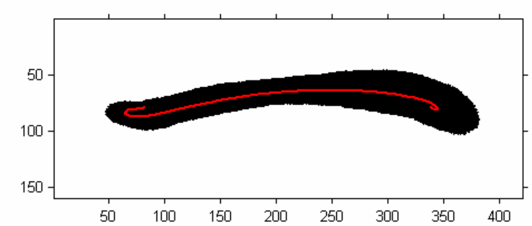
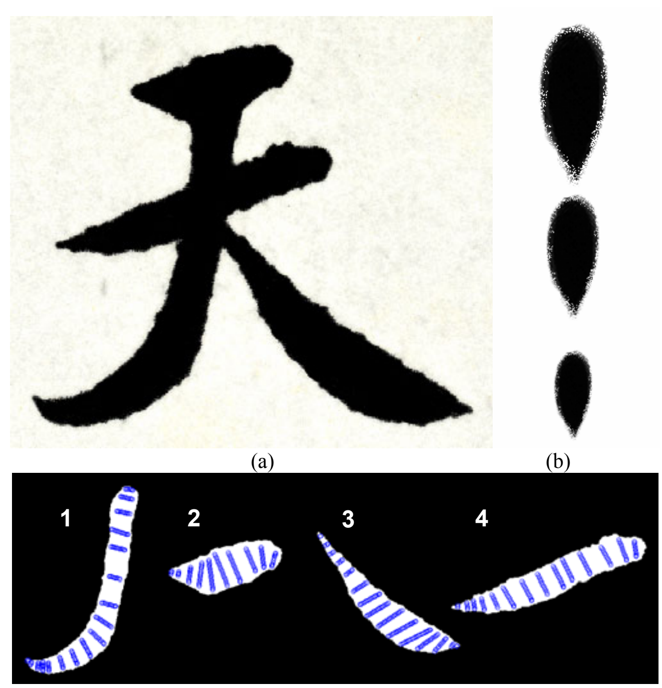


Fig.13. The best stroke trajectory under generated by GA using water drop template brush model




(a) (b)

(c)

Fig.14. (a) A 636×570 image Chinese calligraphy character "天" from Lan Ting Xu "蘭亭序" (source: [11]); (b) Water drop templates with spraying effect; (c) Corresponding control points set inside the extracted four strokes

For another demonstration, we conduct stroke generation for the Chinese character as depicted in Fig.14(a). The character “天” (means Sky) came from Wang Xianzhi (王羲之)’s well known masterpiece called Lan Ting Xu (蘭亭序). Fig.14(c) shows the four stroke elements extracted manually using image editor software. Corresponding control points in groups distributed along the individual strokes are also shown. This time around, another kind of brush templates combining the effects of two previous templates in Fig.4(a) and (b) is adopted. As shown in Fig.14(b), the new templates exhibit some spraying effects on its boundary. Fig.15 shows some results of the GA-based evolution processes for the four strokes. Finally, the evolutionary strokes recomposed to form the given character of Fig.14(a) is shown in Fig.16(a), and corresponding painting trajectories in Fig.16(b). The resulting depth values $Z_i$ are also given as follows:

**Stroke 1:** $m$=27, $Z_i$=[2, 30, 26, 32, 36, 31, 41, 32, 36, 32, 36, 32, 49, 41, 32, 39, 34, 29, 29, 18, 25, 10, 11, 11, 8, 5, 1]
**Stroke 2:** $m$=15, $Z_i$=[5, 7, 11, 12, 6, 5, 7, 7, 32, 42, 24, 24, 33, 49, 42, 46, 30, 31, 33, 26, 18, 14, 13, 5]
**Stroke 3:** $m$=20, $Z_i$=[8, 11, 11, 17, 16, 26, 28, 40, 42, 48, 48, 50, 50, 48, 30, 33, 23, 14, 8, 3]
**Stroke 4:** $m$=20, $Z_i$=[3, 16, 14, 14, 31, 32, 33, 45, 44, 45, 49, 44, 47, 46, 45, 42, 47, 40, 16, 13]

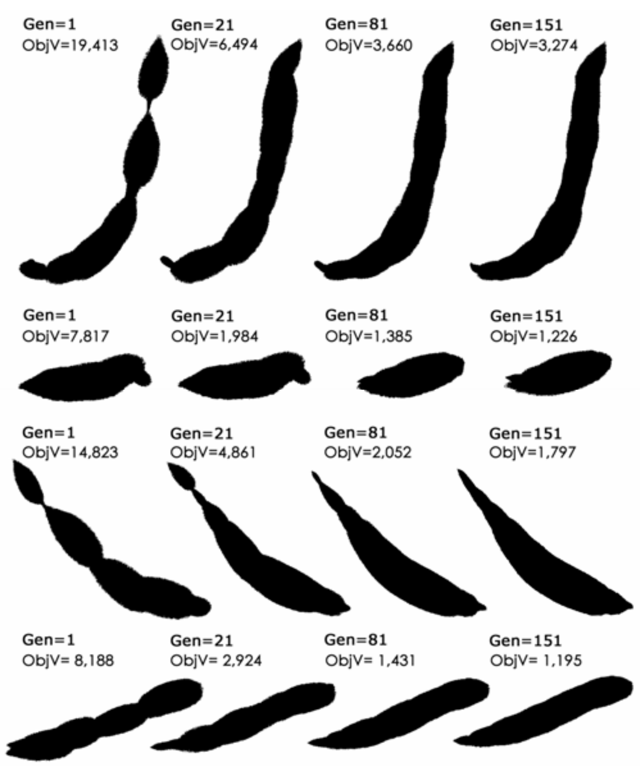


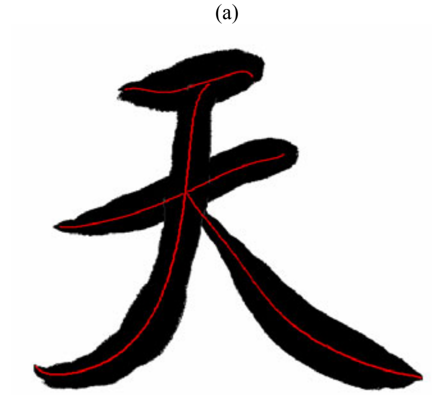

Fig.15. Evolutionary processes of four element strokes in Fig.14

## V. Stroke Painting Order Assignment By TSP Solver

When a beginner was learning to write, he/she was probably taught to write the letters in a certain order and direction. For example, it is the convention to write western languages from left to right. For Chinese characters, it is very important that the strokes are written in a certain defined order. As a baseline, a simple rule is adopted which is consistent with most Chinese calligraphy: from Up to Down and Left to Right. Other than that, proper order of execution would be extremely difficult to define by any set of rules upon a given character. In this work, we attempt to use “efficiency” as a rule to determine the stroke ordering. Efficiency is in line with our expressed intention to execute the generated strokes via a robot platform, in the sense that execution would be done in the shortest time, assuming a constant speed of execution of all strokes. We will attempt to proceed by formulating the problem at hand as a Traveling Salesman Problem (TSP).

TSP is a deceptively simple combinatorial problem. It can be stated very simply: A salesman spends his time visiting $n$ cities (or nodes) cyclically. In one tour he visits each city just once, and finishes up where he started. The problem is this: in what order should he visit the cities to minimize the would-be-distance traveled? Translating to our robot drawing problem, the robot should be spending least time to draw the $n$ strokes, and returning to the homing position near the first stroke after completing its drawing. In contrast to TSP, however, the strokes here are not zero dimensional nodes, but one dimensional length with *entrance* and *exit* points. That implies in this case the distance matrix is not symmetric.

(a)



(b)

Fig.16. (a) The calligraphy character recomposed using the four GA-generated strokes; (b) The painting trajectories as resulted from the process

Fig.17. A 875×1011 image Chinese calligraphy character "龍" from Kō Fukuji Dampi "興福寺斷碑", (source: [14])

In the present work, we use a TSP solver called the Cross-Entropy Method (CE-Method), which is relatively new method for estimating rare-event probabilities (see [12]-[13]). In our mechanism, the stroke distance are defined such that $d_{ij} = \left| p^{i}_{end} - p^{j}_{start} \right|^{1/2}$ is the shortest Cartesian distance from the end point $p^{i}_{end}$ of stroke $i$ to the starting point $p^{j}_{start}$ of stroke $j$ on the drawing plane. For example, given a tour path of stroke sequence 5→4→6→7, the cost of the tour distance would be $T_{dist}=d_{54}+d_{46}+d_{67}$.


Fig.18. The character "龍" recomposed with strokes generated from beginning of evolution process

To demonstrate our results, Fig.17 shows another more complicated calligraphic character "龍" (means Dragon) by Wang Xianzhi taken from the Kōfukuji Dampi (興福寺斷碑). The character is decomposed manually into 14 strokes, which all undergo the GA-based evolution process. Fig.18 shows the recomposed character formed by the 14 strokes generated from the first generation of the evolution process. The evaluated objective cost is 357,975. Upon 151 generations later, the objective cost is reduced to 6,224 and the corresponding calligraphy stroke results are shown in Fig.19. Now, continuing with the CE-Method, we set the number of sample to generate at each round as $N$=1000, the fraction of best samples to take as *rho*=0.05, and the smoothing parameter of *alpha*=0.8. Moreover, node placement setting is adopted. The stroke execution sequences as resulted is depicted as Fig.20, with the "x" and the "o", respectively, indicate the start and the end points. The tour cost is reduced to $T_{dist}$ =815 pixels. For comparison, Fig.21 depicts the same but according to traditional and accepted stroke ordering of the character. The corresponding cost $T_{dist}$ =1,120 pixels.


Fig.19. The character "龍" recomposed with strokes obtained after 151 generations of evolution process

## VI. Conclusion

Good painting trajectory and stroke thickness control are essential for high artistic values in Chinese calligraphy. In this paper, a novel scheme for stroke generation using GA is presented. We model the brush stroke as composed of 2D brush template along a painting trajectory described by Bézier curve. Depending on the z-depth values, templates of different sizes would be applied along the brush trajectory the union of which would form the stroke. The application GA-based algorithm for stroke generation enhances the flexibility and freedom for matching a given stroke sample through the use of objective function, rather than having to analyze the detail geometric information. The results as produced and shown in this work are promising and indicate that GA is able to generate a full stroke with the adoption of appropriate brush template. The given calligraphy can be replicated quite closely in our simulation. The present work also tackled the problem of stroke ordering for future calligraphic execution by a robot platform. We formulate the problem as a TSP and apply the CE-Method to yield the appropriate stroke drawing sequence. Simulation results are presented with a rather complicated Chinese character “龍”. The present work contributes to our eventual objective in automatic analysis, decomposition, stroke trajectory generation, execution sequence determination, and robot replication of Chinese calligraphy, and ultimately, Chinese painting.


## ACKNOWLEDGMENT

This research was supported by the Hong Kong Research Grant Council under CUHK Direct Grant 2050317. The authors would like to thank Professor Kwong-Sak Leung for his views on the GA part of the work. We are also grateful to the reviewers whose comments have been most useful and encouraging.


## REFERENCES


[1] Siu-Leung Lee, “Asiawind Art Gallery 亞洲風畫廊”, [Online]. Available: http://www.asiawind.com/art/callig/Default.htm

[2] Nelson S.-H. Chu and C.-L. Tai, “An Efficient Brush Model for Physically-Based 3D Painting”, *Proc. of Pacific Graphics 2002, Oct. 9-11, Beijing, China, IEEE Press.*

[3] 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.

[4] Lee, J., “Diffusion Rendering of Black Ink Painting using New Paper and Ink Models,” *Computer and Graphics 25*, 2, pp. 295-308, 2001

[5] Nelson S.-H. Chu and C.-L. Tai, “MoXi: Real-Time Ink Dispersion in Absorbent Paper,” *ACM Transactions on Graphics (SIGGRAPH 2005 issue)*, August 2005, Vol. 24, No. 3

[6] Songhua Xu, Francis C.M. Lau, William K. Cheung, Yunhe Pan, “Automatic Generation of Artistic Chinese Calligraphy,” *IEEE Computer Society*, 2005, pp. 32-39.

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

[8] Yam, Y, Lo, K.W, Kwok, K.W., “A Robot Drawing System: Preliminary Design and Demonstration,” Proceedings of the *2003 International Conference on Intelligent Technologies, Chiang Mai, Thailand*, Dec. 17-19, 2003, pp. 545-551.

[9] Zhang, Y.Y.; Wang, P.S.P., “A parallel thinning algorithm with two-subiteration that generates one-pixel-wide skeletons”, *Pattern Recognition, 1996.*, Proceedings of *the 13th International Conference on* , Vol . 4 , 25-29 Aug. 1996, pp. 457 – 461

[10] J. E. Baker, “Adaptive Selection Methods for Genetic Algorithm”, *Proc. ICGA 1*, 1985, pp. 101-111.

[11] 墓誌銘 / [解說者中田勇次郎], **“**書道藝術”, 東京 Dongjing : 二玄社 Er xuan she, 1967

[12] D.P. Kroese and R.Y. Rubinstein., *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization*, Monte-Carlo Simulation and Machine Learning, Springer, 2004.

[13] P.T. de Boer, D.P. Kroese, S. Mannor, R.Y. Rubinstein. (2003, Nov.). A Tutorial on the Cross-Entropy Method. [Online]. Israel Institute of Technology. Available: http://iew3.technion.ac.il/CE/tutor.php

[14] 王羲之 Wang Xianzhi, “興福寺斷碑 KōFukuji Dampi : 行書 Gysho*”*, 東京 Dongjing : 二玄社 Er Xuan She, 1986

[15] A. J. Chipperfield, P. J. Fleming, H. Pohlheim and C. M. Fonseca, *“Genetic Algorithm Toolbox User's Guide”*, ACSE Research Report No. 512, University of Sheffield, 1994.

[16] Chen S Y and Li Y F, "Automatic Sensor Placement for Model-Based Robot Vision", *IEEE Transactions on System Man and Cybernetics, Part B*, USA, Vol.34, No.1, Feb 2004, pp. 393-408.

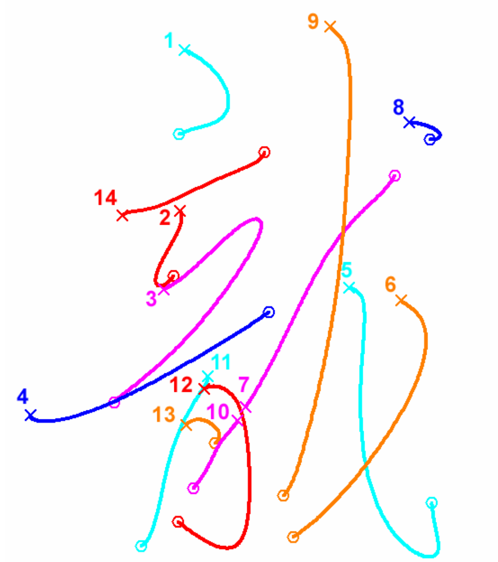




Fig.20. Stroke ordering for the character “龍” obtained from TSP solver -- CE-Method

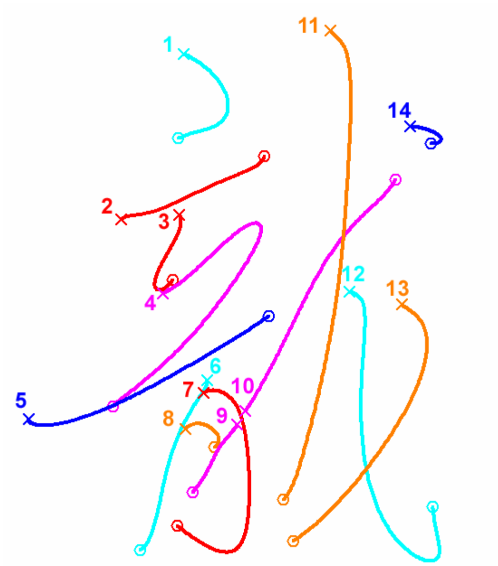



Fig.21. Traditional and established stroke ordering for the character “龍”