Visual-based Decision for Iterative Quality Enhancement in Robot Drawing

by

Kwok, Ka Wai

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Master of Philosophy

In

Automation and Computer-Aided Engineering

©The Chinese University of Hong Kong

June 2005

ABSTRACT

The advent of new technology inevitably filtered into various realm of human society, including art. In time, new technology induces new horizon for artists to spread their talents, thus giving rise to new forms of art. The rapid advance in robotics and computing technology is no different.

In this research, we apply artificial intelligence and computer vision techniques for robot drawing. We use robot drawing platform supporting five degrees of freedom (x, y, and z translation, z-rotation, and pitch) of a brush-pen motion in our laboratory for the study. The degrees are independently commanded to do away with the kinematics problems associated with many other manipulator-based drawing systems. The platform is aimed at both the replication of existing works and rendition of new Chinese painting and calligraphy. This thesis is specially focusesd on visual-based capabilities for drawing quality enhancement and artistry imitation. For vision system, high accuracy projective rectification is implemented automatically via Genetic Algorithms-based (GA-based) homography transformation, which allows the readily comparison between the original line drawing and the robot executed drawing. Such comparison can be used to determine corrective drawing actions for iterative improved drawing. To start the process, a drawing plan is first generated by extracting the line stroke using vectorization and Bezier curve interpolation techniques.

In additional to line stroke drawing, we have also carried out simulations and experiments on GA-based full stroke generation and real brush stroke characterization to demostrate the effectiveness of our proposed approach. Our

results show that iterative drawing process is a novel approach for robot to learn human artistry and practice on its own.

摘要

機器人學與計算機科技的發展一日千里，新科技的出現無可避免地滲透到社會不同領域，包括藝術。假以時日，新科技將爲藝術者帶來一個新的領域，以展示他們的才能，同時帶出新的藝術模式。

在本研究中,我們應用人工智能技術及計算機視覺技術來進行機器人繪畫。而我們在實驗室所開發的機器人繪圖平台能支援筆桿五個軸的運動 (x, y, z 軸的移動, z 軸的旋轉, 與橫向扭動)，每個自由度均直接及獨立地執行，平台式的五軸運動避免了很多複雜的動力學運算如其他機械手型的繪畫系統。此平台是以研究中國書畫爲目標的機器人，以設計用作仿製現有的及新穎的藝術作品。此論文會描述我們的繪畫系統，透過視覺功能裝置以提高繪畫質素及其藝術效果的仿真程度。在這視覺系統，由基因演算法爲基礎之對應距陣變換所自動履行的高準確度透視投影轉換，提供了原線條圖畫與機器人所繪畫的效對。透過這視覺訊息的運用，繪畫的糾正動作能得以測定並用作反覆性的繪畫改良。爲提高其效果，書畫的方案設計以先，將會利用貝氏曲線插補法向量化來抽取其線條筆觸。

除線條繪畫之外，以基因演算法爲基礎的粗筆觸衍生及真實毛筆筆觸的參數化將會分別由模擬和實驗所履行，並展示其建議方法的成效。總括而言，我們的實驗示範了反覆繪畫程序是新穎的門徑去仿真機器人透過自我訓練來學習人類藝術技巧。

ACKNOWLEDGEMENTS

I would like to express my most sincere thanks to my advisor, Prof. Yam, Yeung, for all his help during the past years. His guidance and encouragement were vital elements in the completion of this work.

Moreover, I would also like to thank the thesis examiners Prof. Xu, Yang-Sheng, Prof. Chung, Chi-Kit Ronald and Prof. Ma, Wei-Yin as well as other faculty members for their patient instruction throughout my academic study at the Chinese University of Hong Kong.

Besides, I would like to express greatly thanks to Lo Ka-Wah, who is the pioneer of the robot drawing motion platform, and other close colleagues: Dr. Yang Chi-Tin, Chiu Kit-Chow, Lei Kin-Fung, Wong Sheung Man, Tse Kim-Fung, Yip Wai Lee and Lee Wai Man, for supporting me in all these years. And I would like also to thank Simon Tong, Raymond Lam and Ng Yiu Ming for all their love during my study at the Chinese University of Hong Kong.

Finally, I would like to thank my family for their persistent encouragement and support.

Ka-Wai Kwok

Contents

List of figures

1. INTRODUCTION

Progress in technology inevitably permeates into the expression of art form. One example is the arrival of photography in the late 19th century, which helped to promote impressionism as a painting style transcending over mere duplication of the world. Today, The robot technology is the same. With the rapid advancement of computer and robot-related engineering, there has been a host of works on robot paintings, e.g., [1]-[10]. Based on methodologies such as artificial intelligence, genetic algorithms, fuzzy rules, and expert systems, specially designed robots are built capable of exhibiting various painting styles and forms, and to some extent, creating new art as well. Some of the art works are so impressive that they are being displayed in museums.

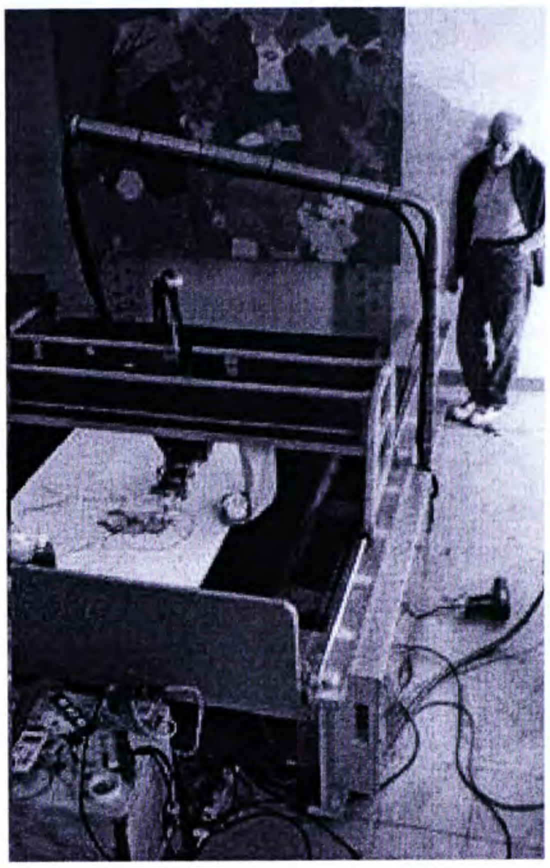


Figure 1.1. The robot painter Aaron (source: [2])

1.1 Artistic robot in western art

The first well known case of robot drawing is the robot painter Aaron pioneered by Harold Cohen [2], see Figure 1.1. Following that, numerous systems have been constructed [3]-[4], etc. there is now a yearly show of artistic robot (ARTBOT) [5], featuring drawing systems which come with different sizes and shapes; some as big as a printing press, or as small as a toy car [6]; some use a x-y plotter to draw, some operate based on music [7], or just with a pen attached to a canti-levered beam [8];

and some are more successful than others. Almost all of these reported drawing robots, however, aim for Western style of drawing, and are designed to create new art only. There was no attempt to replicate an existing painting or to imitate the drawing techniques of an artist, except for [2], which worked on the imitation part somewhat. Rigorously, speaking, these previous robots are merely programmed to draw as according to their particular designs. There was no learning of human skill in their operations.

Previous works on

1.2 Chinese calligraphy ~~robot~~

The drawing robots reported so far are mostly focused on the free style drawing of Western art. However, the configuration of Chinese calligraphy robot has been designed before. Guoliang Tao et. la. [11] developed 3-R pneumatic-servo calligraphy robot and employed the continuous trajectory tracking control to electro-pneumatic-servo system. The robot is just able to track the hard-coded trajectory of the Chinese calligraphy under the control strategy from the mathematical model of electro-pneumatic-servo system and friction. In a recent work, Fenghui Yao et. la. [12] has developed a Chinese character calligraphy robot which is categorized as an art robot. The system consists of a pre-defined calligraphy dictionary of five styles only. However, the system does not allow users to input his/her own writing style. Moreover, since the robot manipulator employs an open-loop control strategy, trajectory tracking error was not easily compensated under the resulting inverse kinematics problem.

objective of the present work

1.3 ~~Our robot drawing system~~

By contrast, our robot platform constructed with totally 5 degrees of freedom of the brush pen for the full emulation of hand and wrist movement, which are independently commanded, doing away with the kinematics problems associated with many other robot-based drawing systems such as [12] and the closed-loop PID motor control is employed for producing accurate trajectory with high repeatability [13]-[14]. Also, Industrial grade components are used to achieve the high precision and repeatability need for the fine execution of brush strokes. Furthermore, to participate in Chinese culture heritage, the present system is developed with two objectives in mind. The first is that it aims at the execution of Chinese art, including calligraphy and painting. One needs to be pointed out here is that painting and calligraphy go naturally together in Chinese artistry. The spirit of a good Chinese painting is always accentuated if accompanied by a good poem expressed in good calligraphy. The second is that the machine will be applied to investigate human skill acquisition of drawing techniques – we will attempt both the replication of selected existing pieces and also the rendition of new styles and art work. To this end, we desire a visual-based system capable of evaluating the executed robot drawing with the human art work and giving the correct action for the next execution. This iterative drawing process seems that the robot learns human artistry by the practice on its own.

What do you want to achieve in this work?

1.4 Thesis outline

In what follows, Chapter 2 gives a detail description of the Robot Drawing system implemented to imitate the human artistry.

Chapter 3 then introduces the raster-to-vector conversion for line drawing. Line stroke extraction is preformed using skeleton-based vectorization and Bezier curve interpolation are developed. The algorithm is based on eliminating the distortion at the junctions of the skeleton. After extraction, the stroke drawing orders would be assigned by a TSP solver (CE-Method) for fast effective drawing.

The vision capabilities are presented in Chapter 4. The camera system serves to off-line monitor the executed drawing. The system allows the captured image to be rectified in a full plane view as the original input image so that the camera needs not to be one of sight vertical to the drawing plane. This high accuracy rectification is implemented automatically via GA-based homography transformation.

Thus, the rectified image of the executed drawing acts as a vision feedback to improve the drawing scheme designed in chapter 3. Chapter 5 describes the iterative line drawing process together with demonstrations such that the executed drawing contour becomes closer to the input-art-piece.

For future exploration in visual-based iterative full stroke painting, Chapter 6 and Chapter 7, respectively, present a novel calligraphy stroke replication using GA and a preliminary study on experimental brush stroke "footprint" model. A variety of stroke styles and forms are simulated by GA stroke generation based on a given simple 2D brush template model. A camera is installed looking up strictly under the transparent drawing plane and serves to capture the footprint image. The footprint images are then characterized upon certain measures from which we hope to generate the relationship between the robot commands and the resulting footprint for a specific brush.

Finally, conclusions and some recommendations for future work are given in Chapter 8.

2. ROBOT DRAWING SYSTEM

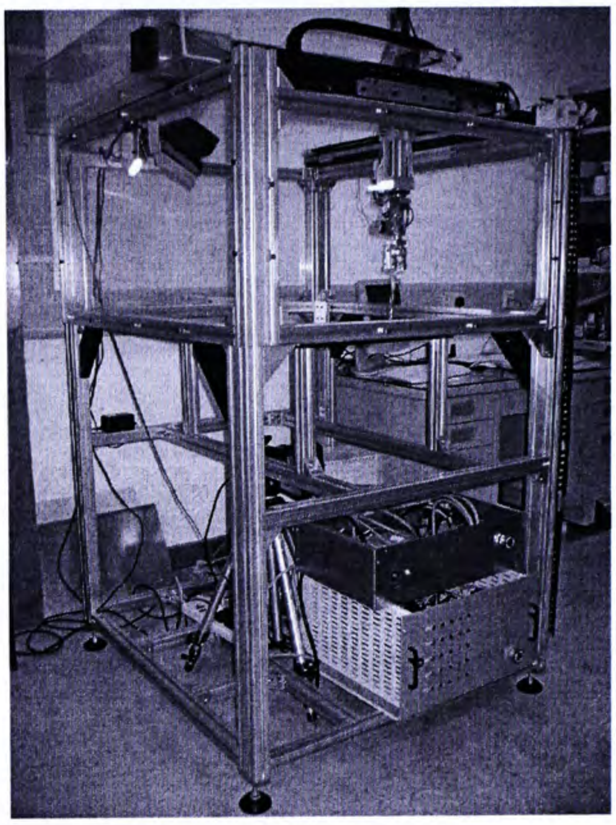


Figure 2.1. Hardware design of the Drawing Robot

This chapter describes the Robot drawing platform developed in our laboratory aims towards Chinese painting and calligraphy. The platform utilized industrial grade components to achieve the high precision and repeatability needed for the fine execution of brush strokes. The eventual goal of the platform will be on the acquisition, learning, and execution of human techniques in Chinese brush pen painting and calligraphy. In hardware construction, base on the existing drawing platform, I contribute mainly towards the implementation of visual capabilities and footprint characterization setup.

2.1 Robot drawing manipulation

The platform as developed consists of a x-y-z axis translational mechanism housing a robot gripper with a z-axis rotation and a pitching degree of freedom, making a total of 5-axis degrees of freedom for the pen movement, see Figure 2.1. The x and y-translation are executed by two AC servomotors each with an angular torque of 0.51 Nm and a length of travel of 1m. The corresponding accuracy is ± 0.001 mm. The z-axis AC motor has an angular torque of 0.08 Nm, and a vertical support load of 50 N to provide the needed support for the specially designed robot gripper. The z-axis stroke length is 0.15 m and the accuracy is ± 0.03 mm. For safety purpose, limit switches are installed on all three axes. The overall dimension of the setup is 1.1 m by 0.96 m by 1.9 m, with a drawing size of up to 0.8 m by 0.7 m. The gripper design, as shown in Figure 2.2, embeds two more degree of freedom, the z-axis rotation and a pitching motion. These two degrees of freedom are essential to emulate hand wrist motions usually employed during Chinese painting and calligraphy. All the 5 degrees of freedom commands are forwarded to the motion controller of the drawing platform for execution. The motion driver performs the execution through the implementation of a PID type controller.

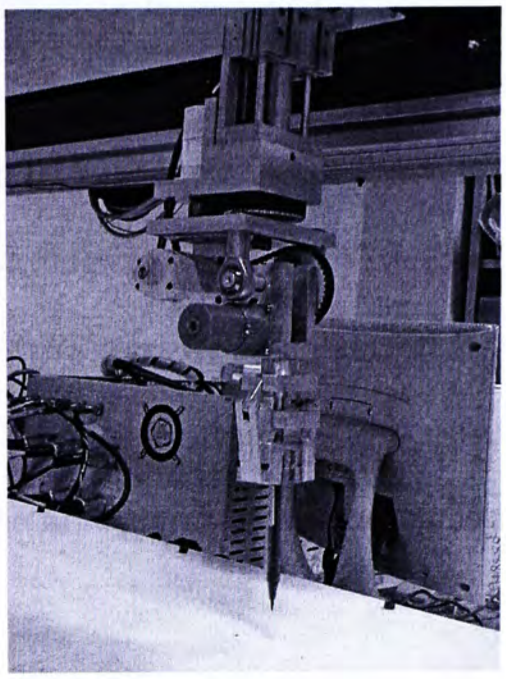


Figure 2.2. Gripper design of drawing robot system

2.2 Input modes

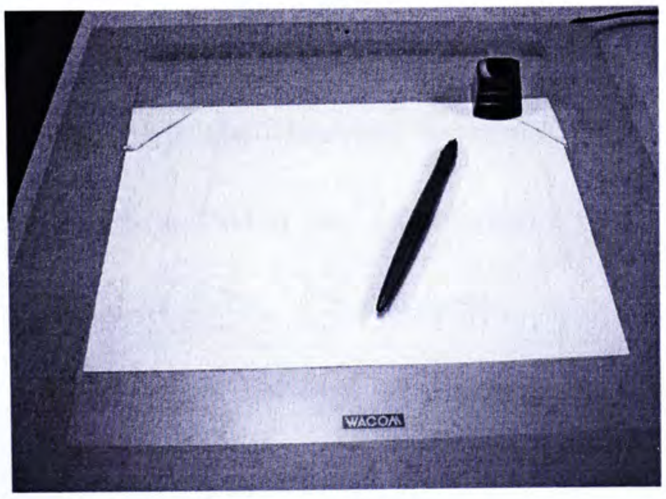


Figure 2.3. The writing tablet and pen system

The drawing platform accommodates two possible ways to input the to-be-replicated drawing. One is by direct hand drawing on a writing tablet, and the other is through a digital image file.

For the hand drawing, we use the *Intuos*^{c}2 12 by 12 Tablet and pen system as shown in Figure 2.3 to capture the drawing and writing motion of a user. As he/she starts to draw or write on the Tablet, the pen's time-tagged positions relative to a reference point are recorded and stored. The maximum rate of recording can be increased to once per 5ms. The data of recording include the x, y, z-motion, the angles of tilt, as well as some measurements of the pressure being exerted onto the pad. The recorded data can then be applied to conduct human skill acquisition [15]-[16], and analyzed to extract *trademarks* of the drawing style and technique of the person.

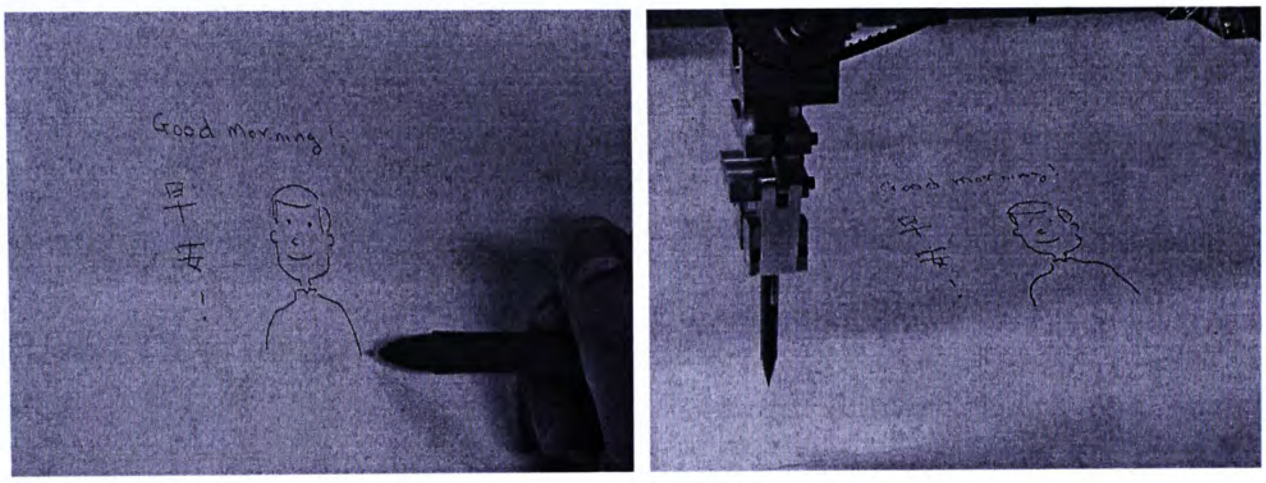


Figure 2.4. (a) Hand writing and drawing on Tablet, (b) Executed writing and drawing by our system

Figure 2.4 (a) and (b) compare, respectively, the hand writing on the Tablet and the corresponding version executed by the drawing system. Furthermore, Figure 2.5 shows the hand calligraphy of a user as executed by the drawing system. This illustrates the "what you draw is what you get" kind of ability in our system.

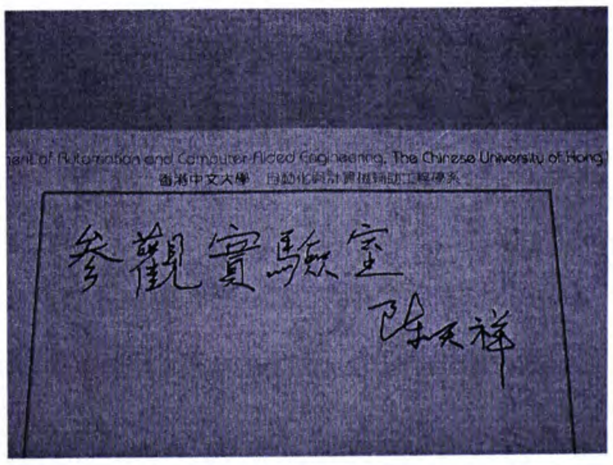


Figure 2.5. Robot execution of Chinese calligraphy captured by writing pad

The other method of feeding in a raster pixel image would be our focus in this thesis. A raster-to-vector conversion is then conducted. All the line strokes would be extracted under the vectorization information and the stroke drawing order would be assigned by a Traveling Salesman Problem (TSP) solver for fast effective execution. These are described in chapter 3.

2.3 Visual-feedback system

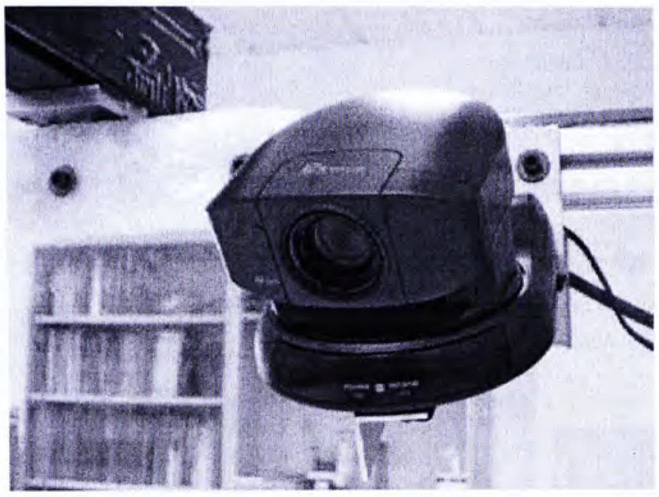


Figure 2.6. Camera looking down at drawing area

Figure 2.6 shows the installed camera system looking down at the drawing area at an angle of roughly 30 degrees. The system uses a Sony EVI-D30/D31 Pan/Tilt/Zoom Color Video camera. The camera system serves to monitor the executed drawing on the drawing board and generate corrective actions upon comparing the executed drawing with the original image. It includes rectification of an angled image captured by the camera to a full plane view. Rectification is conducted using homography matrix generated in which the errors were compensated by a Genetic Algorithm (GA) -based upon initialization of 4 selected correspondence points. This will be described in chapter 4.

2.4 Footprint study setup

The robot platform of Figure 2.1 also incorporates a setup to study the actual dynamics and footprints of a brush pen. The basic idea is shown in Figure 2.7. 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 caputured footprints are then be correlated with the input commands for non-parametric characterization.

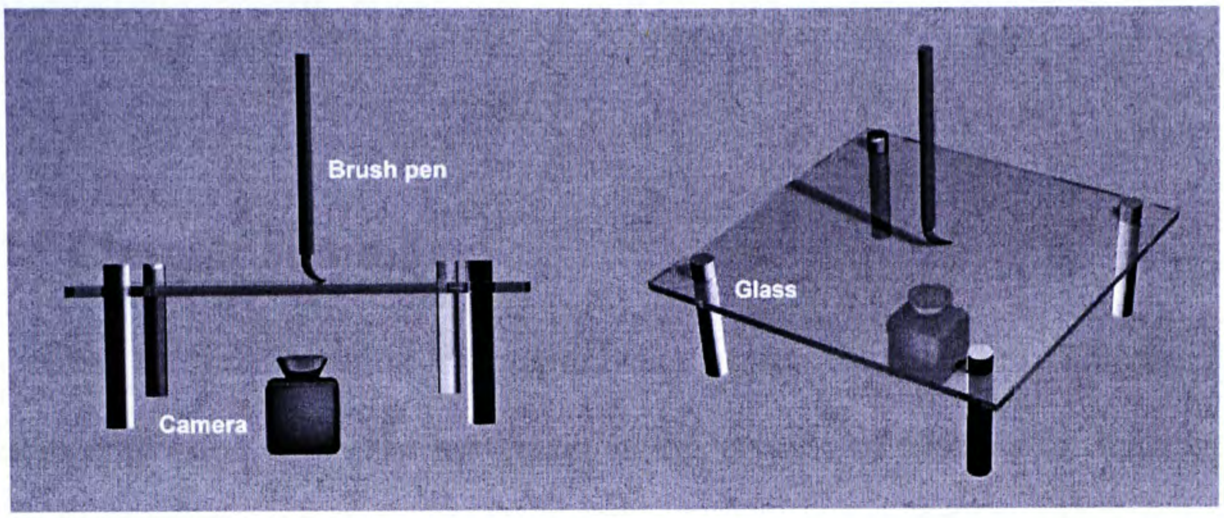


Figure 2.7. Conceptual idea and hardware for footprint acquisition

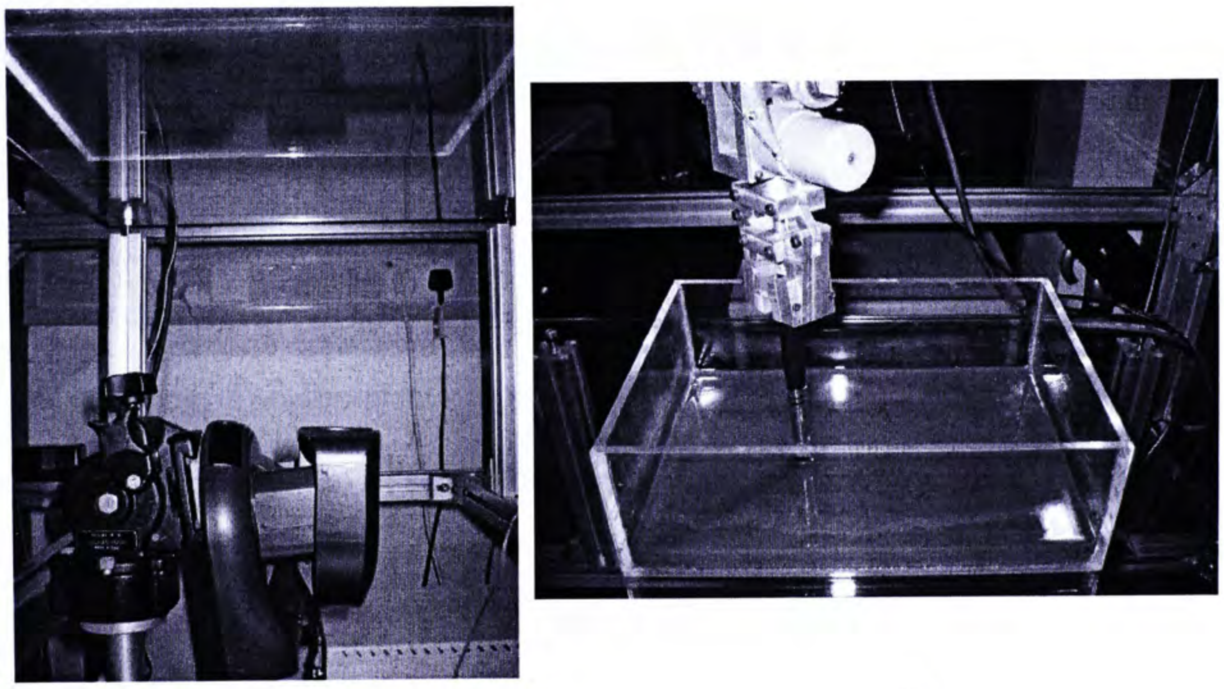


Figure 2.8. Actual hardware for footprint acquisition: (a) Video camera, (b) the transparent drawing setup

The same camera model as in Figure 2.6 is adopted here and is installed looking up strictly vertical under the transparent drawing plane, see Figure 2.8(a). Figure 2.8(b) shows that the robot in this case is drawing on the bottom of a flat transparent container filled with a blue liquid. The blue liquid would act as the background of the video captured, providing a good contrast to the yellow brush tuft that constitutes the footprint. Our goal is to determine the brush model experimentally, i.e., footprint, as a function of robot movement. This part will be also described in chapter 7.

2.5 Chapter summary

Here, the configuration of our drawing system is described. For the full emulation of hand and wrist movement in the process of Chinese art making, 5 degrees of freedom pen motion are supported by using the high precision industrial grade components. Two input modes for art imitation: digital image input and direct hand drawing/writing input are introduced. For iterative drawing process, the addition of the camera system and the incorporation of visual-based capabilities to the platform are both implemented as the visual-feedback system. Furthermore, we would like to extent the techniques of line drawing quality enhancement to full stroke painting. The footprint study setup provides an appropriate environment to investigate the formation of stroke under the brush pen motion. Finally, the eventual goal of the system will be on the acquisition, learning, and execution of human techniques in Chinese brush pen painting and calligraphy.

3. LINE STROKE EXTRACTION AND ORDER ASSIGNMENT

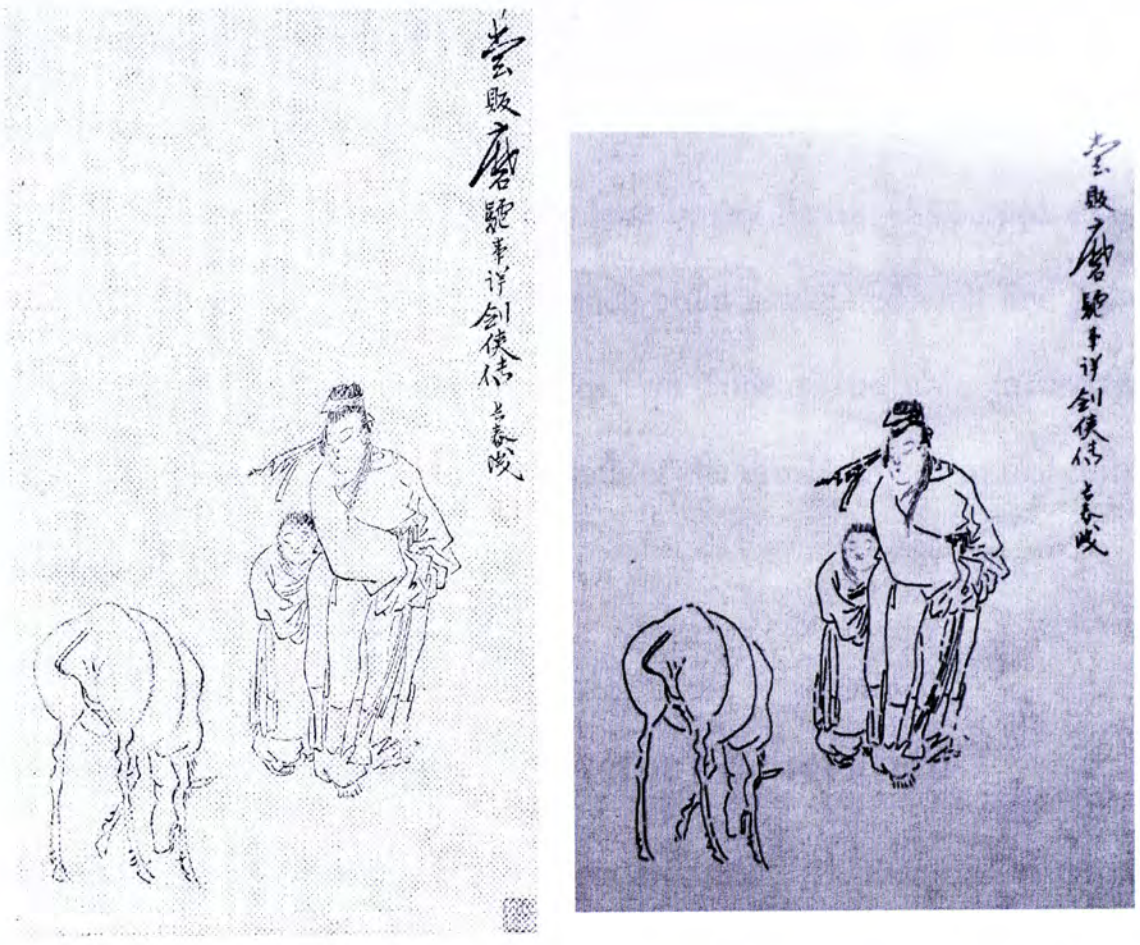


Figure 3.1. (a) Original Chinese ink line drawing from famous ancient artist "穌仁山" Su Renshan, (b) Replicated with downward offsetting of inscription done by image editing

Line drawing replication is utilized as starting point for demonstration of vision capabilities. The fundamental component of Chinese painting, as in Chinese calligraphy is the line. Due to this shared feature, these two arts have had a close mutual relationship since the beginning of time. Line has been used in many ways throughout art history, not only with contour drawing, but also with variations of the contour [17]. Van Gogh also did many ink drawings using line and used it in the form of variously shaped marks rather than limiting his use of line to contour, which served to create a textural effect, as well as to help delineate spatial depth. Here, we would like to choose some line drawings that the width of strokes forming the picture is quite uniform and rather thin. This kind of artistry can be found in many

Chinese ink drawing, i.e. Figure 3.1(a). This contrasts to those with full brush strokes of thick and varying width, which would require pen motion control other than (*x, y*) degrees of freedom. Thus, the line trajectories and their order of execution are only expressed in the form of (*x*, *y*) coordinate sequences. Figure 3.1(b) shows the robot replication art piece of a Chinese line drawing Figure 3.1(a).

Hence, visual-based corrective actions later in this thesis will be processed for improved performance on the aspect of branch point associated with line drawing. Eventually, with the full brush capabilities, we hope to be able to use visual feedback to correct the thickness and line path of the brush strokes in real time and online in the future.

3.1 Skeleton-based line trajectory generation

The to-be-replicated art piece corresponding image file then passes through a Matlab-based library containing user-designed algorithms to analyze the input data and to extract feature points and lines of the image. The process involves raster to vector conversion, which include thresholding to grayscale image [18], image noise reduction filter [19], skeletonization [20] and thinning algorithm [21]. The last step is one of the most important preprocessing steps for feature extraction on many pattern recognition systems. The above process results in one-pixel-wide skeleton of the image foreground. For the present case of line calligraphy, the resulting skeletons would follow more or less the middle of the line segments, giving rise to the trajectories in line calligraphy.

Assignment of the drawing sequence is then generated. As a baseline, a simple rule is adopted which is consistent with most Chinese calligraphy: from Up to Down and Left to Right. After that, the Preorder Traversal Point sequence as described below is used. Figure 3.2 gives an example of the above application:

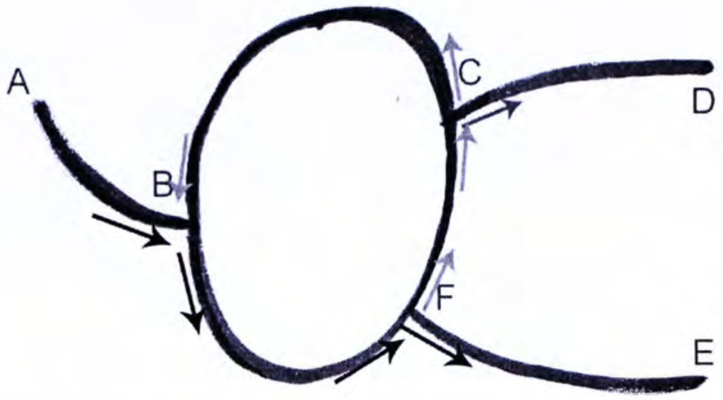


Figure 3.2. Preorder traversal sequence (Edges: A, D, E. Branches: B, F, C)

- The edge A is first chosen as the starting point (root) of the graph
- 1st line drawing segment: A-B-F-E. The branch list = {F,B} (the last passing branch placed at the head of the list)
- 2nd line drawing segment: F-C-B (start the point from the head element of the branch list). The branch list = {C} (Delete F and B from the list after all possible ways of those branch are visited)
- 3rd line drawing segment: C-D. The branch list = {}
- If the branch list is empty, that means all connected skeleton in the graph are visit, then try to start from the edge which locates at the left-top side of another graph.
- Until all graphs in the same picture are visited, the drawing/writing plan would be finished.

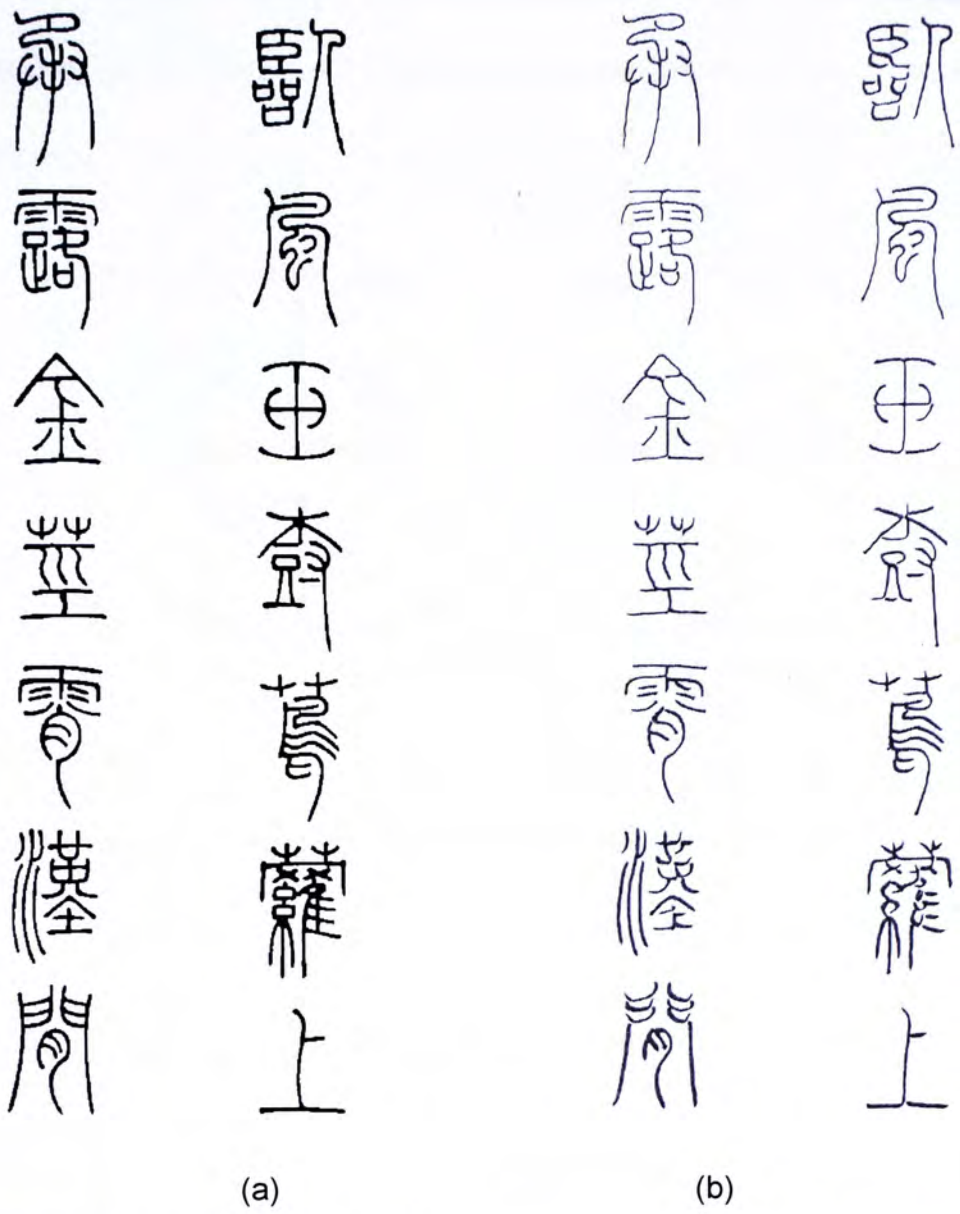


Figure 3.3. (a) The original calligraphy in ancient Seal Character "篆書", (b) The corresponding writing execution

Figure 3.3(b) shows an example of the trajectory generation using preorder traversal point sequence and its original calligraphy is shown in Figure 3.3(a). One can see that the generated strokes are not all consistent to human drawing and writing sequences. Actually, proper order of execution would be extremely difficult to set by any rule especially in free style Chinese calligraphy and painting. In Chinese artistry, improper drawing and writing sequences will highly affect the quality of executed work. One such example is depicted in Figure 3.4(b), which corresponds to the third character of the right column in Figure 3.3(b). Note that brush tip deformation during the drawing process yields rather significant offset in the brush lines which may be used to indicate that an improper drawing sequence has been executed by comparing with Figure 3.4(a). The line stroke extraction capabilities to be incorporated later aim at this problem.

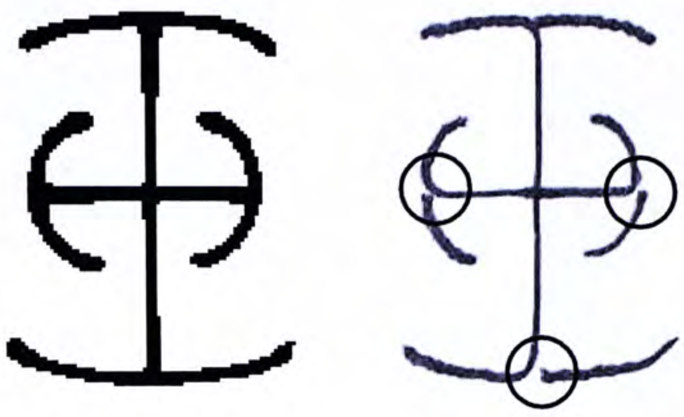

(a) (b)

Figure 3.4. (a) The original Chinese seal character, (b) The robot writing executed under improper stroke sequence and those unconnected junctions are encircled

3.2 Line stroke vectorization

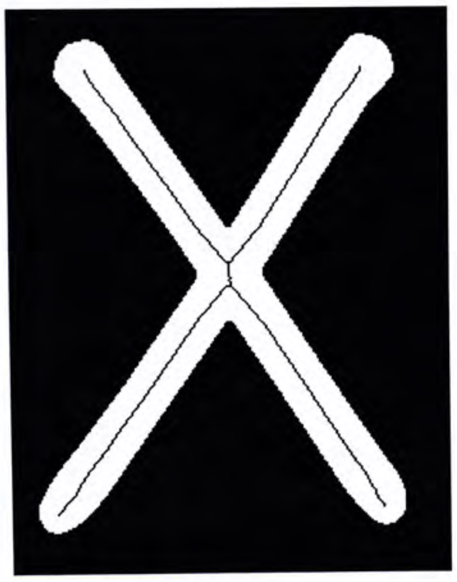


Figure 3.5. The cross strokes raster image containing its skeleton

Vectorization [23] (raster-to-vector conversion) consists of analyzing a raster image and converting its pixel representation to a vector representation. The basic assumption is that a vector representation is more suitable for the interpretation of the image, which typically is a scanned graphical document (map, scheme, technical or construction drawing). In this section, we present the algorithms for automatic transformation of raster images into the vector that gears toward the line stroke extraction. Such vector format would give important information for directional strokes detection as well.

However, obtaining reliable measures of skeleton tangential slope at a point in a thinned digital image is often difficult because these skeleton images tend to be locally ragged. Generally speaking, it is difficult to approximate the slope at a point by the discrete skeleton pixels without proper smoothing [24]. *e.g.* $(y_{i+k} - y_{i-k})/(x_{i+k} - x_{i-k})$ for some smoothing factor, $k > 1$. The choice of the value k is problem-oriented and thus can not be determined a priori. We adopt some of the ideas from in which Chiang et. la., [25], who proposed a region-based method to recognize straight lines from images directly. The method is to use the Maximum Inscribed Circle (MIC) to detect the characteristic of straight line. It recognizes and extracts straight lines with junctions directly to eliminate the distortions at junctions. In our case, MIC is formed inside the strokes along the skeleton pixels to determine the characteristic of the corresponding slopes. This is described in the following subsections.

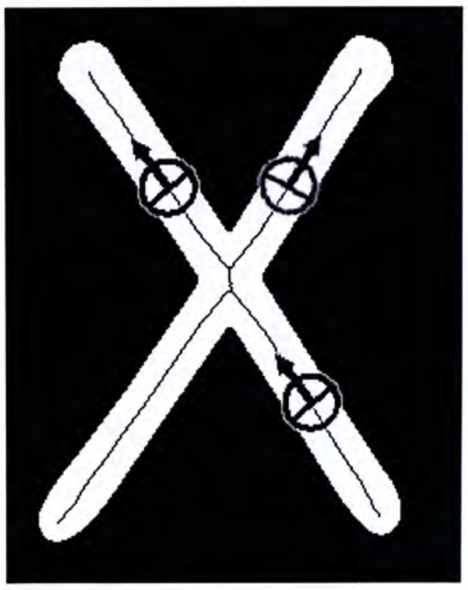
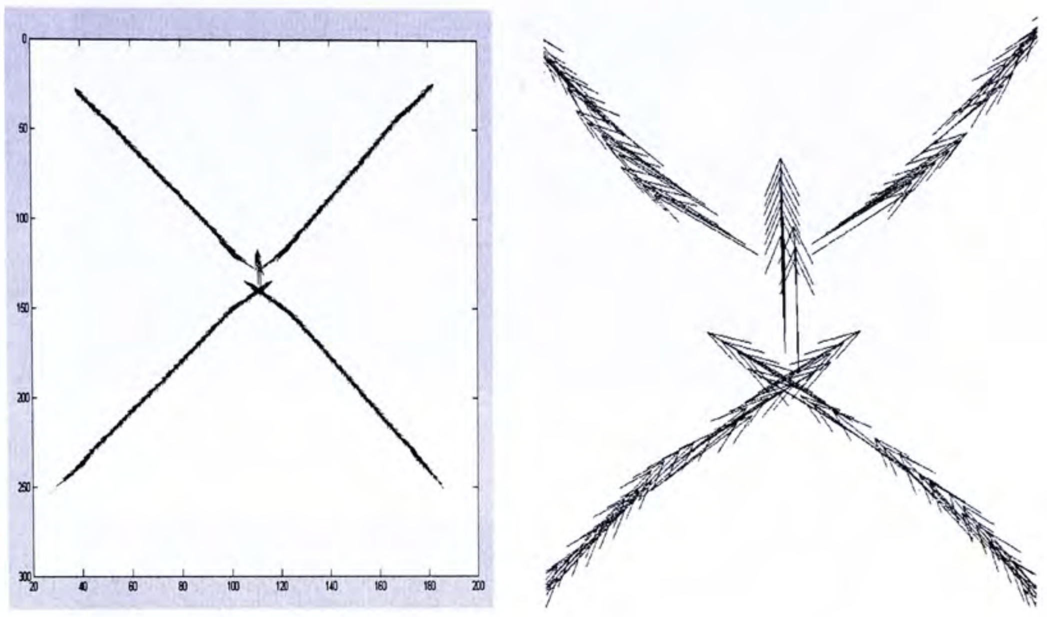


Figure 3.6. MIC forms inside the stroke centered at its skeleton

3.3 Skeleton tangential slope evaluation using MIC

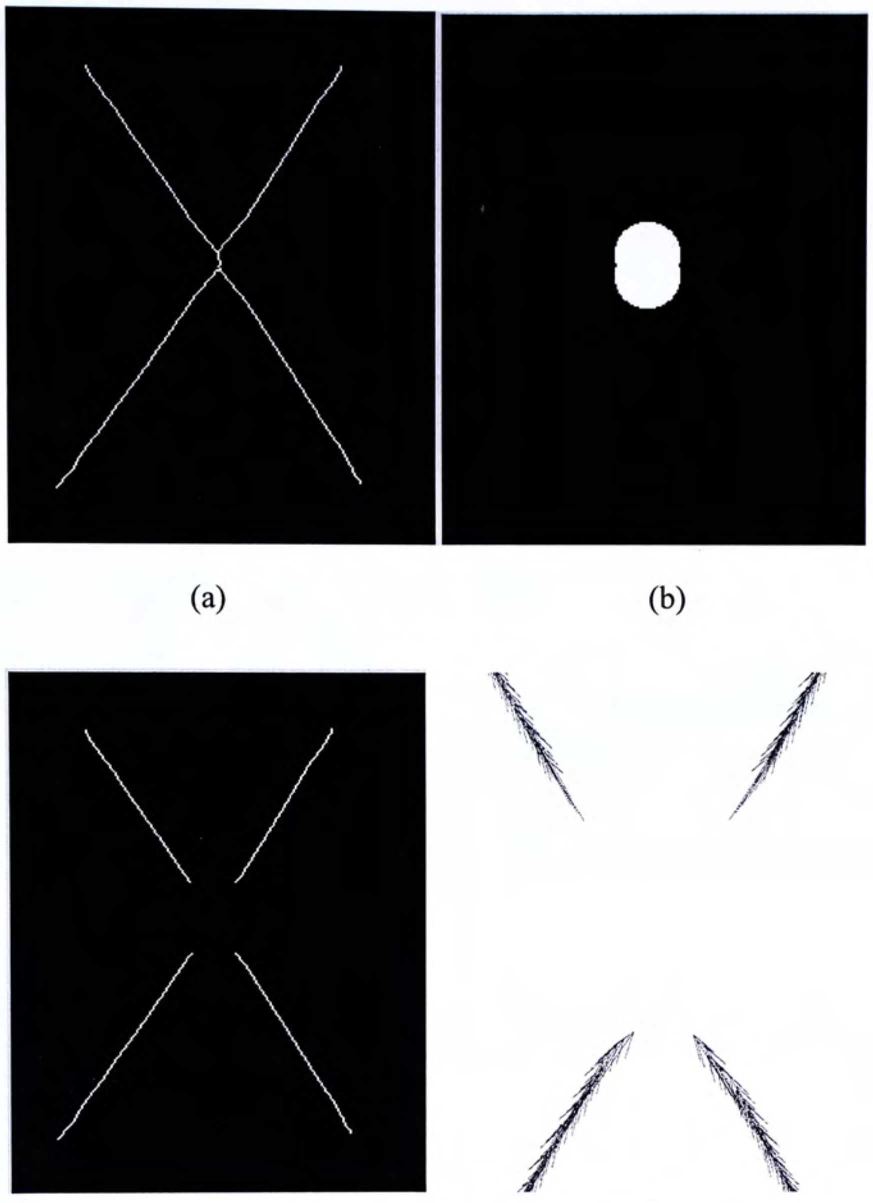
Figure 3.5 shows the resulting one-pixel-width skeleton inside two crossed strokes. By forming a circular disk centered at skeleton point, those disks are the MIC inside the strokes foreground as depicted in Figure 3.6, then the two tangent points which touch the boundary will give sufficient information in order to assign all the skeleton points a slope, as well as the vector shown in Figure 3.7(a) indicates the sense of direction while drawing the cross strokes.



(a) (b)

Figure 3.7. (a) The vector representation by MIC shows the directional drawing strokes, (b) The zoom in of (a) at junction

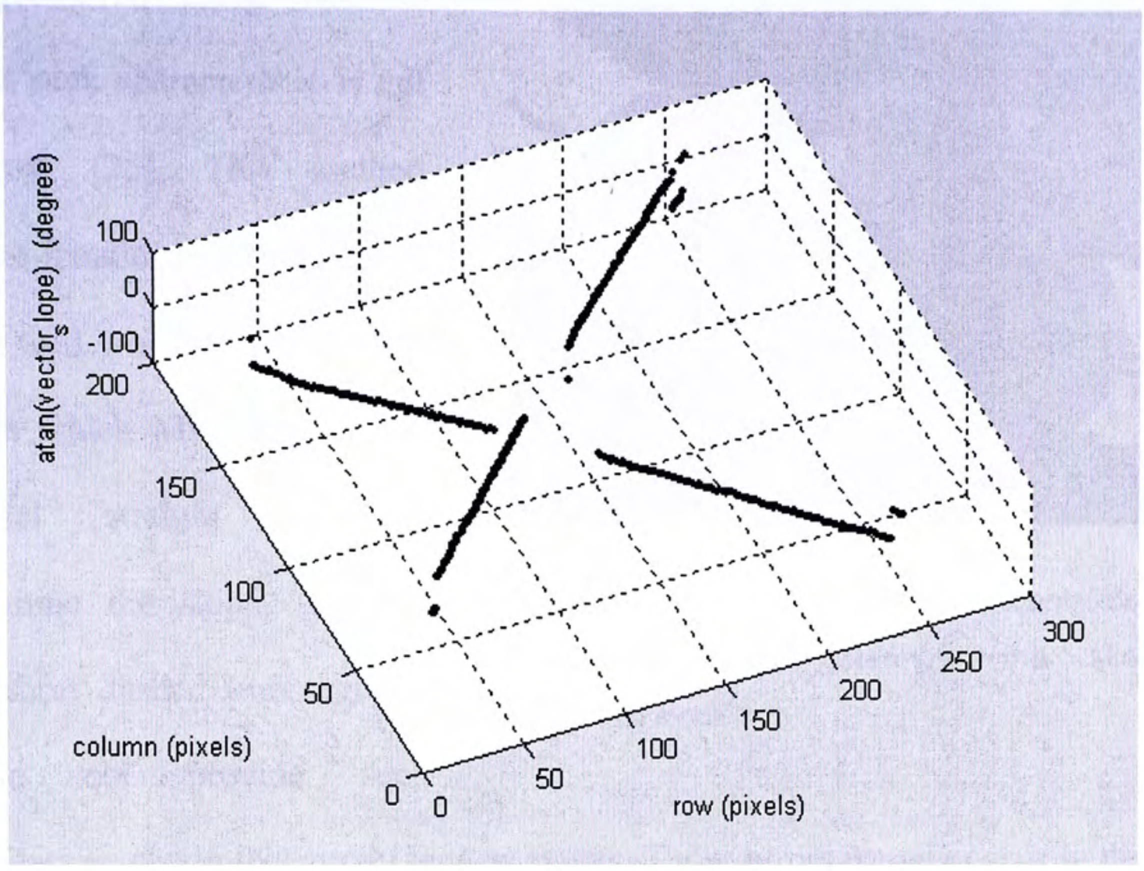
Figure 3.7(b) depicts the wrong vector assignment obviously at the junction region. Those vectors will mislead the unity of the brush stroke. Thus, the distortion elimination at junction is necessary for brush stroke extraction. Firstly, MICs in Figure 3.8(b) form inside the strokes centered at two distorted skeleton branch point pixels shown in Figure 3.8(a). Then, the skeleton pixels inside this MICs' union will be erased as Figure 3.8(c), and Figure 3.8(d) shows the remaining skeleton vectors which around the junction provide sufficient information to recover the directional strokes. For further demonstration, the direction of those vectors can be represented in a real value such as angle. The slope can be converted to angle under $\tan^{-1}$ function. Thus, using the obtained angle throughout the disconnected skeleton, similar angle value strokes would level at each other, see Figure 3.9.



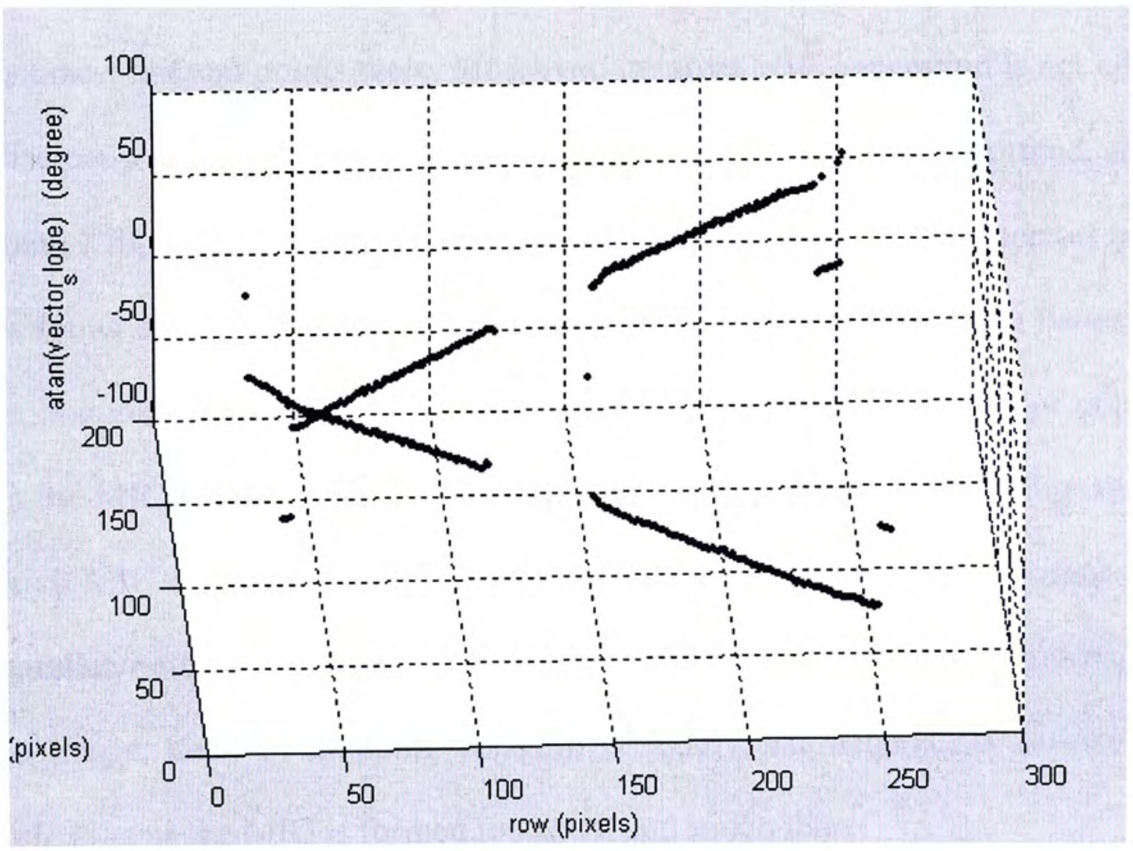
(a) (b)

(c) (d)

Figure 3.8. (a) The thinning skeleton distorts at junction, (b) Two MICs formed centered at the distorted skeleton branch pixel, (c) The distorted skeleton pixels are erased, (d) The zoom-in of vector representation at the junction region



(a)



(b)

Figure 3.9. Positive and negative slope angles which obtained by $\tan^{-1}$ function are separated into two major levels shown in two different isometric view graph (a) and (b)

However, using MIC to detect such characteristic is not accurate [26]. This method cannot handle arcs and dashed lines well. As the two tangent points which MIC touches the parallel straight boundary determine the slope, the arcs and short dashed line segment would not provide such boundary to obtain the correct tangent points. Those errors usually occur in the edge of the disconnected strokes as depicted in Figure 3.9 because the slopes found by the incorrect tangent points there. Moreover, iterative MIC generation is not efficient. For discrete pixel raster image, in case of thin stroke and small MIC formed, actually, the small MIC which is not a smooth circle is a 'gearwheel', and no correct tangent points found at all. Besides, it is difficult to locate the 'touching' point between the stroke boundary and MIC. We choose the area centroids as the tangent points, in which the MIC overlaps the two parallel boundaries, so you can see in Figure 3.10, in which MIC is the smallest circle formed such that the pixel overlapping occurs the parallel boundaries opposite to each other. Especially for low resolution line stroke image, such overlapping centroid is totally not helpful for precise point vectorization as the MIC is formed too small and rough there.

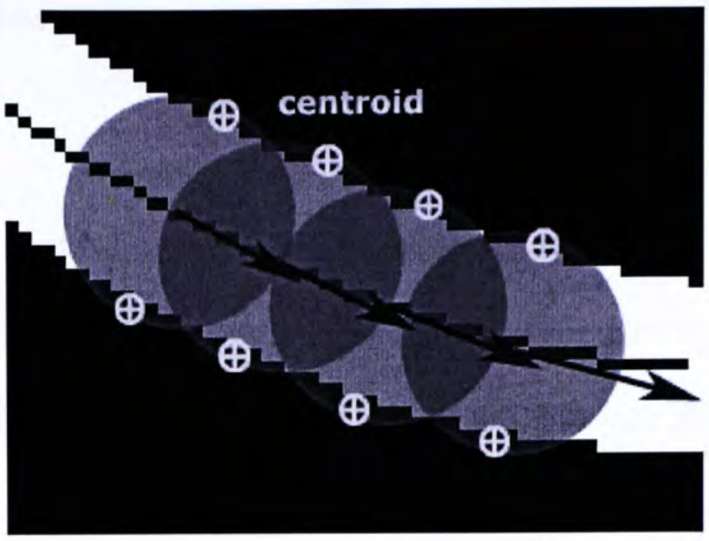


Figure 3.10. Two opposite centroids of overlapping area determine the skeleton tangential slope

3.4 Skeleton-based vectorization using Bezier curve interpolation

Upon the above limitation and disadvantage of vectorization using MIC, we would like to adopt a new method, Bezier curve interpolation, which estimates the tangential slope of the skeleton by smoothing the discrete pixel and converts the skeletons into vector representation as well under the directional stroke drawing. Given a stroke line, all its thinning skeleton one-width-pixels would be chosen as a sequence of control points which determine a Bezier spine curve depicted as Figure 3.11.

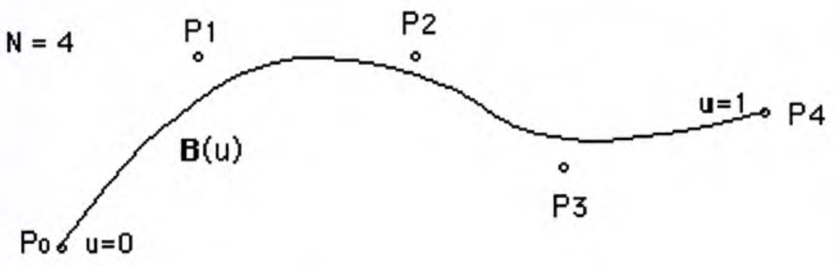


Figure 3.11. Five connected skeleton pixels determine a sequence of control points for interpolating a Bezier curve (source: [27])

The following describes the mathematics for the Bezier curve:

$$B(u) = \sum_{k=0}^{N} P_k \frac{N!}{k!(N-k)!} u^k (1-u)^{N-k} \text{, for } 0 \le u \le 1 \qquad \text{(Eqn 3.1)},$$

$$\text{Blending function: } F_B = \frac{N!}{k!(N-k)!} u^k (1-u)^{N-k} \qquad \text{(Eqn 3.2)}$$

We choose Bezier curve to represent the painting trajectory rather than B-spline or spline curve based on its original properties [27] as follow:

- The curve does not pass through any of the control points except the first and last as shown in Figure 3.11.
- The curve is always contained within the convex hull of the control points, it never oscillates wildly away from the control points, see Figure 3.12.

- The users need not to input any 'knots' as B-spline for determining the degree of curve polynomial. In Bezier curve, *degree*(F_B)< the number of control points. e.g. 3 control points results in a parabola, 4 control points a cubic curve. Thus, the degree order is sufficient to let the interpolation curve fit the control points closely.
- Adding multiple control points at a single position in space will add more weight to that point "pulling" the Bezier curve towards it.

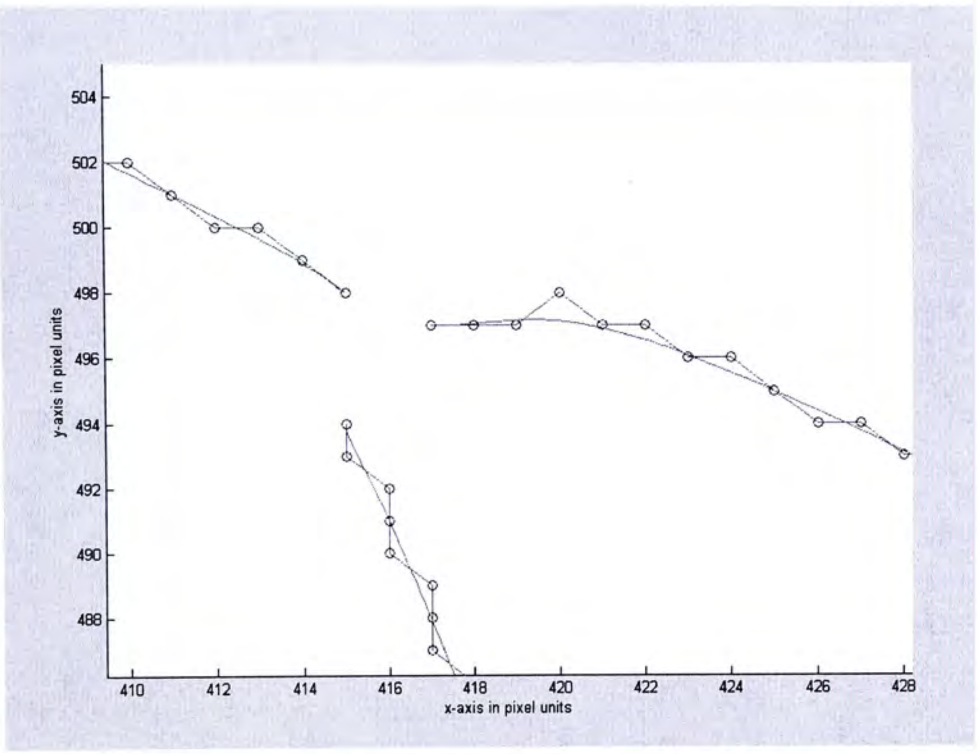
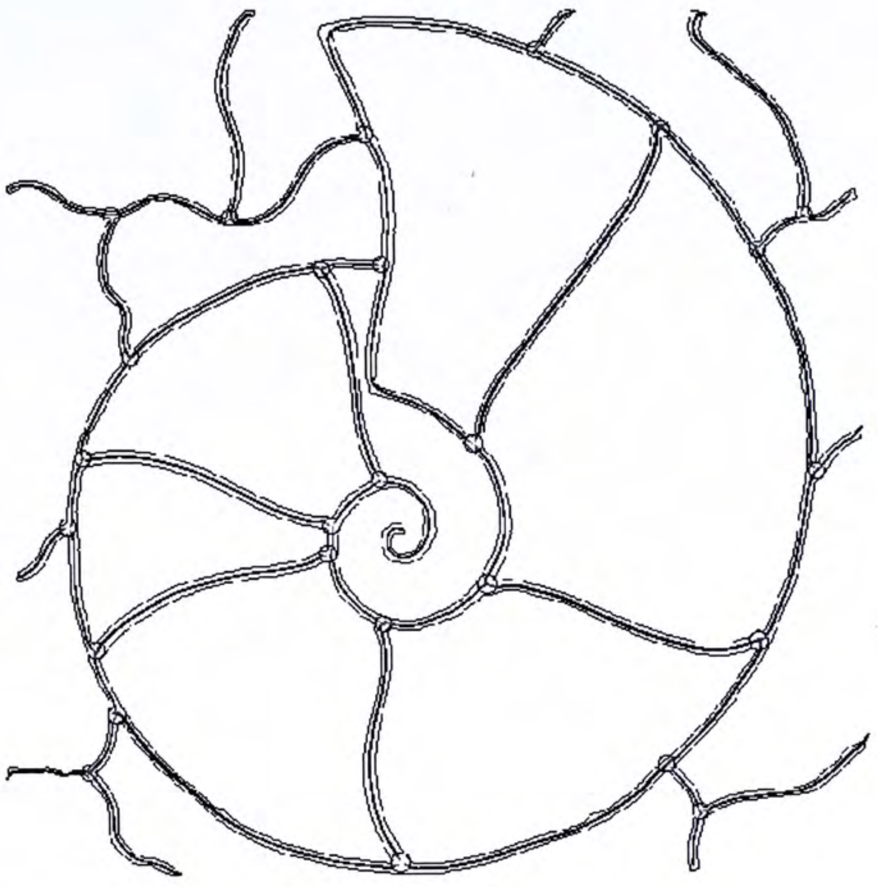
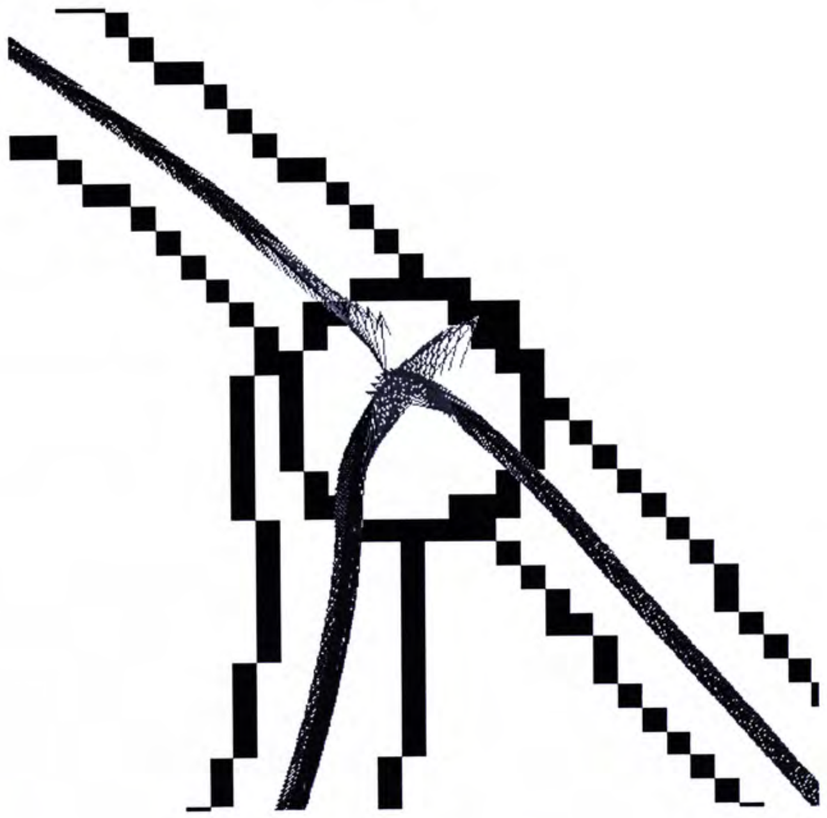


Figure 3.12. The circle bubbles show the location of the skeleton pixels and such Bezier curve does not pass through and oscillate away from them

Similar to MIC method, the vector representation which is achieved by Bezier interpolation can also recognize line stroke with junctions directly to eliminate the distortions at junctions. For line stroke extraction, all the skeleton branch pixels which connect to more than two pixels were erased first and then Bezier curve points would interpolate over the disconnected line strokes as Figure 3.12. Here, MIC is just used to form inside the strokes junction to erase the distorted skeleton pixels and its vectors as well. Figure 3.13 shows the vectors outside MICs at junctions giving sufficient information for extracting those directional strokes. As the vector is formed under the Bezier interpolation based on the stroke skeleton, the appropriate slopes are found independent of the stroke boundary even in arcs and dashed line segment. Figure 3.14 shows the correct and smooth slope angle values computed under Bezier interpolation.



(a)



(b)

Figure 3.13. (a) MIC and the directional stroke vector are formed inside the line stroke. (b). The zoom-in on one of the "T" junction, the red vectors which starting points inside MIC would be erased

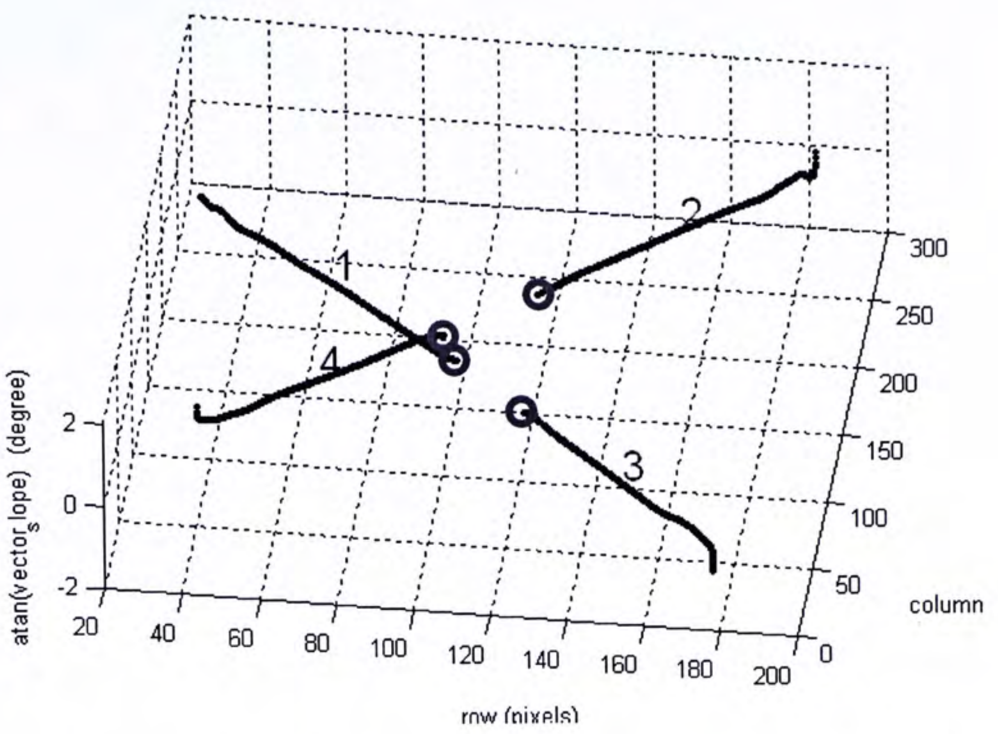


Figure 3.14. Four bubbles indicate the angles value in which the points are attached on the MIC boundary

3.5 Line stroke extraction

Line stroke reconnection can be done by using the slope values on the MIC boundary depicted as Figure 3.14. By comparing with Figure 3.9, the smoother slope angles are achieved under Bezier interpolation. Using (Eqn 3.3), non-negative cross slope angle θ_{ij} can be computed from two slopes m_i and m_j, where i and j are the index label of disconnected line stroke.

$$\theta_{ij} = \tan^{-1}\left(\frac{|m_i - m_j|}{1 + m_i m_j}\right) \qquad \text{(Eqn 3.3)}$$

The symmetric matrix ϑ^k is as follows shows all the cross angles among n slopes at the junction k:

$$\vartheta^k = \begin{bmatrix} 0 & & & & \\ \theta_{21} & 0 & & \bullet & \\ \theta_{31} & \theta_{32} & . & & \\ . & . & . & . & \\ \theta_{n1} & \theta_{n2} & . & \theta_{n(n-1)} & 0 \end{bmatrix}$$ (Eqn 3.4)

There are $\frac{(n-1)^2}{2}$ cross angle θ_{ij} would computed, where $\theta_{ij} = \theta_{ji}$.

Under the condition: $\begin{cases} \theta_{kz} = \min(\theta_{(z+1)z}, \theta_{(z+2)z,...,}\theta_{nz}) < \theta_{thre} \\ \theta_{kz} \le \min(\theta_{(k+1)k,}\theta_{(k+2)k,...,}\theta_{nk}) \quad if \quad z \ne n \end{cases}$ (Eqn 3.5),

where $k = 1,...,n-1$, $k < z \le n$, stroke k and z should be originally connected and θ_{thre} acts as the threshold angle value to allow this connection. Thus, the unity of line stroke will be recovered with this connection. Like example in Figure 3.14, the only cross angle matrix in degree unit is as follow:

$$\vartheta^1 = \begin{bmatrix} 0 & & & \\ 78.0985 & 0 & & \\ 0.4681 & 78.5672 & 0 & \\ 77.6487 & 0.4504 & 78.1167 & 0 \end{bmatrix}$$, and we usually set $\theta_{thre} = 10°$

, i.e., $\begin{cases} \theta_{31} = \min(\theta_{21}, \theta_{31}, \theta_{41}) < \theta_{thre} \\ \theta_{31} < \theta_{43} \end{cases}$

and, $\theta_{42} = \min(\theta_{32}, \theta_{42}) < \theta_{thre}$, stroke 1 would be connected to stroke 3 under the condition suggested in (Eqn 3.5), similar for stroke 2 to stroke 4.

Somestimes the performance of the line extraction is much dependent on the edge slope angles of the disconnected strokes, and is sensitive to the threshold angle θ_{thre} that allows those connections. In what follows, the fuzzy set concept provides us with an intuitive method of representing on form of uncertainty, vagueness. This is useful in our decision-making situation where it is not possible to draw crisp boundaries in deciding if the change of vectors direction is rapid at the junctions. It is envisioned that fuzzy identification may be available to recognize directional stroke and avoid the wrong stroke connection.

In our case, such junction region is always small and short straight line segment is enough to fill the gap between two connected strokes at the junction without any distortion as Figure 3.15. Another two line stroke extraction results are shown in Figure 3.16(b) and Figure 3.17(b) accompanied their distorted junction skeleton image in Figure 3.16(a) and Figure 3.17(a) respectively.

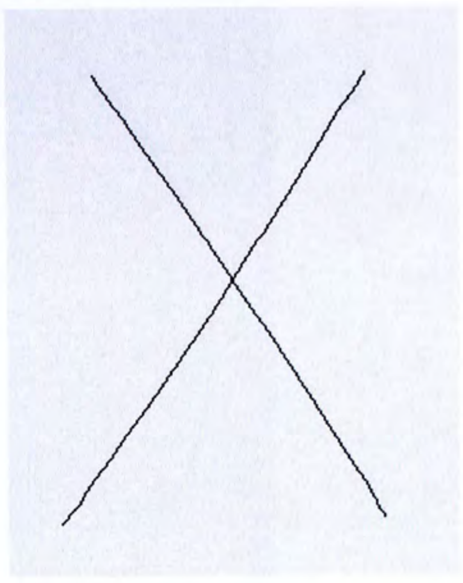


Figure 3.15. The cross strokes are recovered without any distortion

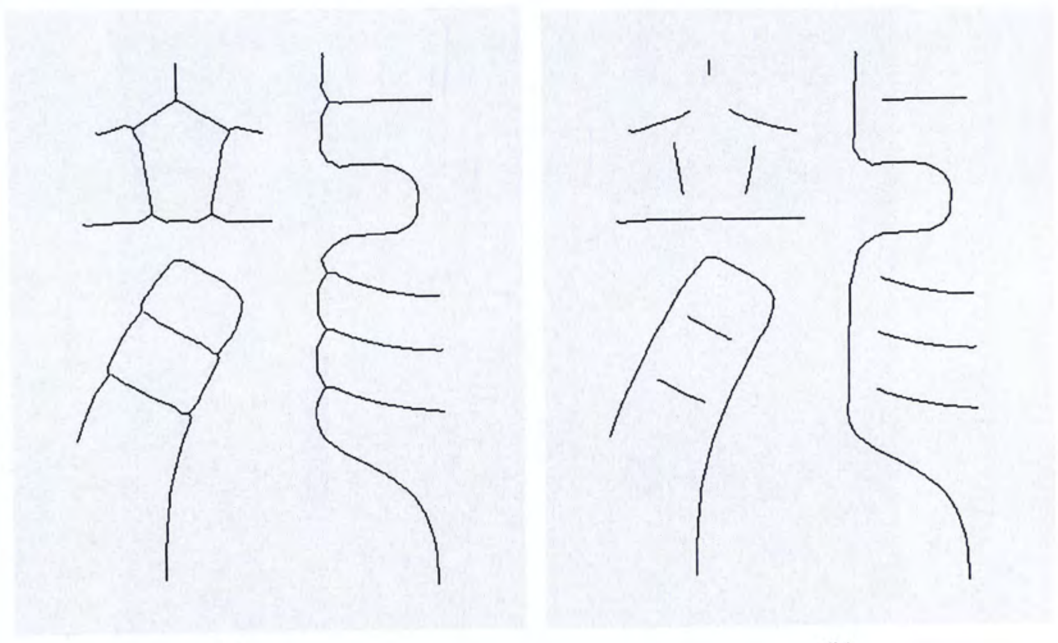
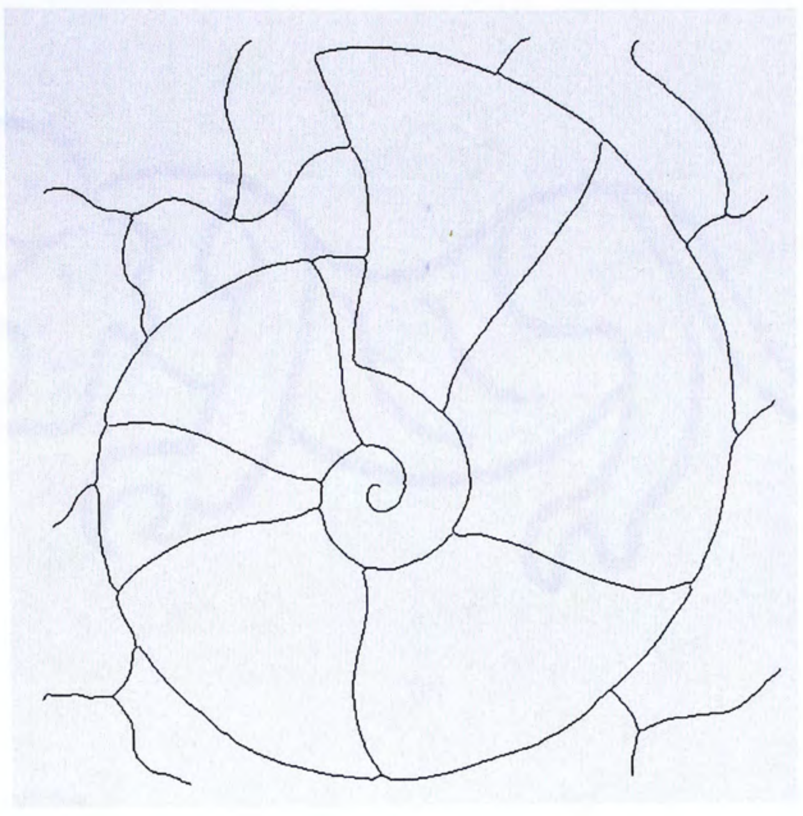
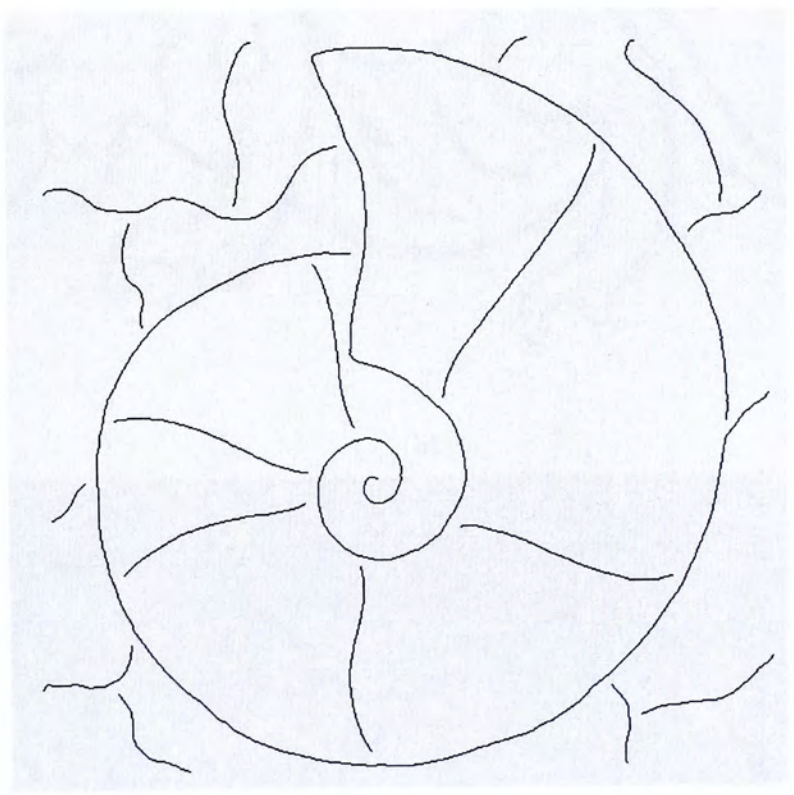

(a) (b)

Figure 3.16. (a) A Chinese character "龍" in ancient Seal style, (b) processed by line extraction

In detail, we demonstrate the Chinese line drawing and calligraphy respectively with higher artistic value shown in Figure 3.18(a) and Figure 3.19(a), those strokes are vectorized depicted in Figure 3.18(b) and Figure 3.19(b), and the final line extraction result overlapped with the original and junction regions is shown in Figure 3.18(c) and Figure 3.19(c).



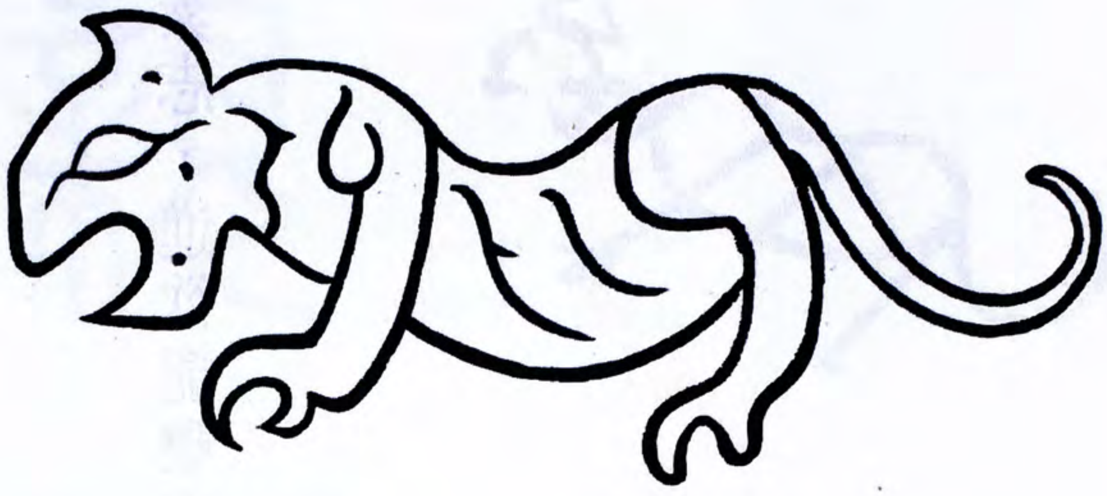
(a)



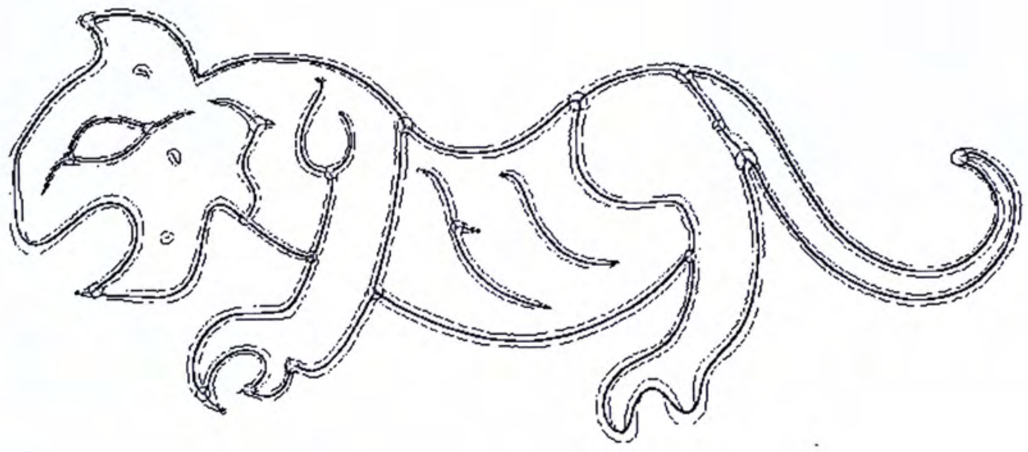
(b)

Figure 3.17. (a) The stroke skeletons are distorted at the junctions, (b) Those distortions are reduced by line stroke extraction

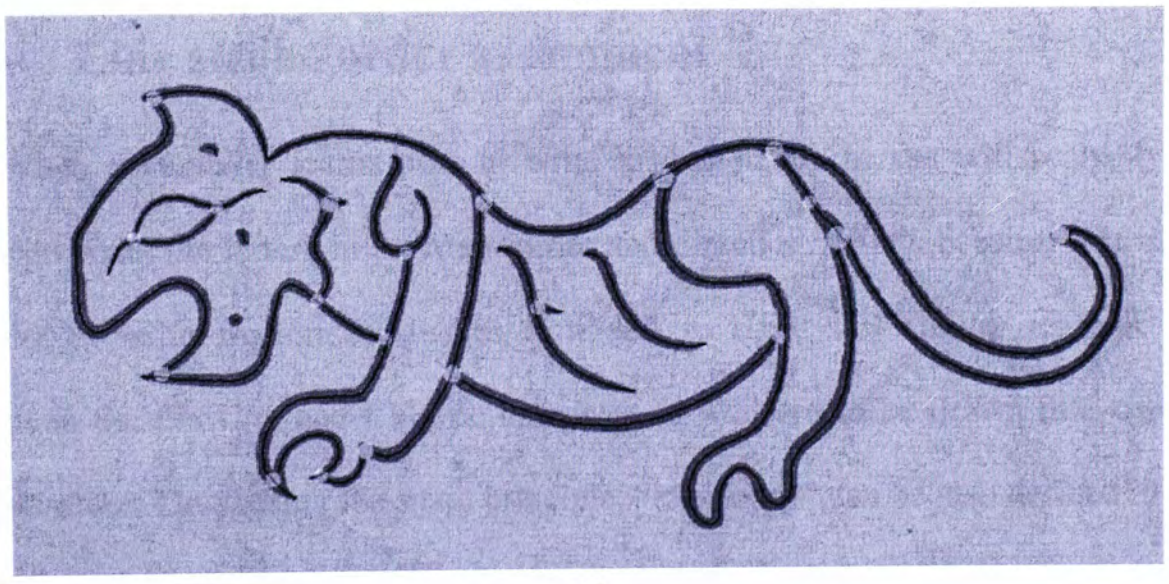
These results will act as the preliminary drawing schemes discussed in chapter 5. Obviously, the stroke connectivity at some junction especially "T" and "Y" junction is not completed yet. This kind of disconnection will be modified by the visual-based iterative drawing process discussed latter in Chapter 4 and 5.



(a)



(b)



(c)

Figure 3.18 (a) The original art piece –Tiger "虎", is a ink drawing of Chinese Character painting "文字畫" (source [28]), (b) The vector representation for stroke extraction, (c) The recovered stroke skeletons after stroke extraction process are shown with MIC junctions

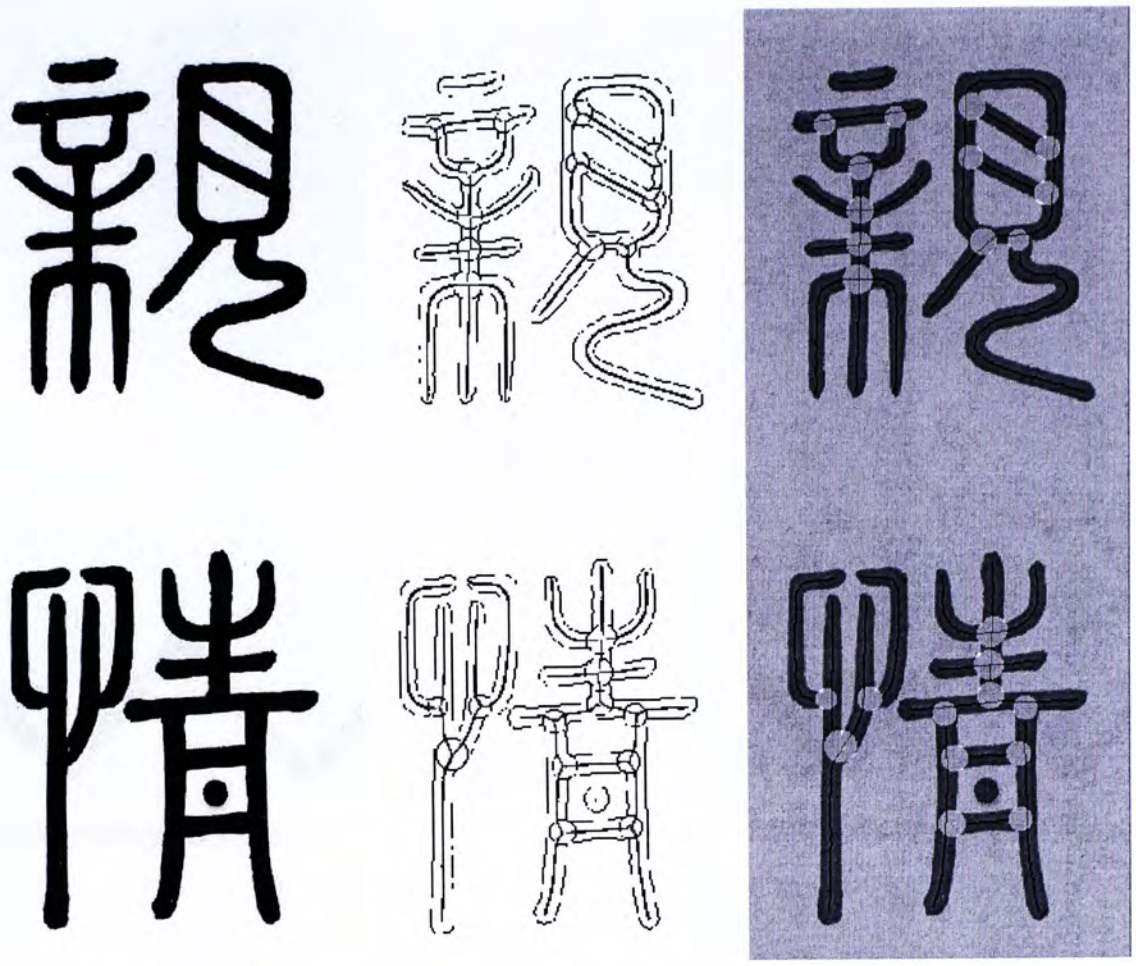
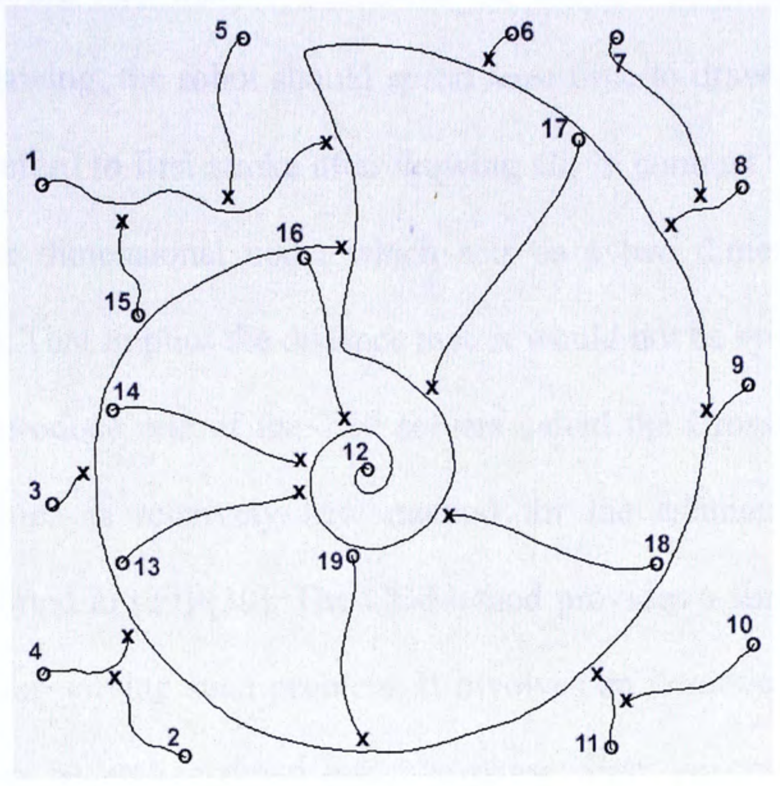

(a) (b) (c)

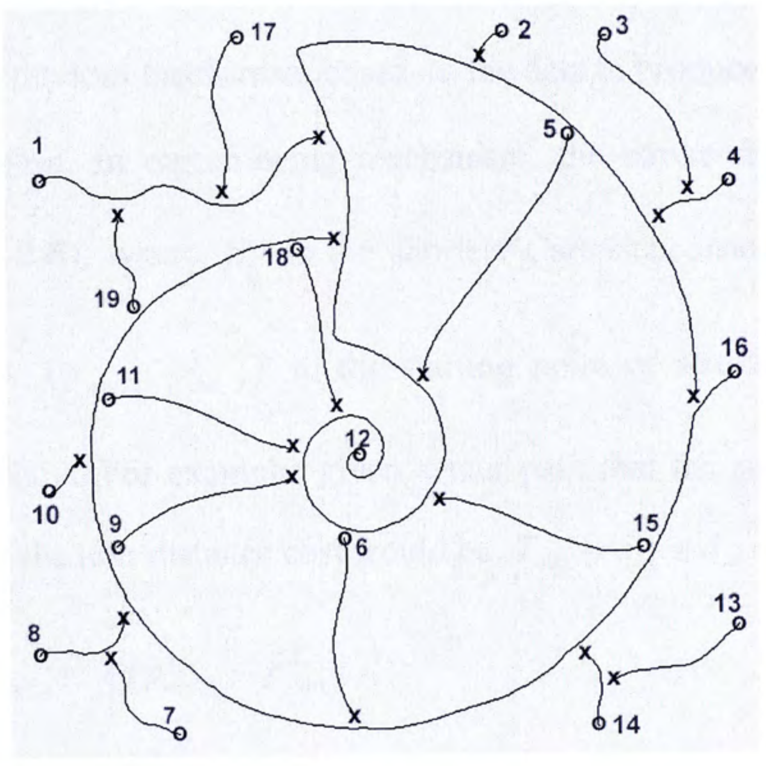
Figure 3.19 (a) The original art piece – "親情" in Seal Character "篆書" (source [28]), (b) The vector representation for stroke extraction, (c) The recovered stroke skeletons after stroke extraction process are shown with MIC junction

3.6 Line stroke order assignment

When a beginner learns how to write his/her name, he/she will probably be taught to write the letters in a certain order and direction. That's because it is more *efficient* to write western languages from left to right. For similar reasons, the strokes in the drawing and Chinese character as well are to be drawn in a certain defined order. Thus, using the same principle, "efficiency" can be also defined in the aspect of robot drawing such that the robot should save most time to draw all the strokes. This efficiency reminded us of thinking about the well known problem called TSP (Traveling Salesman Problem).



(a)



(b)

Figure 3.20. (a) A picture drawn in an arbitrary stroke order, (b) Proper drawing order given by CE-Method

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 that, in what order he should visit them to minimize the distance traveled. Similar to our robot efficient drawing, the robot should spend least time to draw n strokes, once for each except return to first stroke after drawing all. In contrast to TSP, the stroke drawn is not one dimensional node, which acts as a two dimensional city with entrance and exit. That implies the distance matrix would not be symmetric at all.

Here, we introduce one of the TSP solvers called the Cross-Entropy Method (CE-Method) which is relatively new method for the estimation of rare-event probabilities reported in [29]-[30]. The CE-Method provides a simple, efficient and general method for solving such problem. It involves an iterative procedure where each iteration can be broken down into two phase: first, generate a random data sample such as TSP tour according to a specific mechanism. Second, update the parameters of the random mechanism based on the data to produce a "better" sample in the next iteration. In our drawing mechanism, the stroke distance should be defined in (Eqn 3.6), where d_{ij} is the shortest Cartesian distance from the end point of stroke i $(p^{i}_{end,x}, p^{i}_{end,y})$ to the starting point of stroke j $(p^{j}_{end,x}, p^{j}_{end,y})$ on the drawing plane. For example, given a tour path that the stroke sequence are $5 \rightarrow 4 \rightarrow 6 \rightarrow 7$, the tour distance cost would be $T_{dist} = d_{54} + d_{46} + d_{67}$.

$$d_{ij} = (p^{i}_{end,x} - p^{j}_{start,x})^2 + (p^{i}_{end,y} - p^{j}_{start,y})^2 \qquad \text{(Eqn 3.6)}$$

$$D = \begin{bmatrix} 0 & d_{12} & d_{13} & . & d_{1n} \\ d_{21} & 0 & d_{23} & . & . \\ d_{31} & d_{32} & . & . & . \\ . & . & . & . & d_{(n-1)n} \\ d_{n1} & d_{n2} & . & d_{n(n-1)} & 0 \end{bmatrix} \qquad \text{(Eqn 3.7)},$$

which is non-symmetric distance cost matrix such that $d_{ij} \neq d_{ji}$.

Now, we choose Figure 3.17(b) for the demonstration example. For

CE-Method, we set the number of sample to generate each round *N*=1000, the fraction of best samples to take *rho*=0.05, the smoothing parameter *alpha*=0.8 and node placements setting is adopted. In Figure 3.20(a), the arbitrary stroke drawing order is assigned in which "o" and "x" indicate the start and the end point respectively belongs to the line stroke and the drawing order number placed near the bubbles "o". The tour distance cost is computed such that T_{dist}=1,211,533$(pixel)^2$.After using CE-Method, the proper stroke drawing order depicted as Figure 3.20(b) is assigned such that the tour cost is reduced about 75% to T_{dist}=297,242$(pixel)^2$.

3.7 Chapter summary

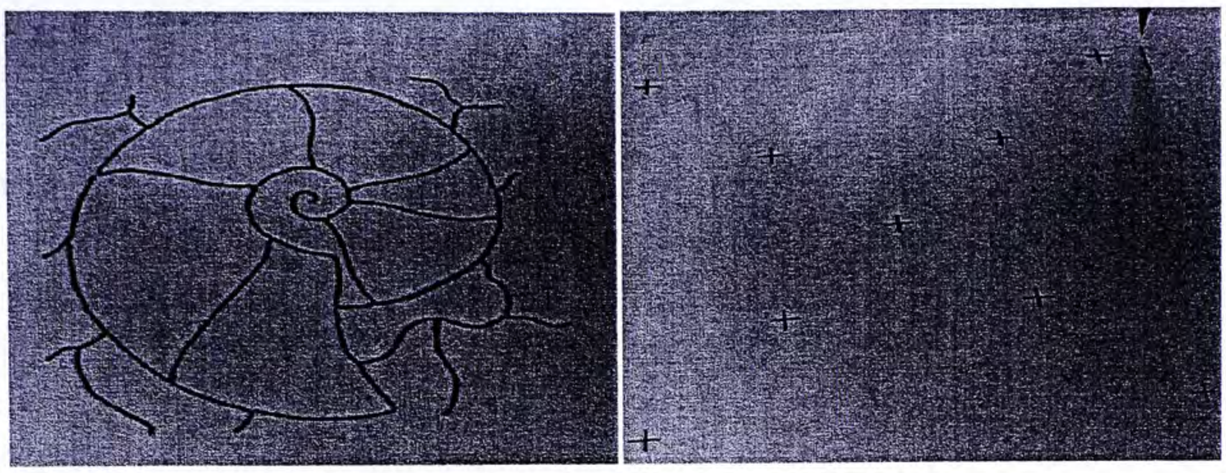
Line drawing technique seems to be simple and natural that a child can more or less handle. The process, however, actually requires pattern recognition, machine intelligence, learning and knowledge acquisition even if an accurate drawing mechanism is available. In this chapter, line stroke extraction is describes. It is shown that skeleton-based vectorization with Bezier curve interpolation is accurate enough to recover the directional stroke in the case that the junction skeleton is distorted by the preceding thinning algorithm. For low resolution raster input, smooth and correct vector slopes computed by Bezier interpolation rather than MIC method is highly preferred in eliminating skeleton distortions at line junctions. After a proper extraction of the strokes, the CE-Method is adjusted to yield the appropriate stroke drawing ordering in the time saving sense. Specifically, it is efficient to replicate the drawing/writing containing large number of strokes. A valuable masterpiece such as Chinese calligraphy epigraph "碑文" is an example.

4. PROJECTIVE RECTIFICATION AND VISION-BASED CORRECTION

This chapter describes the addition of vision-based capabilities in the platform. This includes projective rectification [31] of the executed work by an overlooking camera, and generation of corrective drawing actions for iterative improved drawing. As a first version [32], projective rectification is conducted using homography matrix computed upon 9 selected correspondence points on the drawing plane. The rectified version of the executed drawing is then readily compared to the original image with the intention to generate corrective actions for enhancement in the next iterative execution. To further enhance performance, a high accuracy automatic rectification scheme via GA-based homography transformation is also implemented. A demonstration of how the vision information can be used to evaluate and improve drawing is then given.

4.1 Projective rectification

As mention before, an image of the executed work captured by the camera and the executed image first undergoes a projective rectification. The aim of the rectification is to remove the projective distortion in the perspective image of the drawing plane, to the extent that similarity properties such as angles or relative ratios of lengths can be visualized at the original plane. In theory, projective distortion can be completely removed by selecting at least four reference points (8 degrees of freedom) on the drawing plane to compute the homography transformation. Figure 4.1(a) shows the angled image captured in a case study, and Figure 4.1(b) shows the 9 (over minimum 4) crosses drawn by the robot are selected for homography transformation.



(a) (b)

Figure 4.1. (a) Captured image of an executed drawing, (b) 9 points corresponding for generating homography matrix

4.2 Homography transformation by selected correspondences

In other to compare the executed drawing with the original image, the captured image needs to be transformed into a full plane view as if observed from atop. The well known Direct Linear Transformation (DLT) [33] algorithm is adopted to this effect. The algorithm determines a 3x3 homography matrix upon given at least four 2D to 2D point correspondences, $X_i \leftrightarrow X_i'$. The equation can be expressed as $X_i' \times HX_i = 0$. In the present case, the equation involves non-homogeneous vectors as all correspondences are in the image coordinates. Hence, the 3-vectors X_i' and HX_i are equal. Specifically, given n correspondence pairs, $X_i = (x_i, y_i, 1)^T$ and $X_i' = (x_i', y_i', 1)^T$ for i= 1 to n, the cross product equation is:

$$\mathrm{X}_i'\times H\mathrm{X}_i = \begin{pmatrix} y_i' h^{3T}\mathrm{X}_i - h^{2T}\mathrm{X}_i \\ h^{1T}\mathrm{X}_i - x_i' h^{3T}\mathrm{X}_i \\ x_i' h^{2T}\mathrm{X}_i - y_i' h^{1T}\mathrm{X}_i \end{pmatrix} = 0, \qquad \text{(Eqn 4.1)},$$

where $H = \begin{pmatrix} h^{1T} \\ h^{2T} \\ h^{3T} \end{pmatrix}$.

With $h^{jT}x_i = x_i^T h^j$, (Eqn 4.1) can be written as:

$$\begin{bmatrix} 0^T & -\mathrm{X}_i^T & y_i'\mathrm{X}_i^T \\ \mathrm{X}_i^T & 0^T & -x_i'\mathrm{X}_i^T \\ -y_i'\mathrm{X}_i^T & x_i'\mathrm{X}_i^T & 0^T \end{bmatrix} \begin{pmatrix} h^1 \\ h^2 \\ h^3 \end{pmatrix} = 0, \qquad \text{(Eqn 4.2)}.$$

The third equation in (Eqn 4.2) can be omitted as it is linearly dependent. Each correspondence hence yields two linearly independent equations:

$$\begin{bmatrix} 0^T & -\mathrm{X}_i^T & y_i'\mathrm{X}_i^T \\ \mathrm{X}_i^T & 0^T & -x_i'\mathrm{X}_i^T \end{bmatrix} \begin{pmatrix} h^1 \\ h^2 \\ h^3 \end{pmatrix} = 0 \text{ , or } A_i h = 0, \qquad \text{(Eqn 4.3)}$$

where A_i is the 2×9 matrix and h is 9×1 vector. Then, putting the n 2×9 matrices A_i into a single $2n\times 9$ matrix A and generating the singular value decomposition (SVD) of A, the unit singular vector corresponding to the smallest singular value then yields h, and hence H. Figure 4.1(b) shows the n =9 correspondences picked for generating the rectifying matrix.

The homography matrix 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 (see Figure 4.2) before the two can be compared. Using homography transformation also increases the flexibility that the installed camera needs not be always looking down at the drawing plane. Figure 4.3 shows the homography transformation result of the image in Figure 4.1(a) based on the selection of the 9 correspondences of Figure 4.1(b). For explicit demonstration, Figure 4.4 shows the rectified image of the executed drawing which is overlapped the original image in the same pixel resolution image.

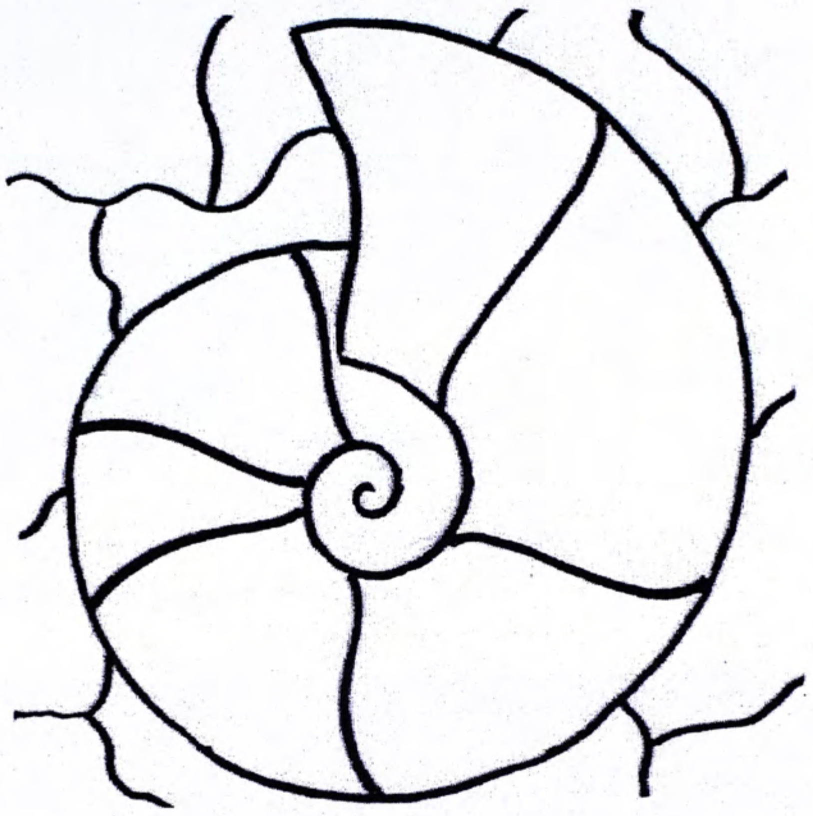


Figure 4.2. The original input image in pixel size 580×580

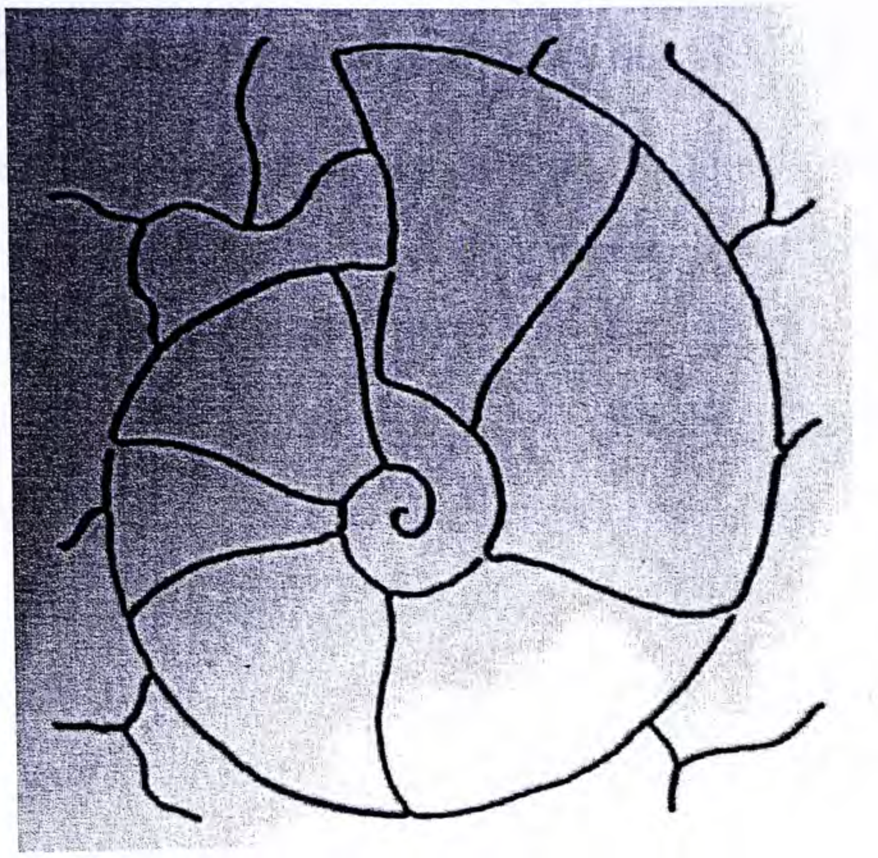


Figure 4.3. The rectified robot drawing upon careful selection of 9 correspondences

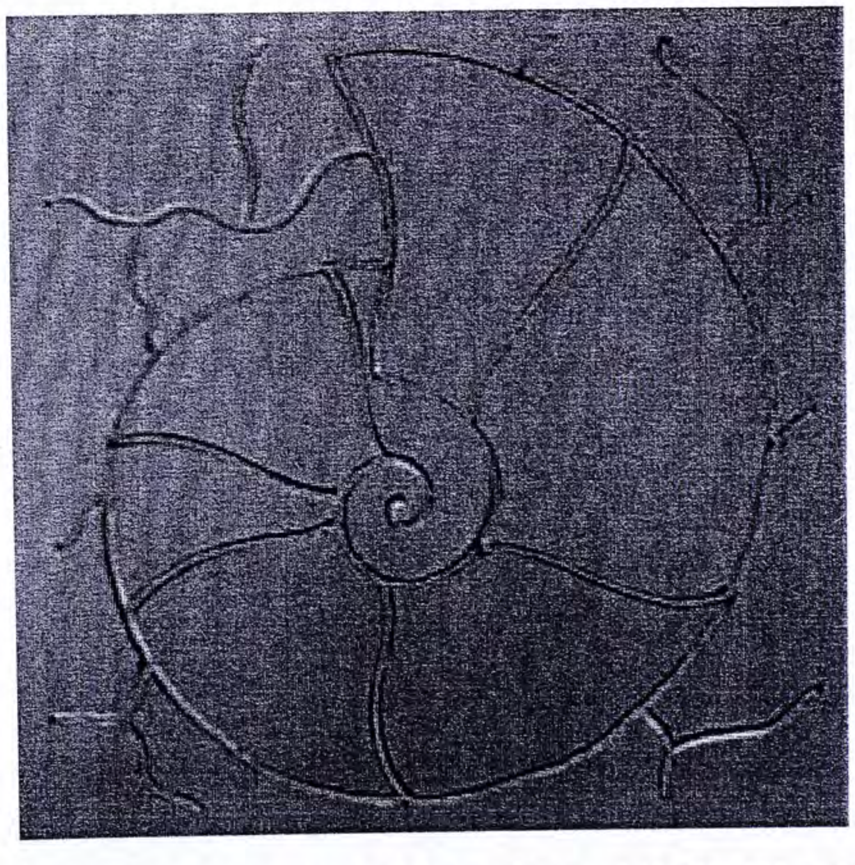


Figure 4.4. Overlapping of the rectified robot drawing and original image

4.3 Homography transformation using GA

The above homography transformation relies on manual picked of the correspondence points. Presumably, one can pick more correspondence points in a quest to reduce the overlapping errors by, e.g., SVD technique [33]. However, it is quite exhausting to precisely pick too many points in the image. Figure 4.5 shows the overlapping between the rectified image of the executed drawing in Figure 4.1(a) and the original image via a homography transformation computed using inaccurately selected points. As can be observed, the overlapping result is far from desirable. This problem is due to the fact that the homography transformation is highly sensitive to the exact correspondence of the selected points in Figure 4.1(b). As such, only manual selection is extremely difficult to produce overlapping performance to guarantee useful corrective drawing actions in our application.

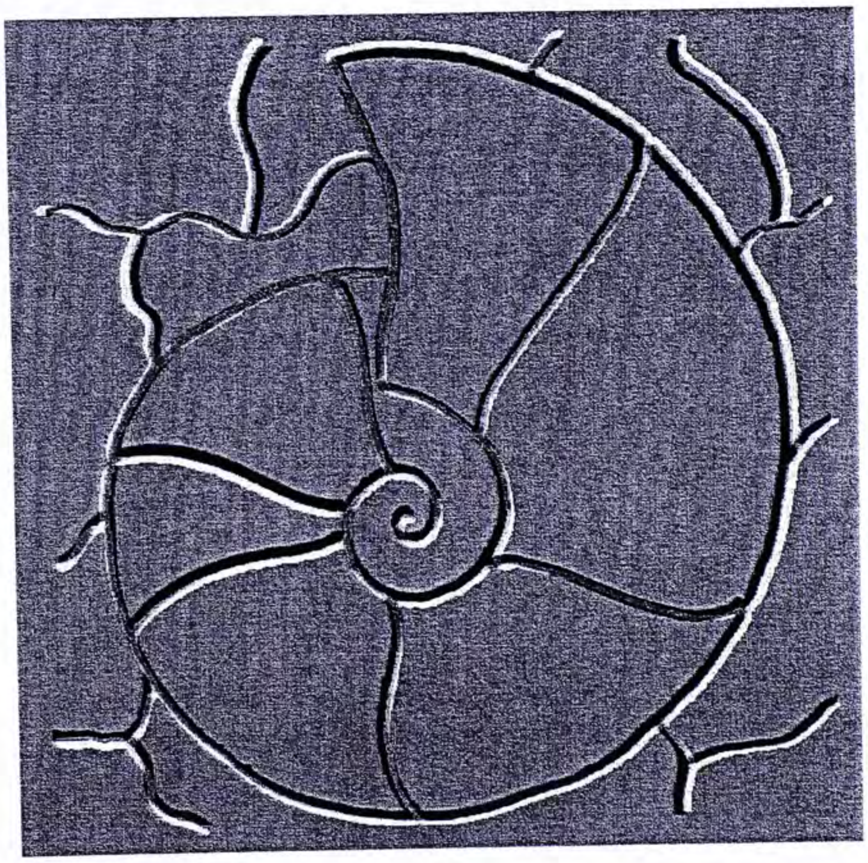


Figure 4.5. Poor overlapping qualities of the rectified executed image in white and the original image in black

In what follows, Genetic Algorithm (GA) is introduced to arrive at high performance homography transformation upon the initialization of only 4 roughly selected correspondences in the image [34]. The various elements of GA are first described.

(i) GA Representation

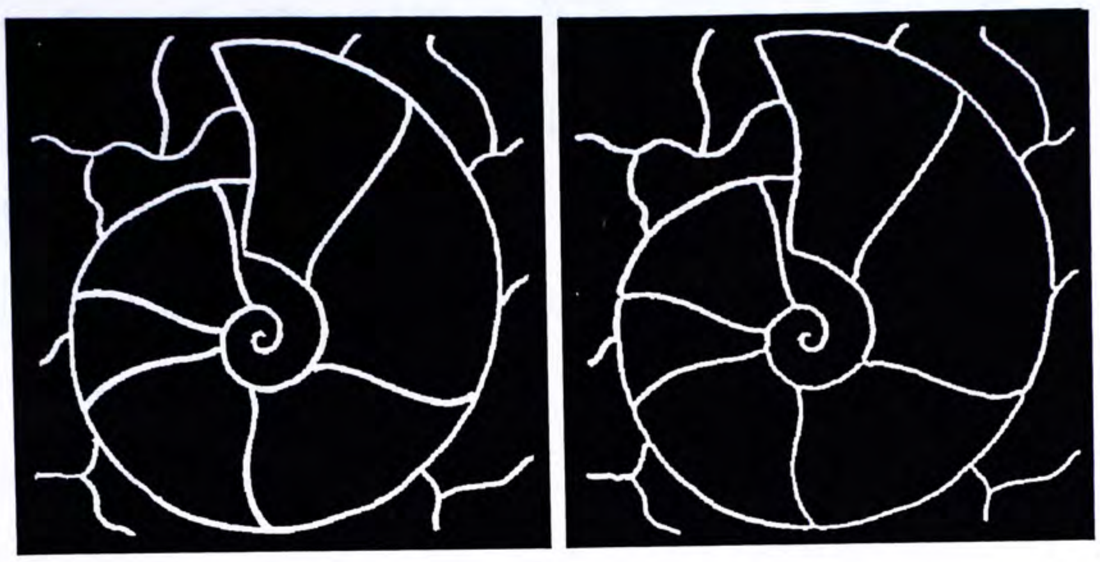
We pick parametric genes ($\Delta P_{i=1,\ldots,4}$) as fine tunings in the displacements of the manually selected correspondence points. The chromosome comprising of the parametric genes is as shown in Figure 4.6.

ΔP_{1x}	ΔP_{1y}	ΔP_{2x}	ΔP_{2y}	ΔP_{3x}	ΔP_{3y}	ΔP_{4x}	ΔP_{4y}

Figure 4.6. Chromosome structure of the correspondence displacements

(ii) Objective and Fitness Evaluation

The objective function serves to provide a measure of how individuals have performed in the problem domain [35]. In the case of a minimization problem, the most fitted individual will have the lowest numerical value of the associated objective function. This raw measure of fitness is usually used as an intermediate stage in determining the relative performance of the individuals in GA [36]. Figure 4.7 shows the original image and rectified image converted to black and white, where the foreground pixels are taken as 1 s and the background pixels as 0 s. The objective cost value can then be defined as:



(a) (b)

Figure 4.7. Image in black and white (a) The original, (b) The rectified executed image

Objective: minimize $o(H)=\sum_{j}^{m}\sum_{i}^{n} p_{ij}^{r}\cdot\overline{p_{ij}^{o}}$ (Eqn 4.4),

where i is the row number, j is the column number, p_{ij}^{o} is the pixel value of $n\times m$ original image, p_{ij}^{r} is the pixel value in $n\times m$ rectified executed image, and $\overline{p_{ij}}$ the conjugate pixel value, ($0\rightarrow 1$ and $1\rightarrow 0$). Upon a 3×3 matrix H calculated based on displaced correspondences induced by the given chromosome, an inverse value of the objective cost function $o(H)^{-1}$ which represents the overlapping quality of two images can be computed.

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 lowered in the quest of a "fitter" individual in the next generation. In this paper, a non-linear fitness assignment is adopted to prevent premature convergence. Individuals are assigned fitness values according to their rankings in the population rather than their raw performance. The fitness of a individual in the population is calculated as,

Fitness: $F(x_i) = 2 - PRS + 2(PRS - 1)\dfrac{x_i - 1}{N_{ind} - 1}$ (Eqn 4.5),

where $x_i = ranking(o(H_i))$ is the position in the ordered population of the i^{th} individual, $i = 1, \quad , N_{ind}$, and N_{ind} is a population size in each generation. As suggested in [37], the variable 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. Note that the fitness assignment is always positive.

(iii) Evolutionary Computing

The following genetic parameters and operations are adopted for the projective rectification problem at hand.

1) Displacement resolution: d_r pixel unit.

2) Displacement range: $[d_{\max}, d_{\min}]$ in pixel unit.

3) Chromosome length: 4 correspondences points; 8 degree of freedoms.

   Chromosome length in binary coded, $l_{bit} = 8\left\lceil \log_2\left(\dfrac{d_{\max} - d_{\min}}{d_r} + 1\right) \right\rceil$

4) Crossover method:

   One-point crossover with probability of crossover p_c

5) Mutation method:

   New individuals take the current population and mutate each element with probability of mutation $p_m = 0.7 / l_{bit}$.

6) Selection method: Stochastic Universal Sampling with population selection $p_s = 90\%$.

7) Replacement: Fitness-base reinsertion to current population.

8) Population size N_{ind}: between 40-100, based on the length of the chromosome.

9) Number of generation past: N_{gen}

10) Initial population: randomly generated within the displacement range according to displacement resolution.

The GA homography estimation is performed on a PC with Pentium 4 (3.06GHz) CPU, 1GB RAM, and using Windows XP operating system. Computation time for finding the optimal homography transformation is roughly 45-60mins. This computation time (averaged over multiple runs) is for reference only since the program is running in the debug mode of MatLab. The actual speed should be much faster. As corrective actions are to be generated off-line, the computational cost here is actually not too important. Moreover, computation is needed to be performed only once after each adjustment to camera orientation.

Result 1

The overlapping result as produced by the evolutionary computing is shown in Figure 4.8. Marked improvement over that of Figure 4.5 is observed. Here, the pixel size is 580×580 image. The parameters are: d_r=0.5 pixel units, $d_{\max}$=20 pixel units, $d_{\min}$=-20 pixel units, l_{bit}=56, p_c=0.7, p_m=0.0125, N_{ind}=60 and N_{gen}=80. The objective cost $o(H)$ of the executed image upon rectification by the manually selected correspondence points, depicted in Figure 4.5, is =12091. After GA homography estimation, the objective cost decreases to $o(\widetilde{H})$=854 in Figure 4.8. The corresponding offset displacement and overlapping quality are ΔP=[19, 4.5, -13, 9.5, -4, 0, -3, -5.5] pixel units.

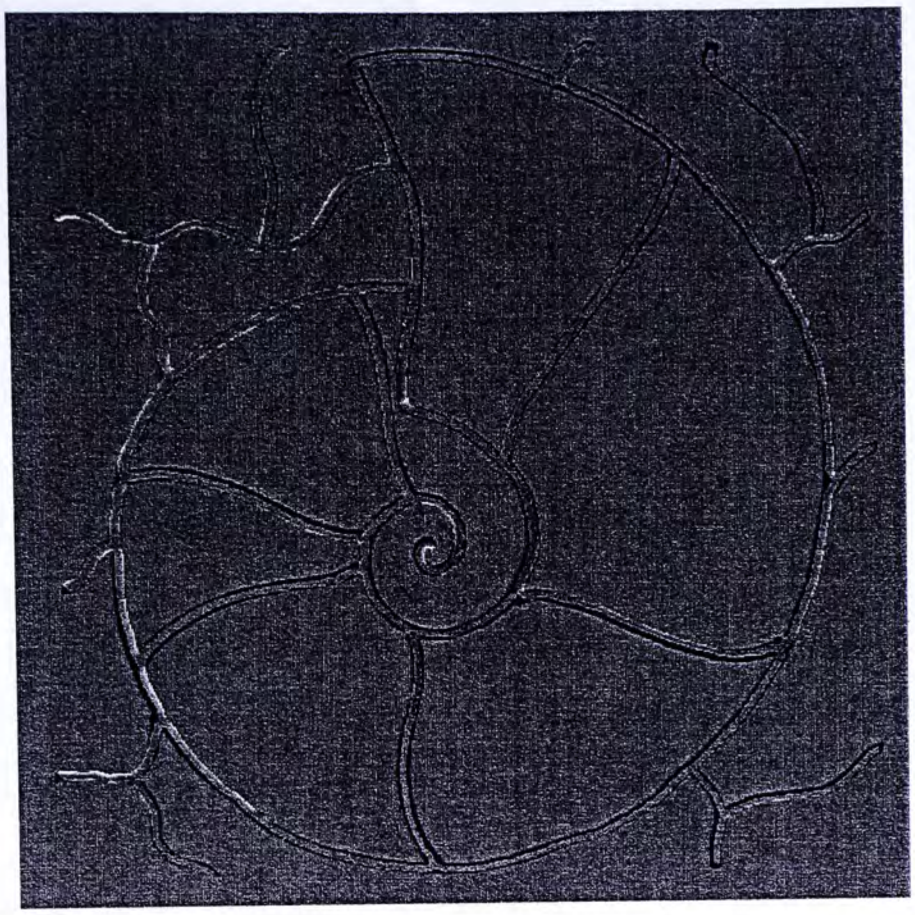


Figure 4.8. Exact overlapping qualities of the rectified executed image in white and the original image in black

Result 2

Figure 4.9 shows another overlapping result. The pixel size in this case is 580×400. The parameters are d_r=0.2 pixel units, $d_{\max}$=15 pixel units, $d_{\min}$=-15 pixel units, l_{bit}=64, p_c=0.7, p_m=0.0109, N_{ind}=70 and N_{gen}=60. Figure 4.9(a) shows the case of homography transformation based on manually selected points. The corresponding objective cost is $o(H)$=10546. Figure 4.9(b) shows the case of GA homography transformation, and the objective cost is $o(\widetilde{H})$=4176. The cost is not reduced as much as previously because the executed stroke is thicker than the original image. The offset displacement ΔP=[1.2, -5, -8.6, 4, -1, -8.8, -0.2, -11.2] pixel units.

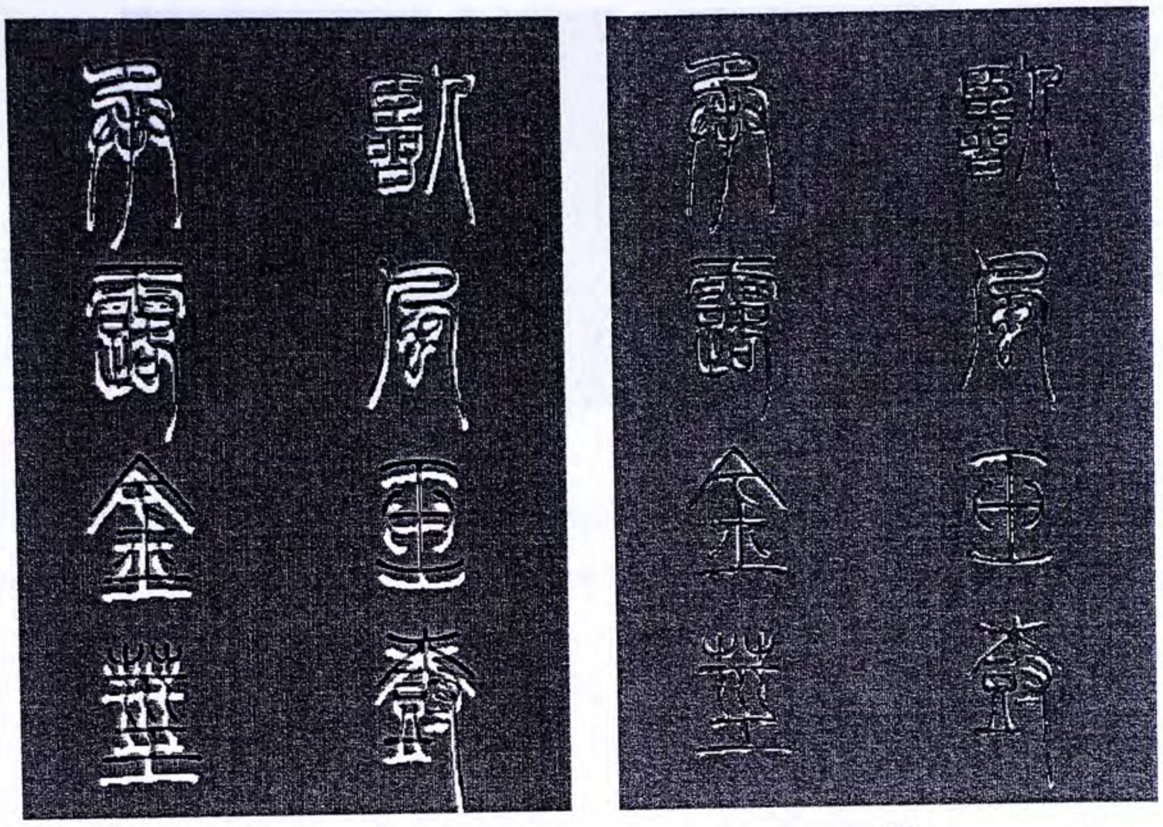

(a) (b)

Figure 4.9. (a) The rectified image before GA homography correction in white, (b) The one after correction shows the executed stroke is thicker than the original image one

4.4 Visual-based iterative correction example

The overlapping results using GA homography transformation in Figure 4.8 is now applied to generate corrective actions in the execution of branch points. This is possible because brush tip deformation during the drawing process usually yields rather significant offset in the brush lines (discussed latter in section 5.1) should improper drawing sequence be executed.

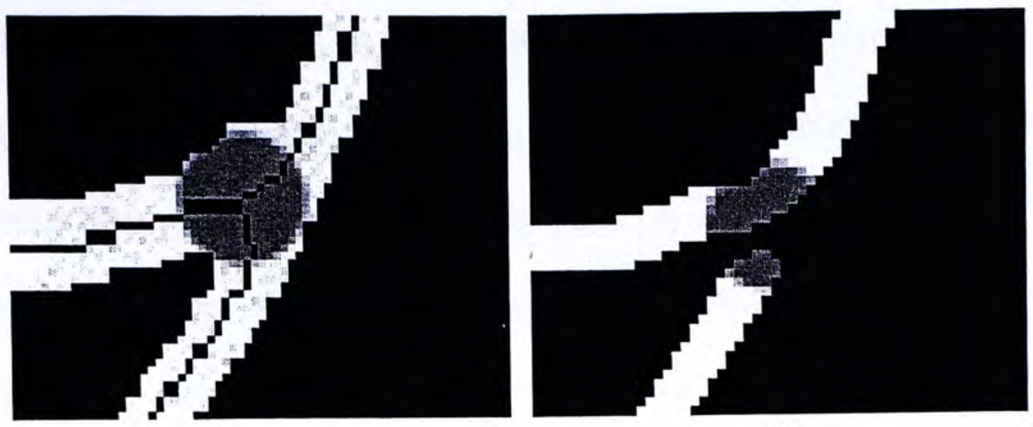


Figure 4.10. (a) A MIC centered at the skeleton branch pixel, (b) The same size disk with same coordinates formed to detect the connectivity in the executed drawing

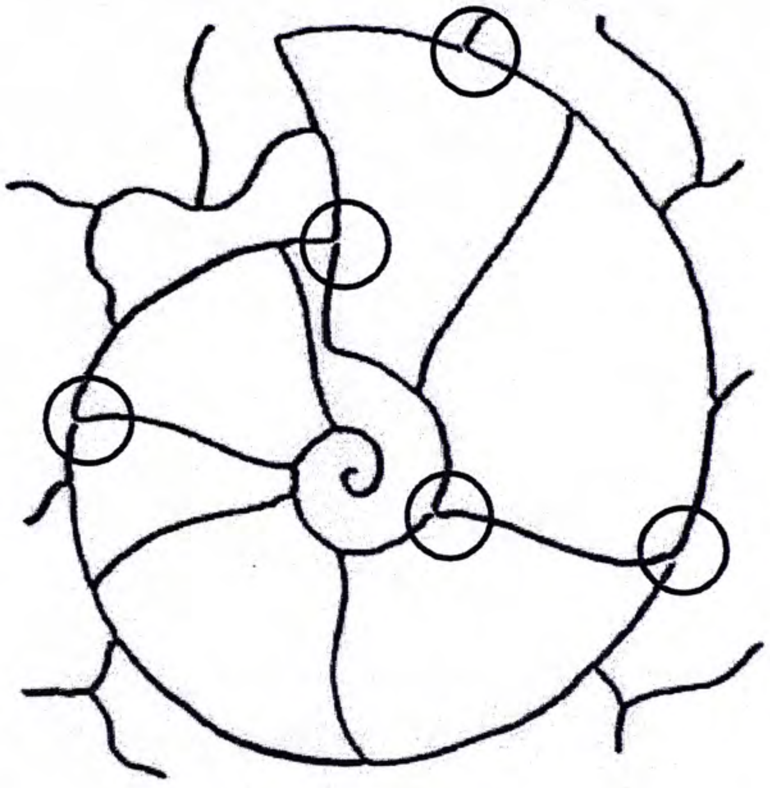


Figure 4.11. Wrong drawing sequence causes improperly executed branch points which are encircled

Focusing on the branch point regions, Figure 4.10(a) shows a circular disk centered in the stroke skeleton branch point of the original image, and Figure 4.10(b) shows the same disk formed with same coordinates in the rectified executed image. It can be seen that the stroke connectivity inside the disks of the two cases do not agree with each other. This indicates that the branch point decision needs to be corrected. Figure 4.11 shows five encircled branch points on the executed drawing image on which drawing sequences have been identified to be improperly executed. This provides the needed information for correct branch point drawing in the second robot execution. The result is depicted in Figure 4.12. The encircled branch points are now properly executed. More iteration may be conducted if the branch points are of more complicated nature than that of the present case.

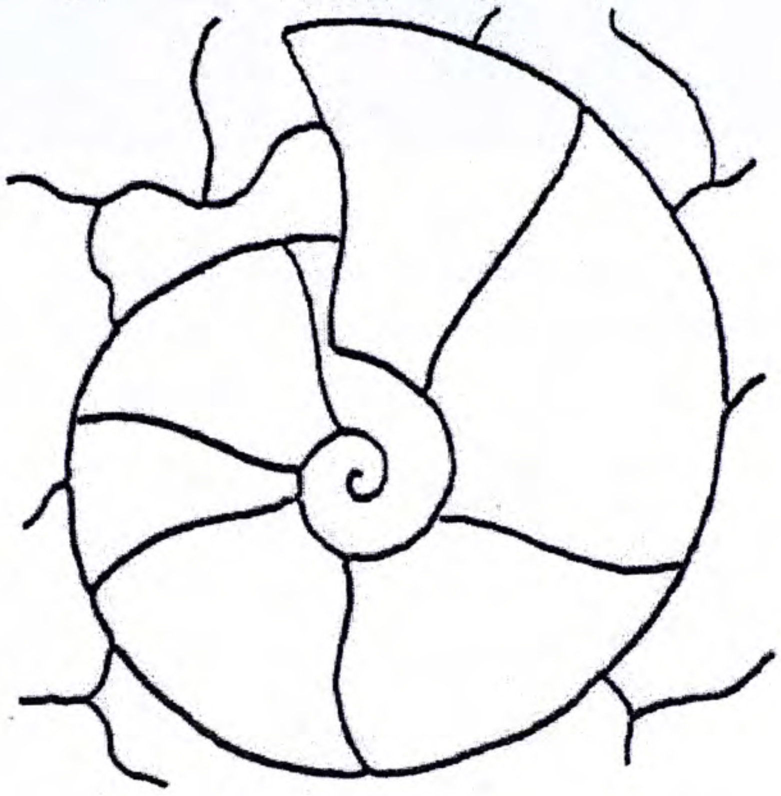


Figure 4.12. Second time executed image with corrective drawing sequence

Besides the branch point decision, other drawing and writing performance may also be measured using the vision information. Figure 4.13(a) depicts the stroke lines executed under difference z-axis (vertical to the drawing plane) values as captured by the camera at an angle. Analysis of projective rectified image in Figure 4.13(b) actually allows the calibration of executed stroke width as function of the z-axis command. The calibration curve is shown in Figure 4.14.

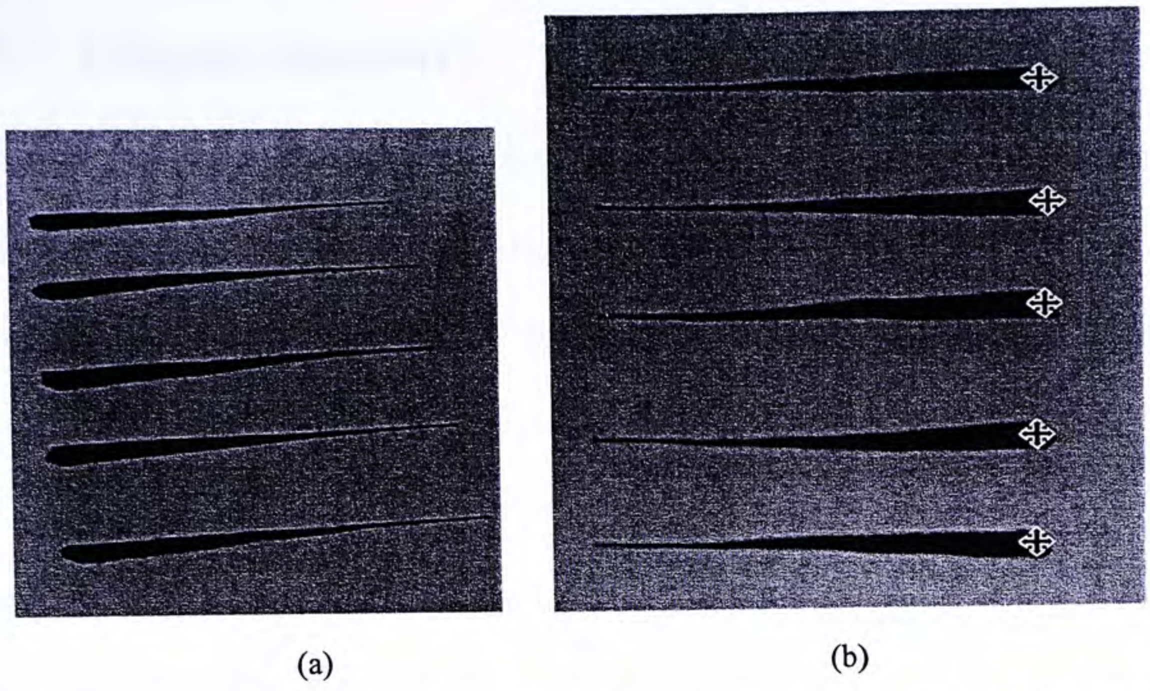


Figure 4.13. (a) The executed image as captured by camera in which strokes were executed in different z-axis value, (b) The projective rectified image for strokes thickness measurement

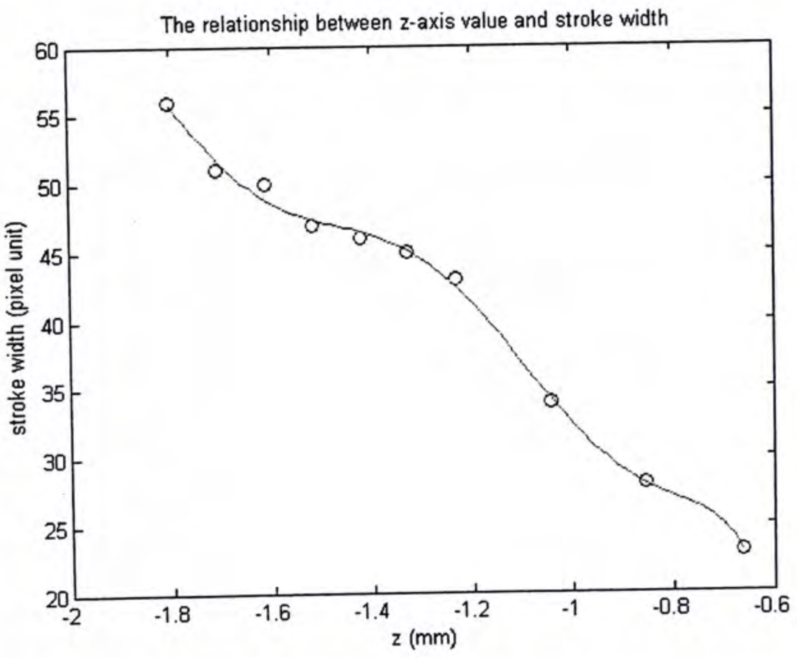


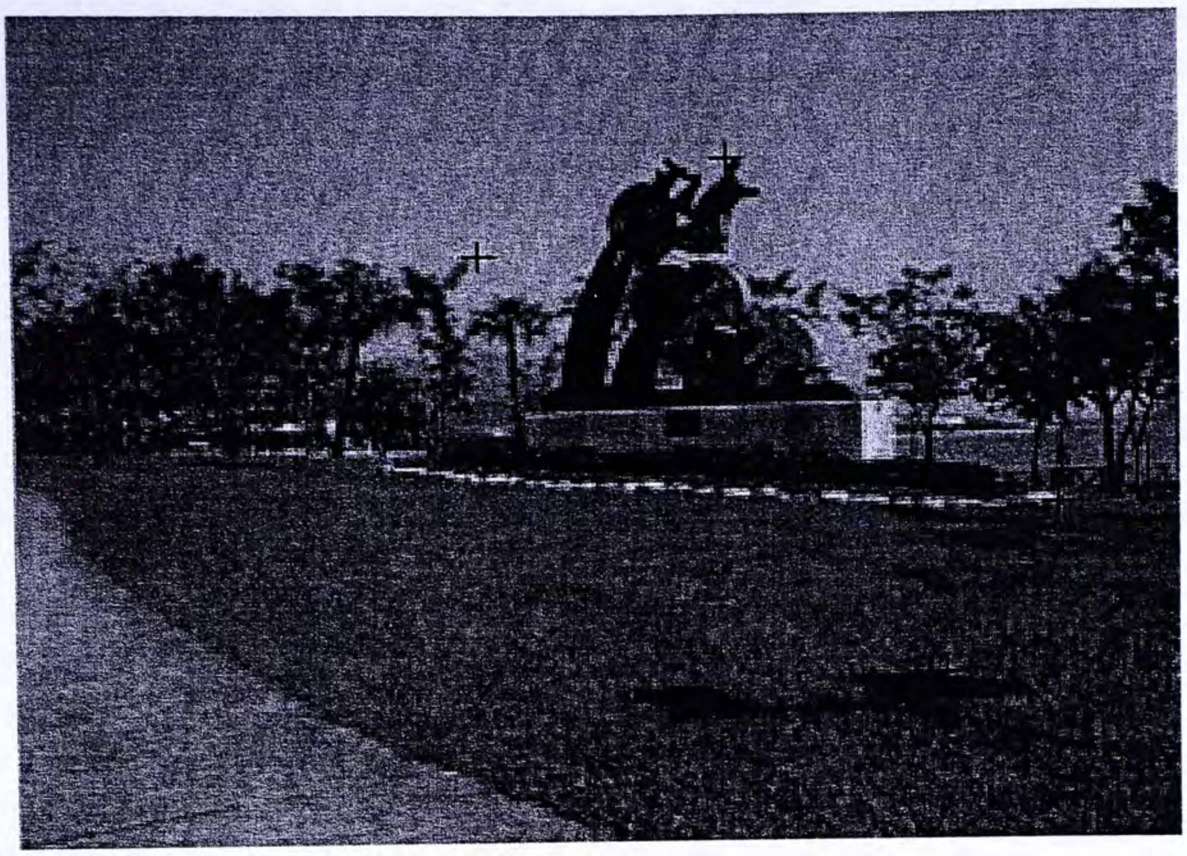
Figure 4.14. The diagram of stroke thickness against the z-axis value from vision information obtained

Application of the overlapping information to generate corrective action in branch point decision and other performances is very useful in analyzing and executing the calligraphy painting, especially for one kind of ancient Chinese font called Seal Character "篆書".

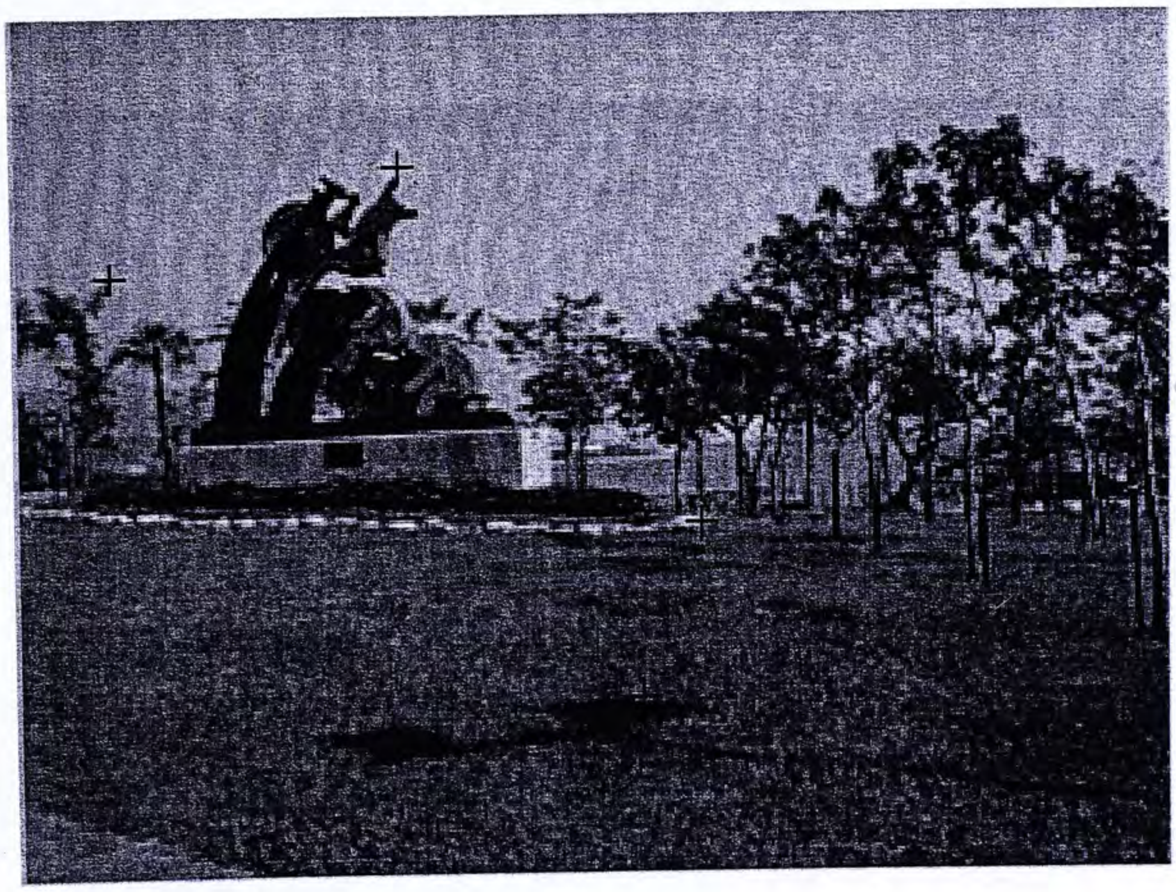
4.5 Chapter summary

This Chapter describes the development of visual-based capabilities for corrective action on improving the executed drawing by comparing with the origin. By manual picking numerous correspondences ($n = 9$) in the image, homography transformation errors may be minimized under SVD-based methods in the solutions for a constrained linear optimization problem. However, the manually and precisely picking the correspondences is difficult and frustrating at times. For close comparison as desired, a GA-based homography transformation to automatically generate a high performance overlapping upon the coarse manual selection of 4 correspondence points is developed. The result shows that the process can be successful applied in actually pinpointing incorrectly executed branch points, which are then modified in the next execution of the drawing. It is envisioned that visual-based capabilities would be expanded in the future to conduct real time online closed loop correction in thickness and trajectory control of albeit simple painting and calligraphy with full brush strokes. Also, the robot can follow the varying thickness along the stroke upon the vision feedback with image high resolution.

The concept of error compensation in homography transformation can have other applications e.g., in image mosaicing. The user just has to select four correspondence pairs roughly as Figure 4.15 (a) and (b). The superior performance of GA-homography correction over that of SVD-based homography correction is clearly shown by comparing Figure 4.16 (a) and (b).

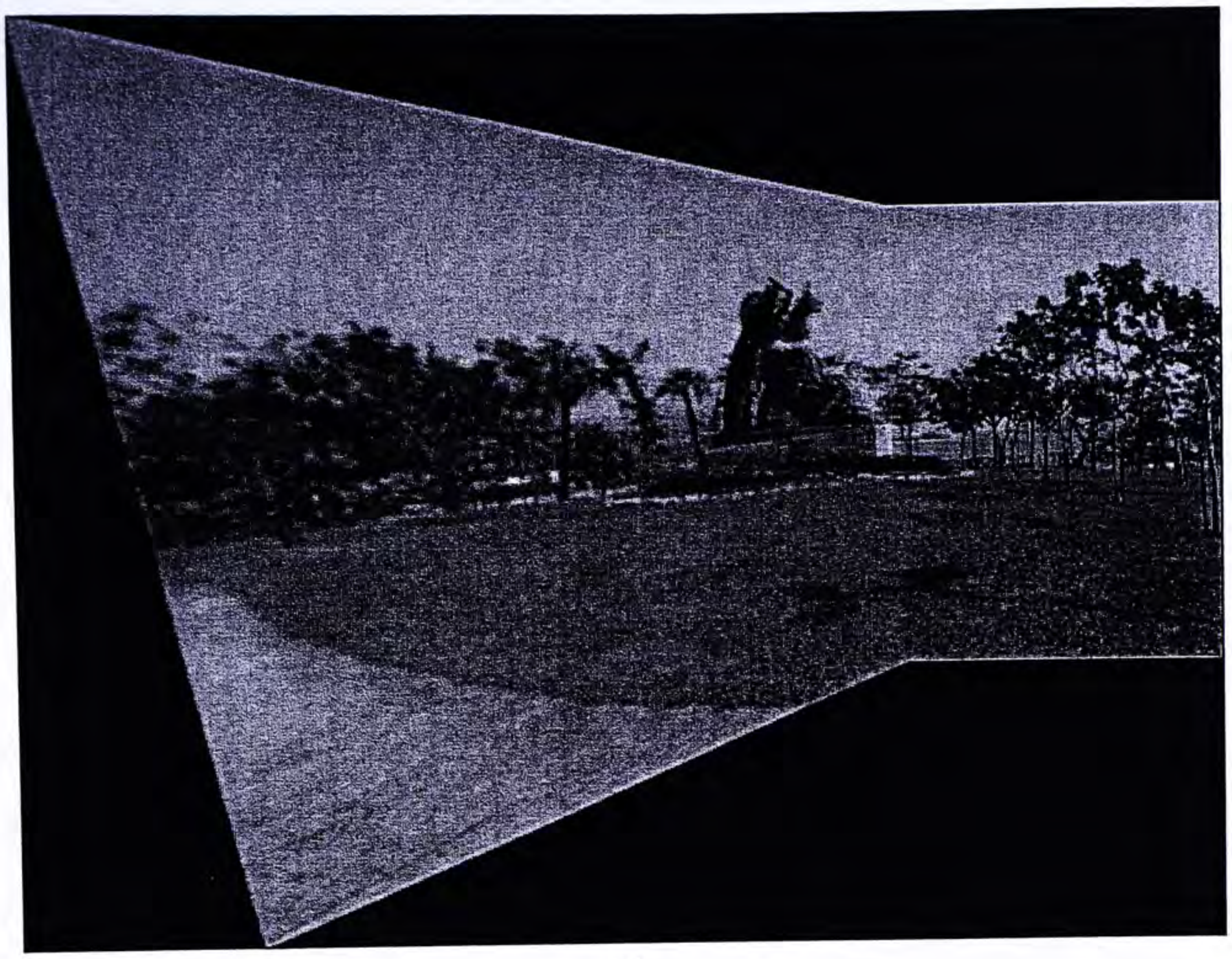


(a)

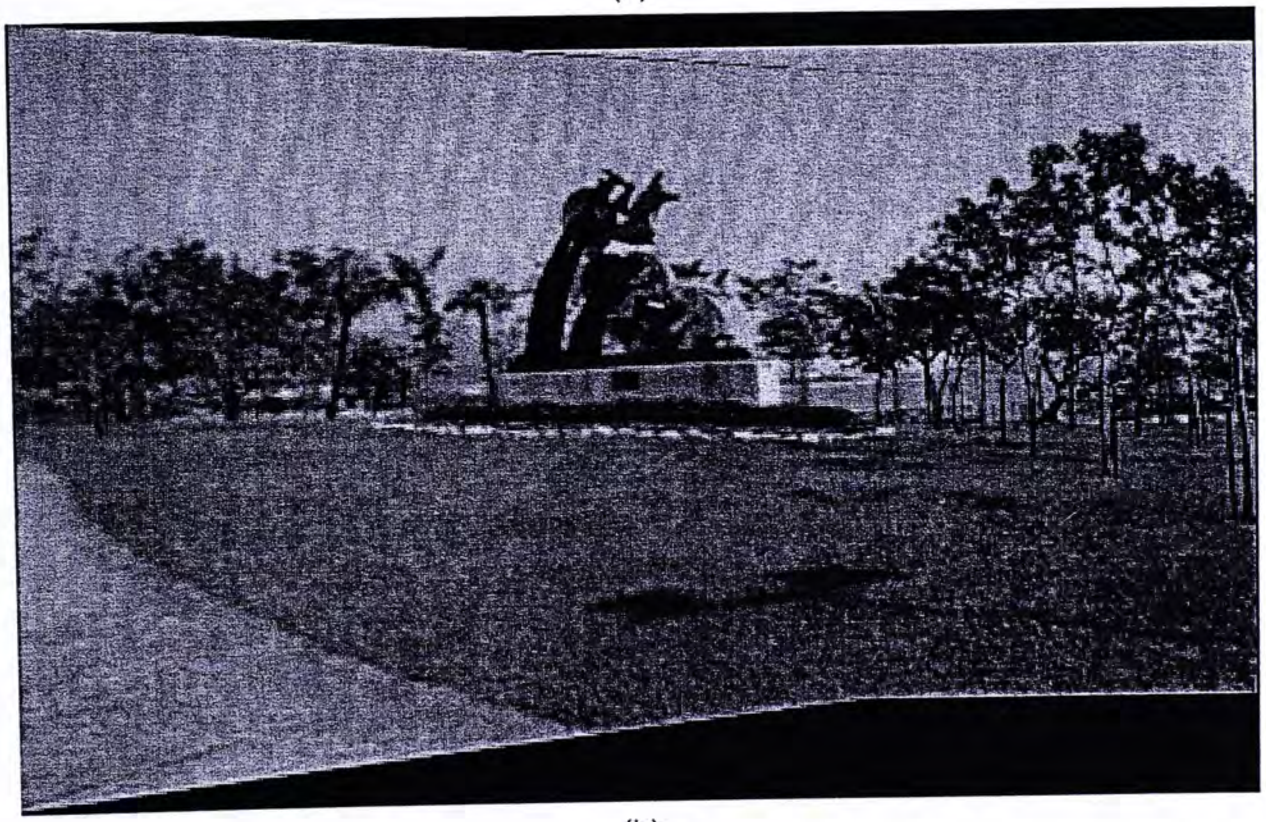


(b)

Figure 4.15. Four correspondence pairs are picked roughly in left side photo (a) and right side photo (b)



(a)



(b)

Figure 4.16. Poor quality mosaicing is shown in photo (a). After GA-homography correction, photo (b) shows the exactly correct image mosaicing

5. ITERATIVE ENHANCEMENT ON OFFSET EFFECT AND BRUSH THICKNESS

Figure 5.1. Several types of auto-ink-loading brush are used in the iterative drawing

For Chinese black ink painting, the brush, the ink, the ink stone and paper are referred to as the "four treasures in a Chinese study". Among the four, the ink brush plays an important role. Usually, the traditional bristles are made from the soft hair of animals such as black rabbit's hair, white goat's hair, and yellow weasel's hair. They measure from 5 to 6 centimeter long and are starched to form a pointed tip [39]. The brushes are classified into three groups: "Hard", "Soft" and "Both". If the tips do not have the proper elasticity and shape, the brush cannot provide stroke lines with adequate quality. For our robot drawing, the brushes with automatic ink loading are used, see Figure 5.1.

5.1 Offset painting effect by Chinese brush pen

Referring to Figure 5.2, as the pressure exerted on the brush increases during drawing, the elastic brush tip is deformed. The bristles spread out such that their contact area widens to maintain a constant total volume. As a result, the brush draws a thicker line on the paper. Moreover, when the position or running direction of the brush changes, the brush reaches its equilibrium shape only after some time delay dependent on the bristles' elasticity. This offset effect is significant for higher exerted pressure and mostly obvious at the end of the stroke as depicted in Figure 5.3. For numerous existing works on 3D virtual brush model reported in [39]-[42], the bristles are simulated as long, thin, elastic rods and the theory of elasticity is applied to model their deformation and stiffness. However, all these qualities are not easy to measure for real brush as they vary with different bristles materials used, number of hairs and size of the bristle bundle. In our robot drawing, the stroke thickness and the offset quality are nearly unpredictable due to all these uncertainties.

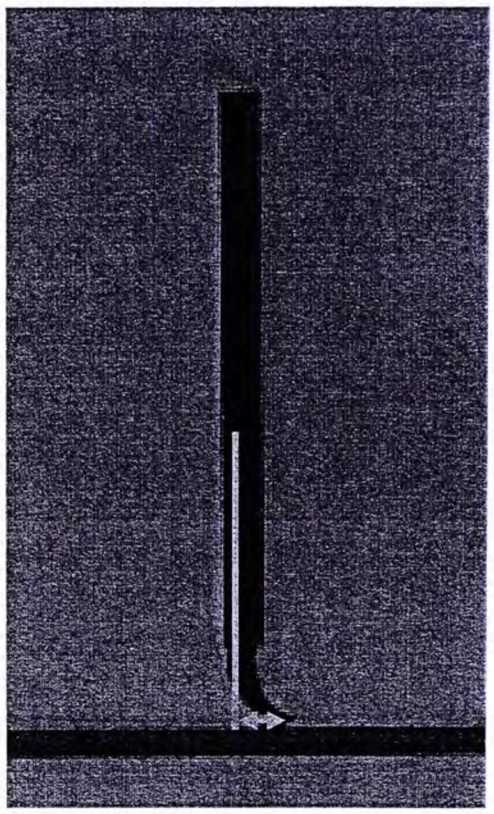


Figure 5.2. The force exerted on the brush while pulled across the paper results the horizontal deformation of the bristles

In what follows, without any precise realistic brush model we will apply, visual-based technique of chapter 4 to iteratively determine a drawing scheme to overcome the offset effect, and to determine the appropriate z-axis commands to match the actual stroke thickness of the original piece.

5.2 Iterative robot drawing process

Figure 5.4 shows the schematic diagram of the visual-based enhancement technique adopted in this section upon the input of the to-be-replicated line drawing. The robot is commanded to execute a drawing without concerning the stroke thickness and junction connectivity due to the offset effect. The executed image is then captured and the image rectified via the high accuracy GA-based homography transformation. The following steps are then carried out by comparing the rectified image and the origin.

A. Stroke thickness checking

The total stroke thickness correction can be determined based on the resulting high accuracy overlapping between the original and rectified image. For this comparison, the exact thickness quality is not necessary to computed. There are just two parameters needed to be considered: O_r is the number of rectified image foreground pixels out of the original image foreground, i.e., $O_r = \sum_j^m \sum_i^n p_{ij}^r \overline{p_{ij}^o}$, and R_o is the number of original foreground pixel out of the rectified image foreground, i.e., $R_o = \sum_j^m \sum_i^n \overline{p_{ij}^r} p_{ij}^o$, where i is the row number, j is the column number, p_{ij}^o is the pixel value of the $n \times m$ original image, p_{ij}^r is the pixel value of the $n \times m$ rectified executed image, and $\overline{p_{ij}}$ the conjugate pixel value, ($0 \to 1$ and $1 \to 0$). For $\frac{R_o}{O_r} < 30\%$, the z-axis value for the next drawing should be to exert more pressure (add a negative Δz) in other to enhance the stroke thickness as the original.

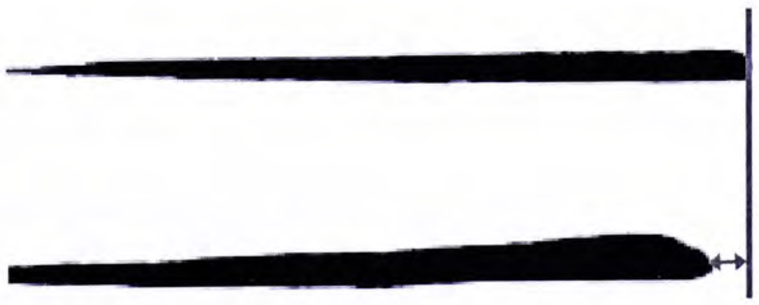


Figure 5.3. Two strokes are painted in same length but the below one which the brush is pressed more during painting indicates higher offset effect

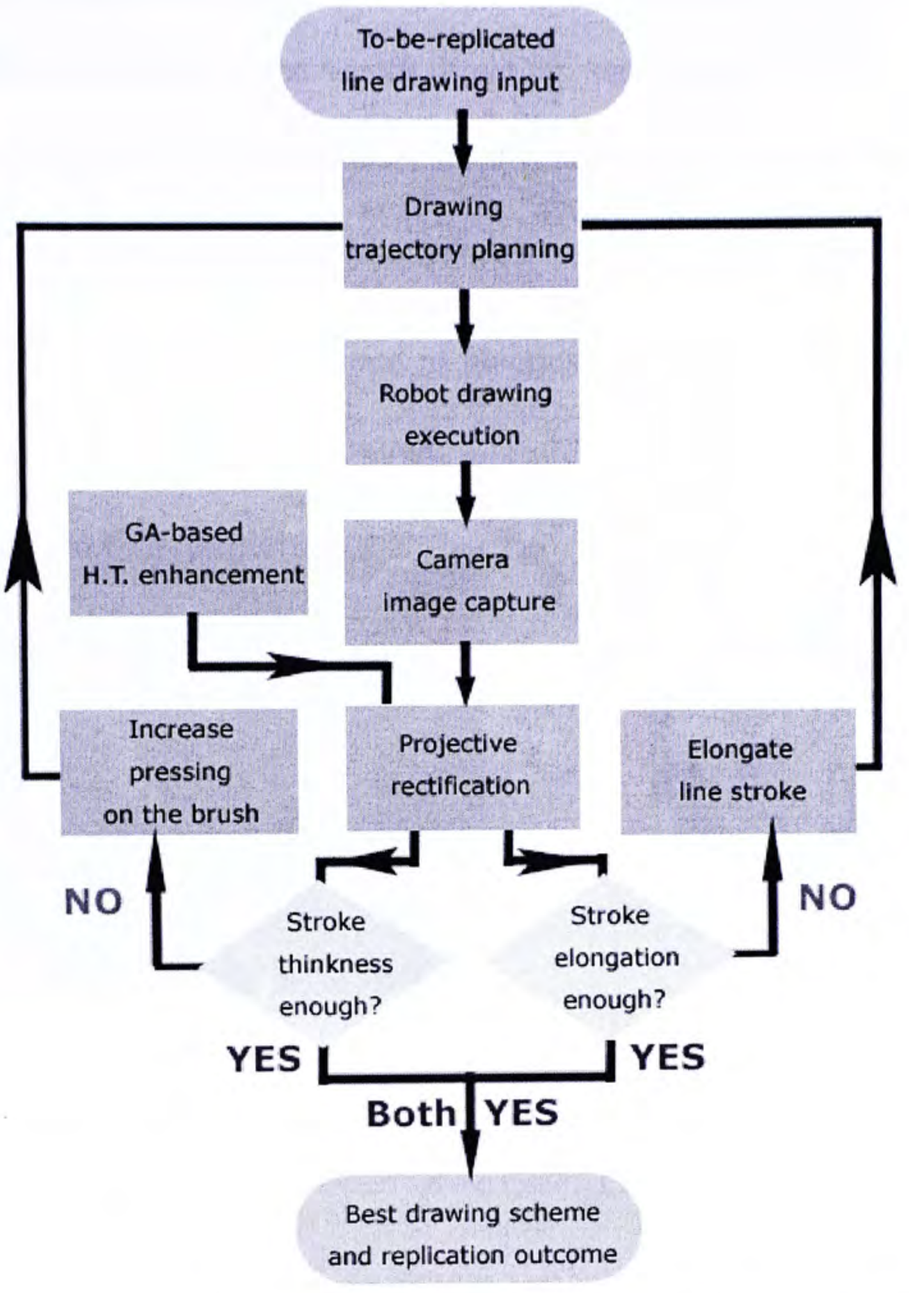

To-be-replicated line drawing input
Drawing trajectory planning
Robot drawing execution
GA-based H.T. enhancement
Camera image capture
Increase pressing on the brush
Projective rectification
Elongate line stroke
NO
Stroke thinkness enough?
Stroke elongation enough?
NO
YES
YES
Both YES
Best drawing scheme and replication outcome

Figure 5.4. The process flow chart of visual-based iterative line drawing in finding the best replication drawing scheme

B. Junction connectivity and stroke end checking

The disconnection at junctions is explicit in the preliminary drawing. Figure 5.5(a) is the skeleton image of an executed drawing after the iterative thinning process. There is no branch pixel inside the junction region. Hence, this junction is treated as disconnected and stroke elongation Δl will be performed on the open end stroke in the next iterative execution. After all junctions are verified to be connected, elongation of stroke edges is then considered. Moreover, if the end of a stroke in the executed image does not cover the edge of drawing trajectory, this stroke would be also elongated next. All the stroke elongation at junctions or at the end of stroke would follow the vector direction, see Figure 5.5(b), which has been achieved by vectorization using Bezier interpolation described in chapter 3.

Eventually, the iterative drawing process will terminate when $\frac{R_o}{O_r} \geq 30\%$ and also no disconnection that are found at junctions. Then, the last executed drawing will become the final replication output.

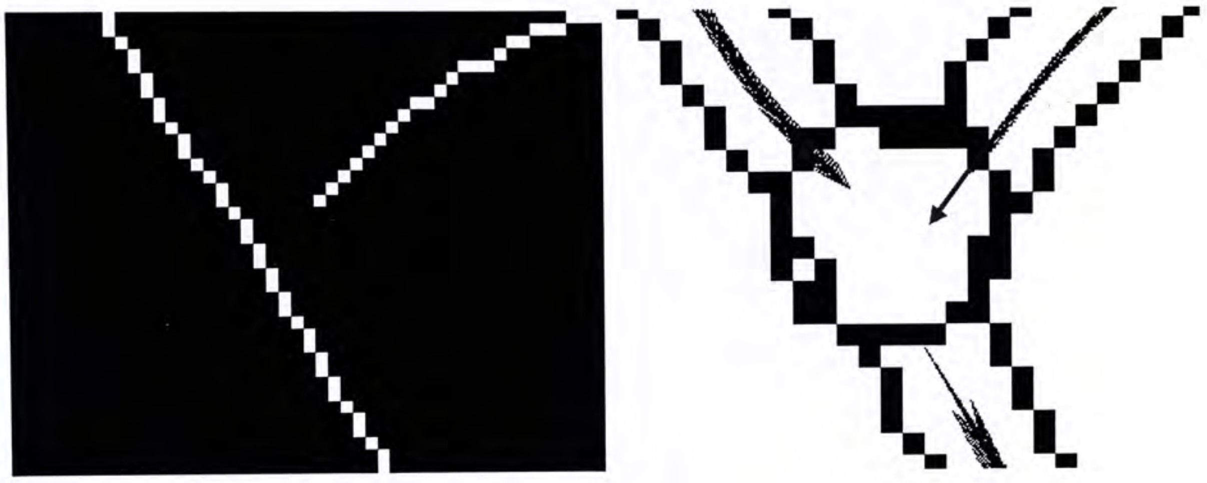

(a) (b)

Figure 5.5. (a) A skeleton image portion without any branch point inside the junction region, (b) The open end stroke would be elongated by following the vector direction

5.3 Iterative line drawing experimental results

Case study 1

The first study is on the line drawing as shown in Figure 4.2. Projective rectification is conducted using the GA-based homography matrix $H = \begin{pmatrix} 1.1245e-003 & -2.3463e-004 & -7.1173e-001 \\ 8.7572e-005 & 1.2376e-003 & -7.0245e-001 \\ 2.8468-008 & -3.7785e-007 & -1.058e-003 \end{pmatrix}$. The correspondences selected which initiate the rectification are: [(731, 506), (88, 550), (84, 54), (630, 25)] in the captured image and they are corrected by GA process as [(727.6, 506.2), (84.8, 551.4), (79.5, 49.7), (628.4, 23.1)] corresponding to four vertices [(10, 10), (590, 10), (590, 590), (10, 590)] in the original image. The picture dimension is 17.4cm$\times$17.4cm. Figure 5.6 shows the preliminary drawing scheme as generated by stroke extraction method introduced in chapter 3 overlapped with the original drawing, and the junction labels serve to indicate which junction we are referring to. Here, Δz=-3mm, Δl=3 pixel units are adopted.

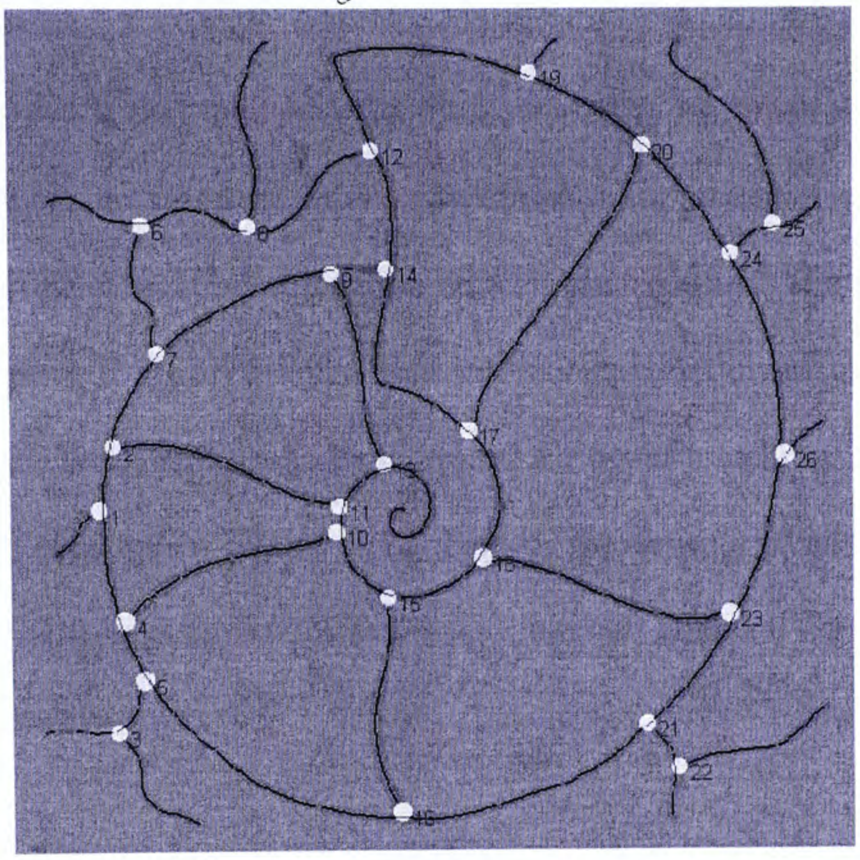


Figure 5.6. The overlapping image of the executed drawing scheme with one-pixel-width skeleton lines and the original drawing, the MIC junction regions with labels are depicted in white

Figure 5.7 shows the drawing trajectories overlapped (white traces) with original image strokes in black. As the z-axis value (z=0mm) applied on the pen is just touching on the paper, the white drawing traces are not clear due to uneven paper flatness. Upon the feedback mechanism described in section 5.2, the robot will increase the brush pen pressure and elongate the stroke at all junctions in next execution. The second execution is depicted as Figure 5.8. The white traces are now clearer, but the brush tip still cannot contact with the paper in some portion of the picture. Also, no connectivity at junctions is observed at all. More pressure and elongation are thus needed for the next execution. In third execution Figure 5.9, the white traces appear and 11 junction connectivity (at no. 2, 4, 6, 7- 9, 15, 17, 19-21) are recovered. Thus, the strokes at junction no. 1, 3, 5, 10-14, 16, 18, 22-26 need to be elongated and press more against paper during the next drawing. Figure 5.10 shows the fourth execution drawing in which nearly all junctions are recovered except junction no. 8 (which was connected in the previous execution). The explanation for this is that the larger the force exerted to the brush tip, the stronger the offset effect can be. For next drawing execution, both incrementing of the brush pressure and stroke elongation at junction no.8 are thus applied. The fifth execution is shown in Figure 5.11, all the junctions are recovered as in the original picture, but the stroke thickness still has to be increased for better matching. This yields the sixth execution depicted in Figure 5.12. Finally, the seventh execution in Figure 5.13, indicates that both stroke thickness and junction connectivity are nearly the same as the original drawing.

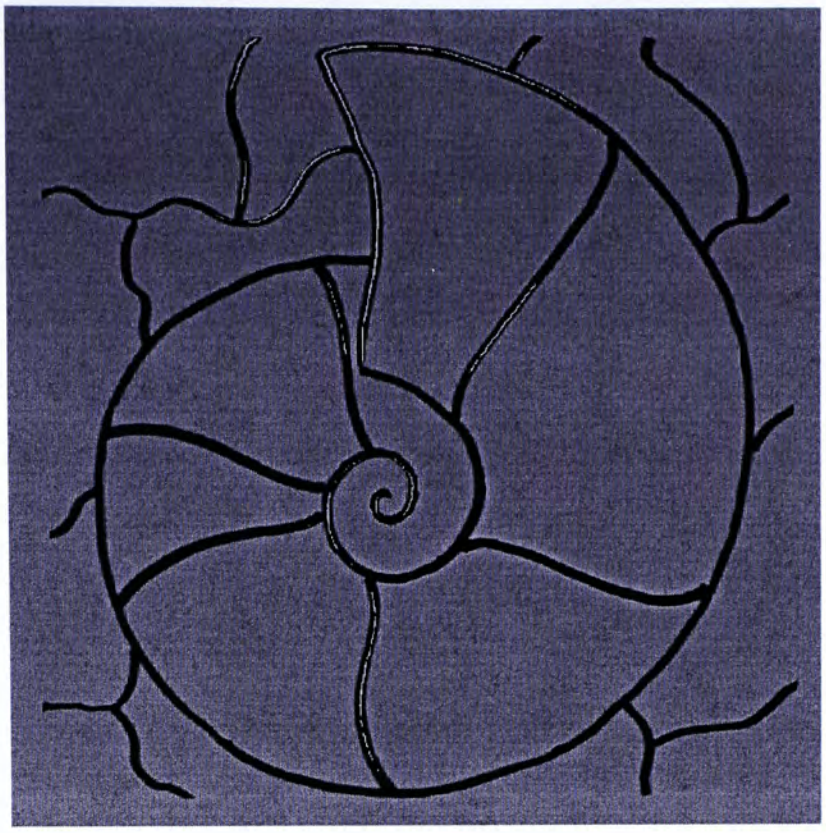


Figure 5.7. First execution from preliminary drawing scheme (z=0mm).

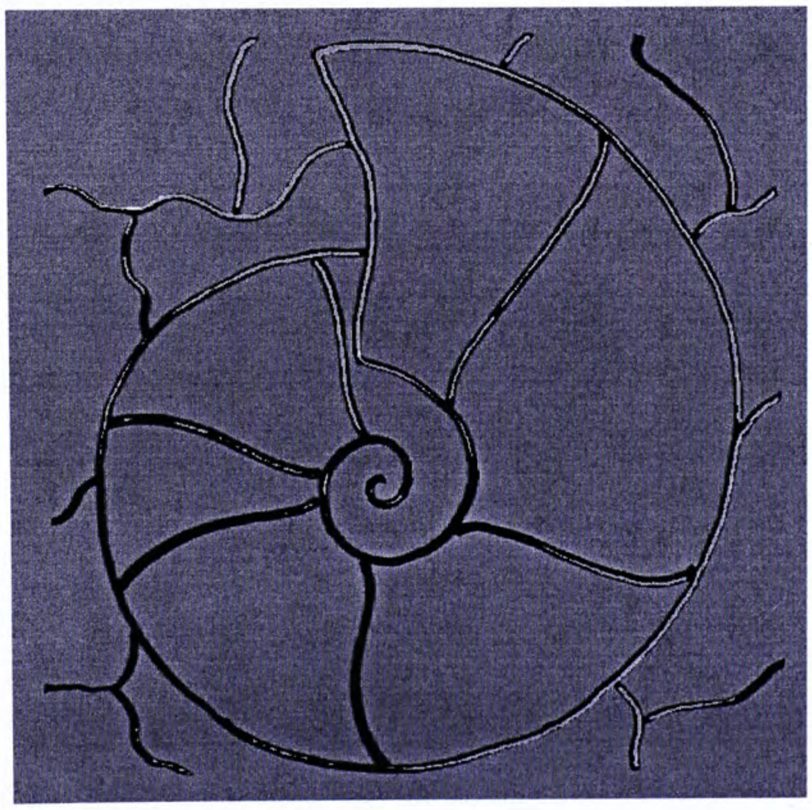


Figure 5.8. Second drawing execution: z=-0.3mm is applied on the brush pen during drawing and strokes are elongated at the junctions with 3 pixels unit.

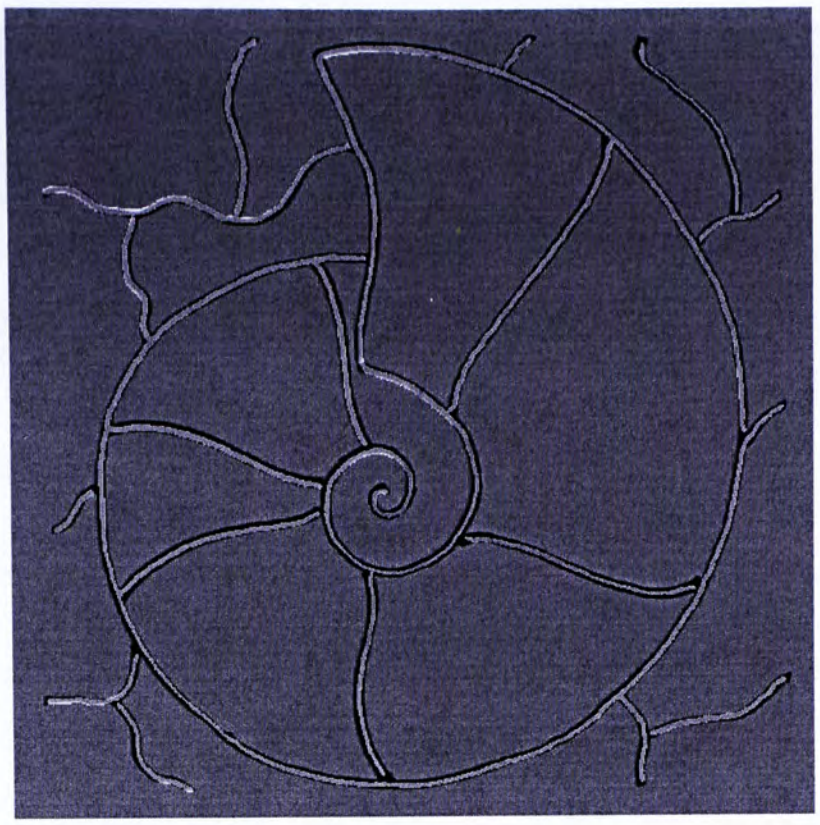


Figure 5.9. Third drawing execution: z=-0.6mm is applied on the brush pen during drawing and strokes are elongated at the junctions with 3 pixels unit. 15 junctions are still not recovered.

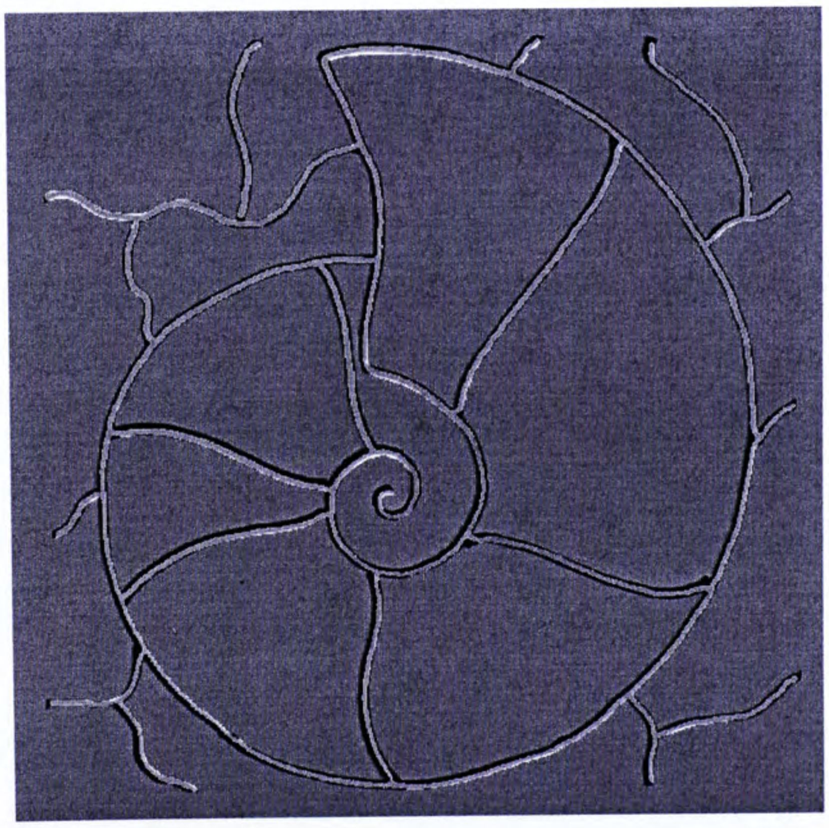


Figure 5.10. Fourth drawing execution: z=-0.9mm is applied on the brush pen during drawing and only strokes at the unconnected junctions of the third drawing are elongated. Now, only junction no. 8 is still not recovered.

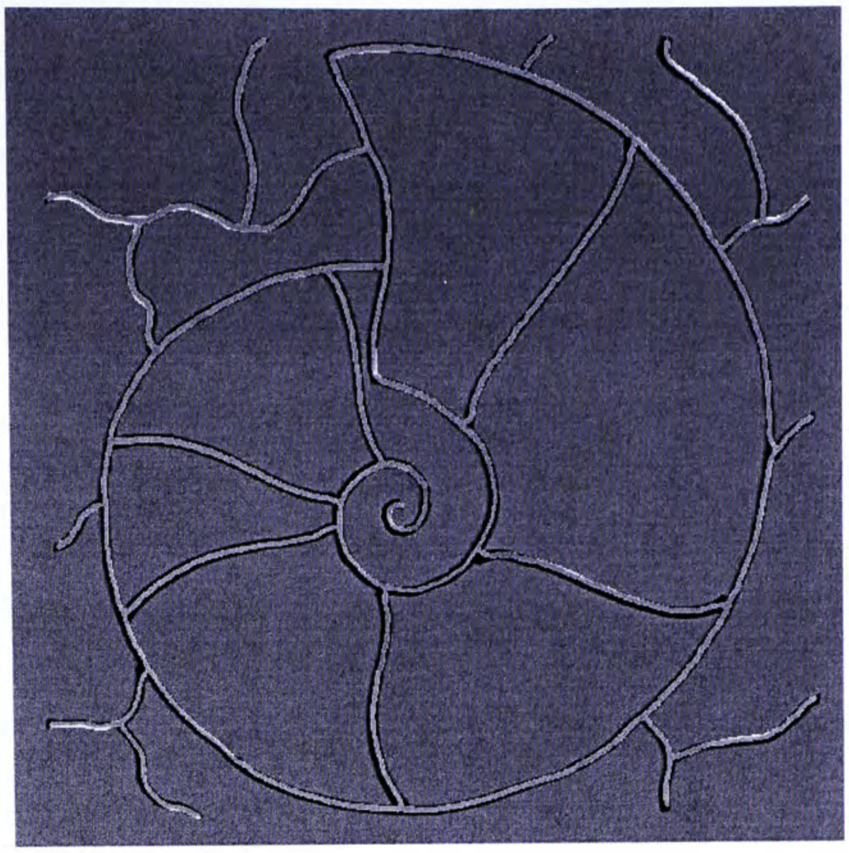


Figure 5.11. Fifth drawing execution: z=-1.2mm is applied on the brush pen during drawing and strokes are elongated at the unconnected junctions with 3 pixels unit. All junctions are connective here.

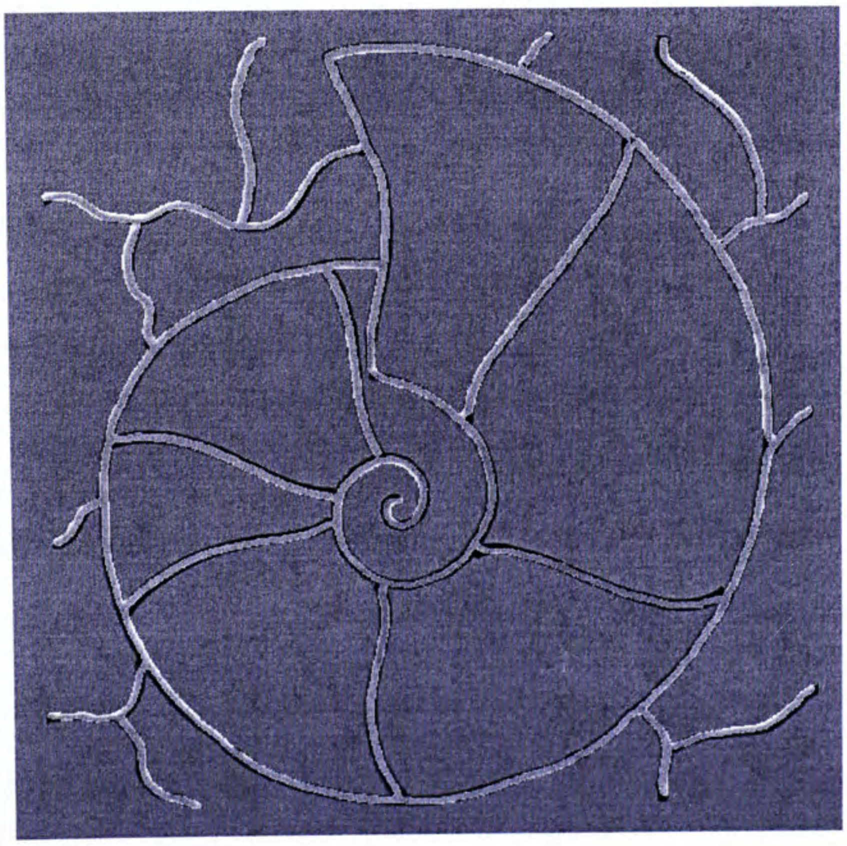


Figure 5.12. Sixth drawing execution: z=-1.5mm is applied on the brush pen during drawing and no stroke elongation is necessary.

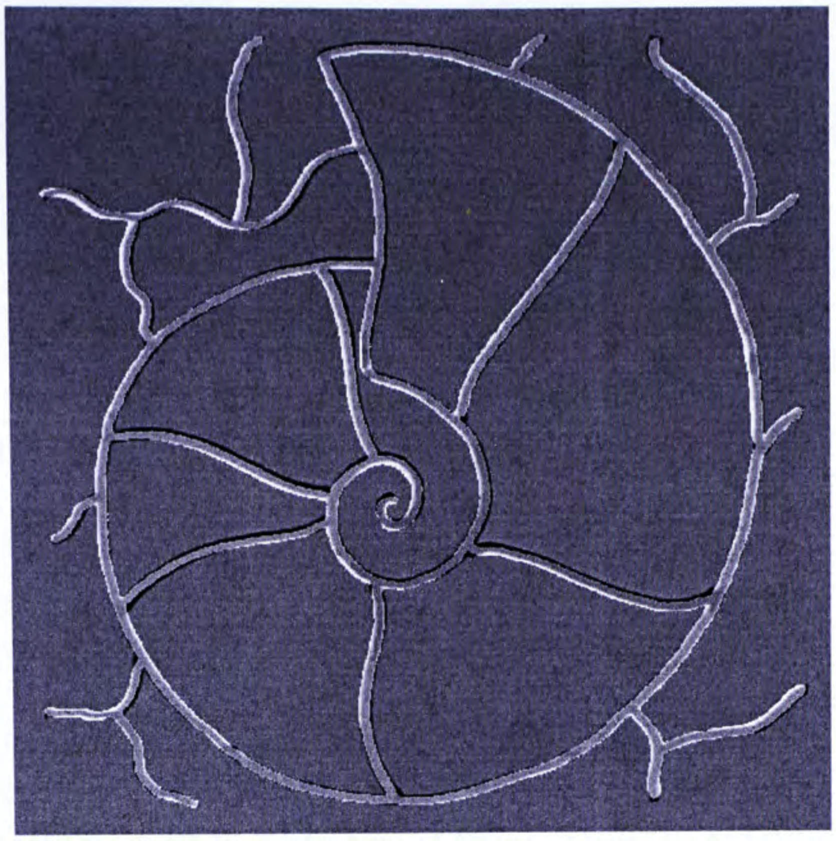


Figure 5.13. Seventh drawing execution: z=-1.8mm is applied on the brush pen during drawing. The best matching finalizes the iterative drawing process.

Case study 2

The second study is on the line drawing as shown in Figure 3.19(a). Projective rectification is conducted using the GA-based homography matrix $H = \begin{pmatrix} 1.1876e-003 & -3.004e-004 & -5.5933e-001 \\ 1.5969e-004 & 1.4208e-003 & -8.2894e-001 \\ 4.7932e-008 & -1.0846e-007 & -1.0725e-003 \end{pmatrix}$. The correspondences selected which initiate the rectification are: [(588, 507), (337, 536), (227, 45), (463, 21)] in the captured image and they are corrected by GA process as [(592.23, 509.13), (342.17, 537.13), (226.53, 40.67), (467.4, 20.57)] corresponding to four vertices [(10, 10), (284, 10), (284, 689), (10, 689)] in the original image. The picture dimension is $8.15\text{cm} \times 3.29\text{cm}$. Figure 5.14(a) shows the preliminary drawing scheme overlapped with the original drawing. Here, Δz=-0.4mm and Δl=4 pixel units are adopted. In this case, we set z=0mm such that the brush tip is pressed more on the paper to reduce the iteration, see the first execution in Figure 5.14(b).

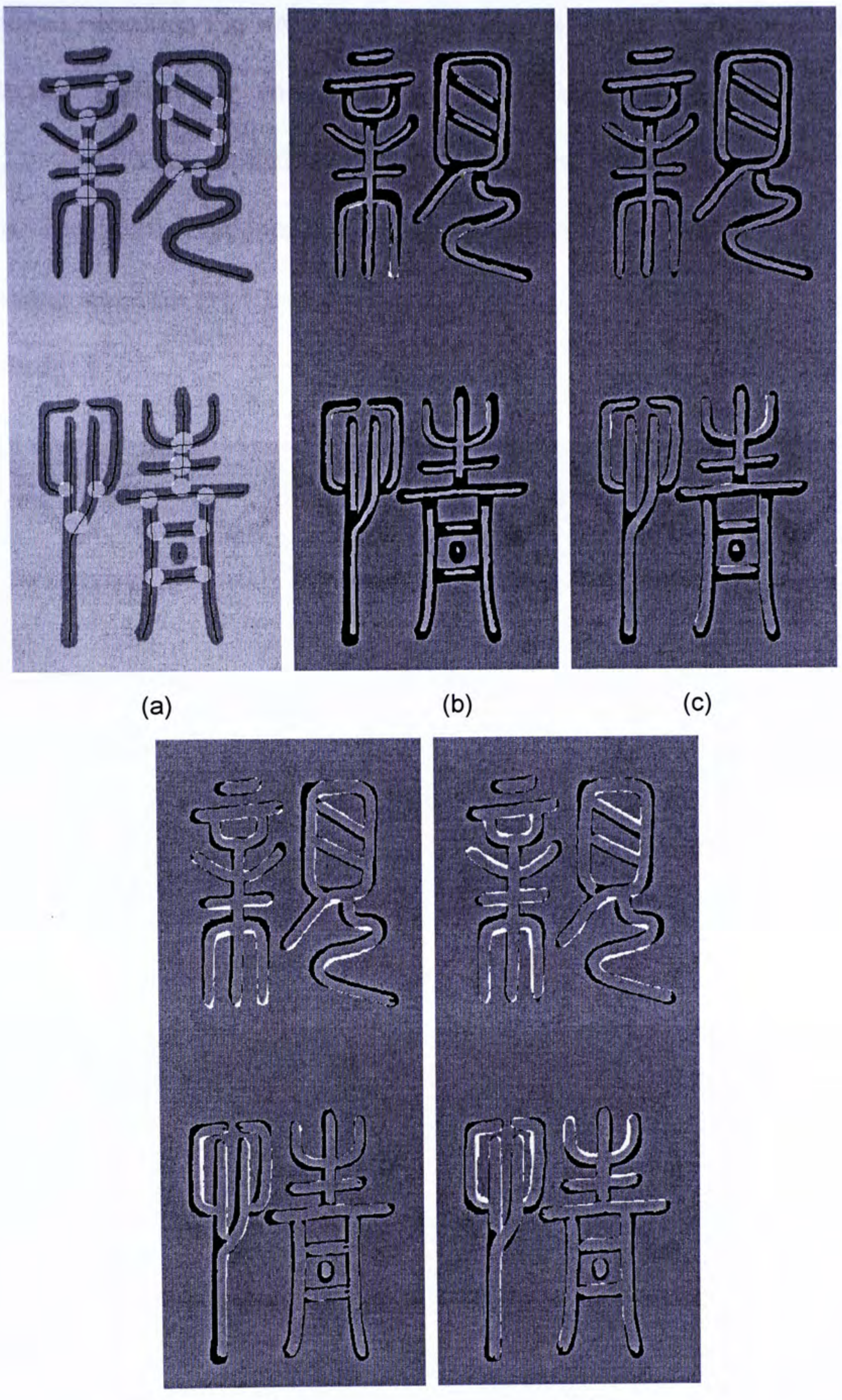
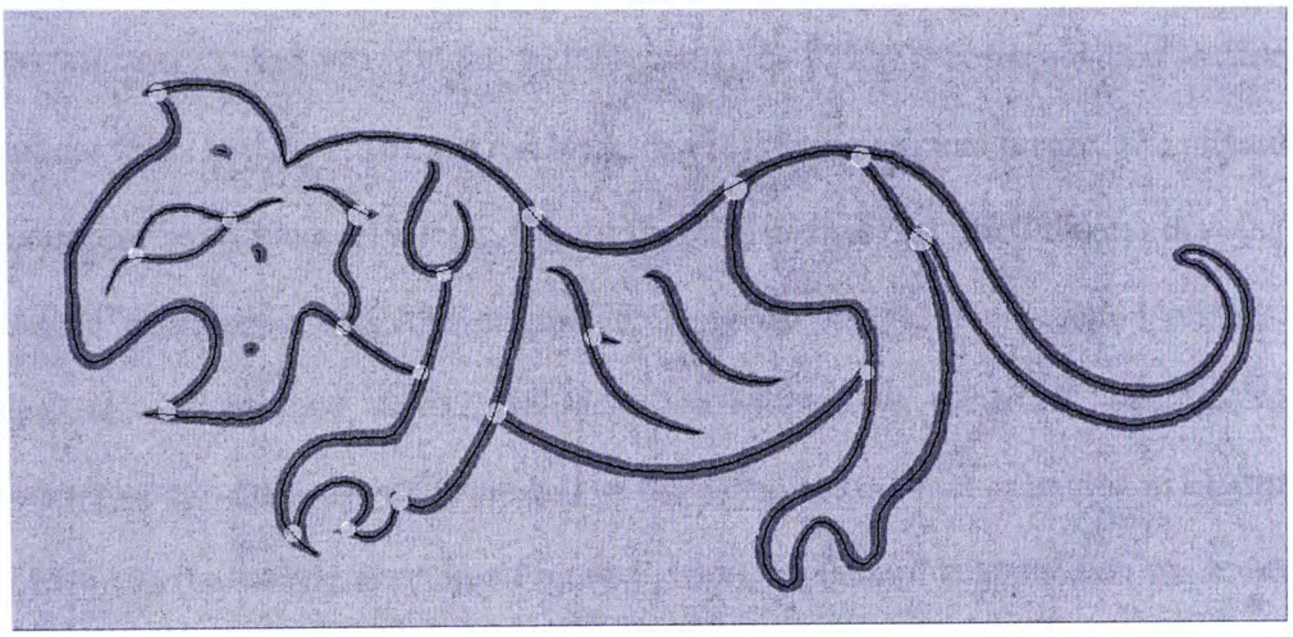


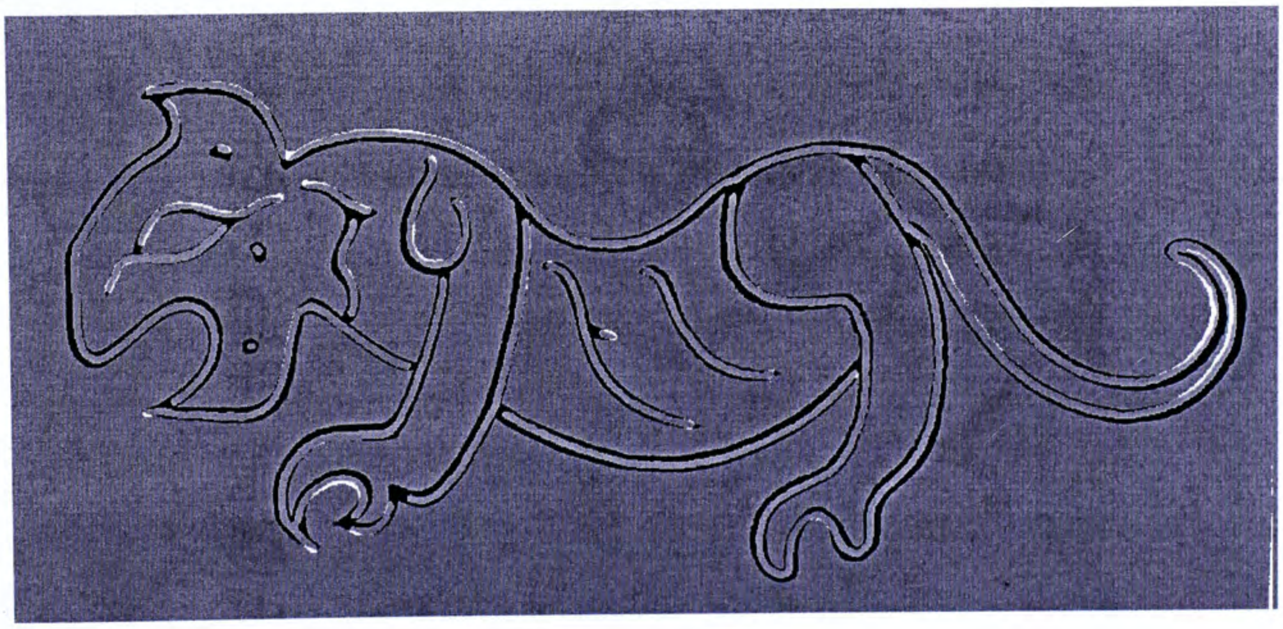
Figure 5.14. (a) The preliminary drawing scheme is shown in skeleton pixel with junction regions and the original picture display, (b) First drawing execution: z=0mm, (c), Second drawing execution: z=-0.4mm, (d) Third drawing execution: z=-0.8mm, (e) Fourth drawing execution is the best which finalizes the iterative drawing process: z=-1.2mm

In the second execution Figure 5.14(c), z=-0.4mm is applied on the brush and only cross-junction regions are connected. The third execution is shown in Figure 5.14(d), z=-0.8mm is applied and all junctions are recovered expect its stroke thickness. The fourth execution are shown in Figure 5.14e) which is the iterative final drawing result for z=-1.2mm.

Case study 3



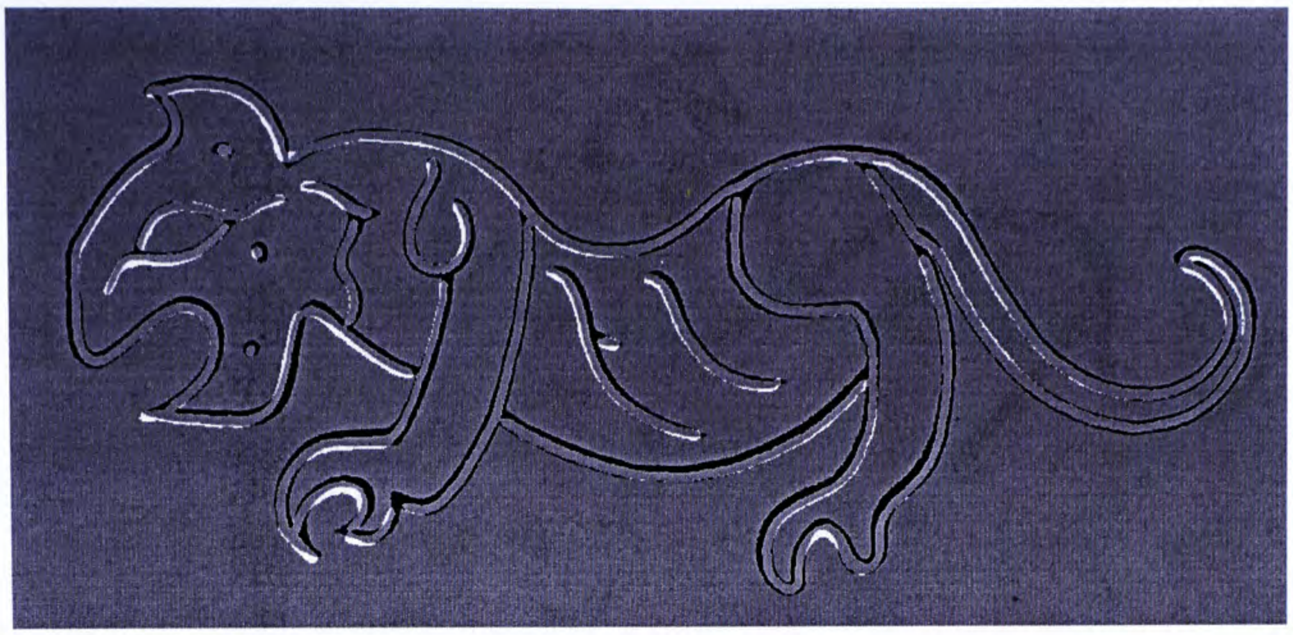
(a)



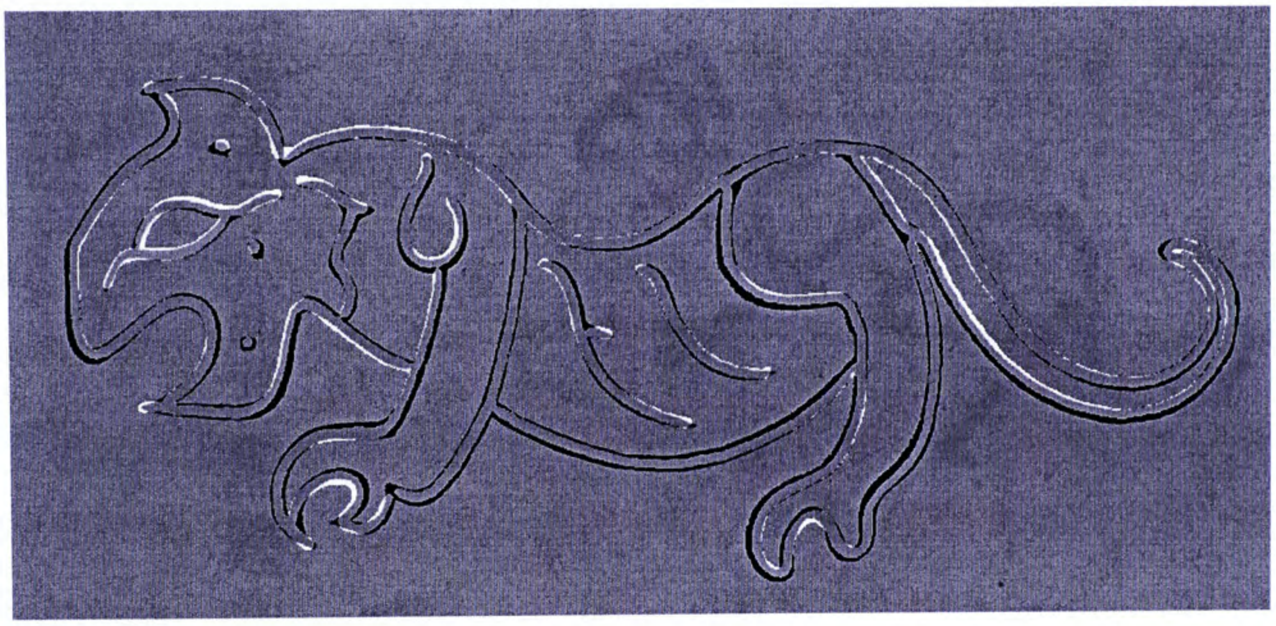
(b)

Figure 5.15. (a) The preliminary drawing scheme is shown in skeleton pixel with junction regions and original picture display, (b) First drawing execution: z=0mm is applied on the brush pen

The third study is on the line drawing as shown in Figure 3.18(a). Projective rectification is conducted using the GA-based homography matrix $H = \begin{pmatrix} -1.1761e-003 & 3.6991e-004 & 7.2985e-001 \\ 1.13156e-004 & -1.13727e-003 & 6.8361e-004 \\ -2.1342e-008 & 2.7337e-007 & 7.2695e-004 \end{pmatrix}$. The correspondences selected which initiate the rectification are: [(747, 419), (64, 485), (35, 223), (671, 173)] in the captured image and they are corrected by GA process as [(745.80, 420.50), (60.23, 485.97), (36.80, 222.70), (668.10, 171.67)] corresponding to four vertices [(10, 10), (977, 10), (977, 474), (10, 474)] in the original image. The picture dimension is $7.89\text{cm} \times 16.4\text{cm}$. Figure 5.15(a) shows the preliminary drawing scheme overlapped with the original drawing as well as the junction regions. Here, Δz=-0.3mm and Δl=3 pixel units are adopted. In this case, we also set z=0mm so that the brush tip is pressed on the paper, see the first execution in Figure 5.15(b). In the second execution Figure 5.16(a), z=-0.3mm is applied on the brush and some junction regions are connected. The third execution is shown in Figure 5.16(b), z=-0.6mm is applied. Only one stroke end at junction is disconnected and the stroke thickness still needs to be increased. The fourth execution are shown in Figure 5.17 which is the iterative final drawing result for z=-0.9mm.



(a)



(b)

Figure 5.16. (a) Second drawing execution: z=-0.3mm and junctions elongation are need in the next execution, (b) Third drawing execution: z=-0.6mm and some junctions are still disconnected

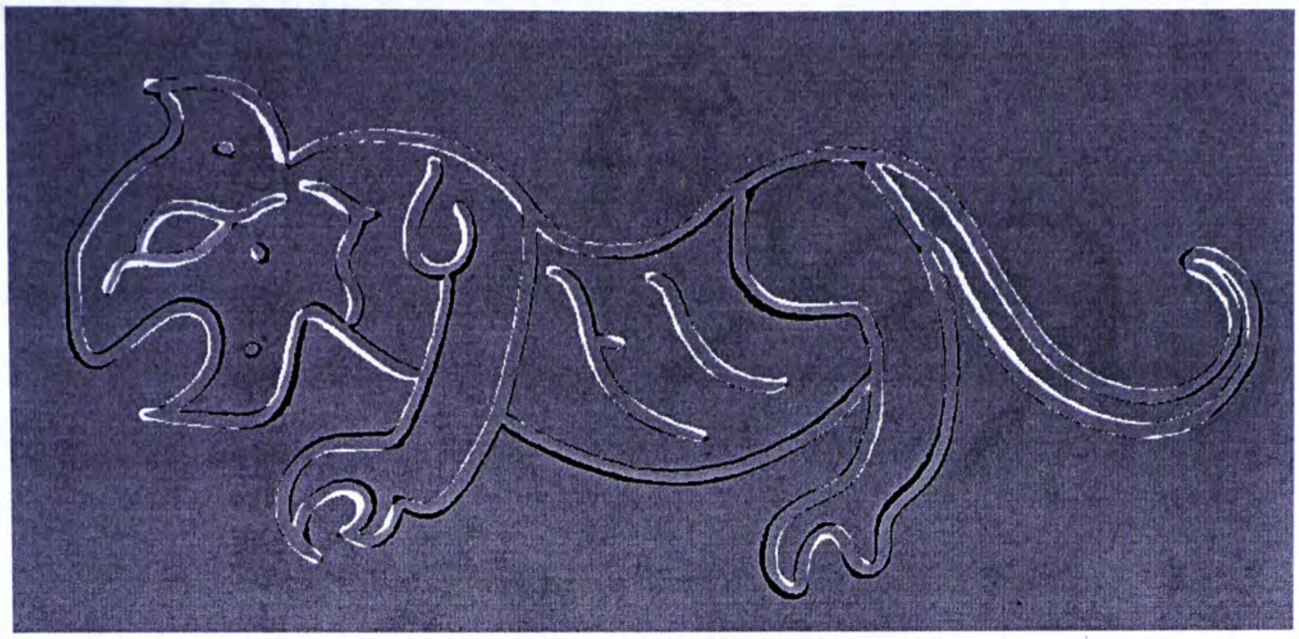


Figure 5.17. Fourth drawing execution is the best which finalizes the iterative drawing process: z=-0.9mm.

5.4 Chapter summary

In this chapter, we extent the iterative enhancement to more details of the drawing; namely, brush width and offset effects. The method depends on the high overlapping quality as resulted in our visual capabilities. Even without adopting a brush model, iterative drawing produces a better and better fine tuning in replicating the original drawing. The results hence show that our approach can improve the total stroke thickness as well as the stroke connectivity. Our future goal is to modify the iterative drawing algorithm such that the robot can follow the varying thickness along the strokes upon the projective rectification of high pixel resolution. At the same time, however, we are also in the process of experimentally determining a simple brush pen model to be used for qualify robot drawing.

6. GA-BASED BRUSH STROKE GENERATION

All Chinese characters are built up from basic strokes. Unlike other visual art techniques, all calligraphy strokes are permanent and incorrigible, demanding careful planning and confident execution. One example of such a complex process is shown in Figure 6.1.

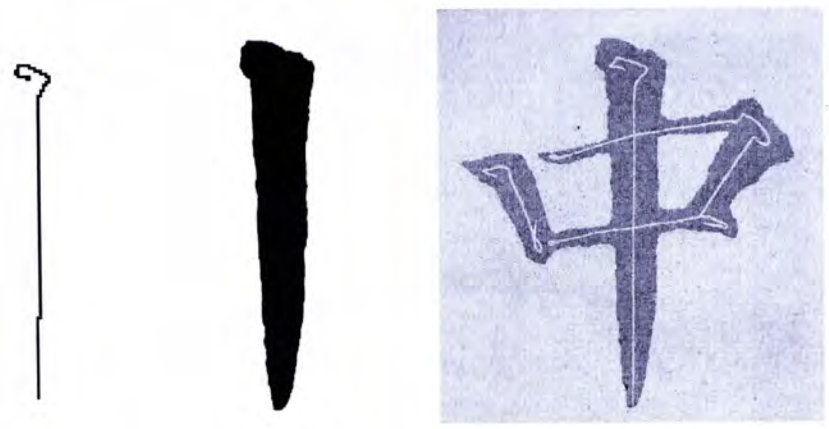
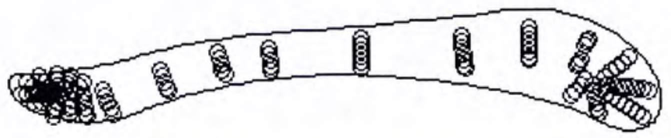
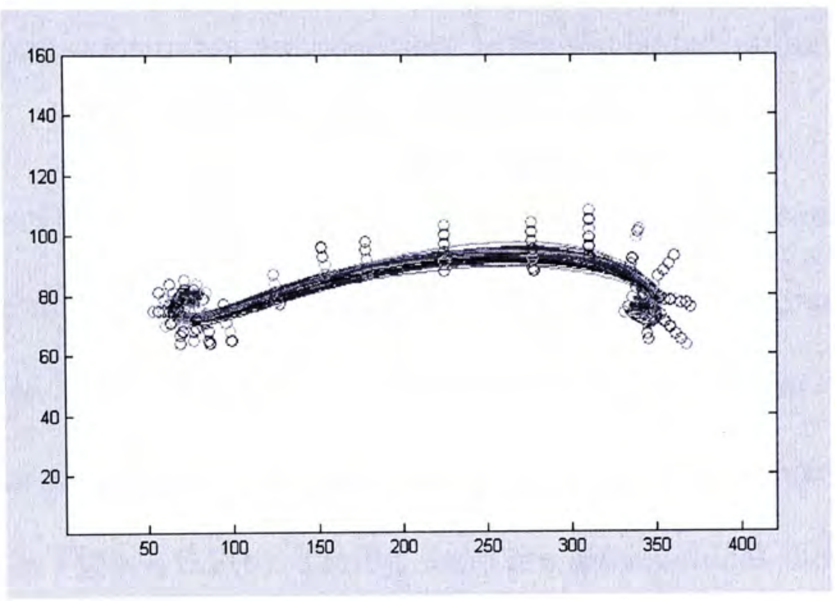


Figure 6.1. (a) Stroke painting trajectory planned by the calligraphy artist, (b) A full stroke painting under the trajectory designed in (a), (c) The calligraphy character "中" is completed from those full stroke painting

Inevitably, good stroke control contributes highly artistic quality of the calligraphy. By controlling the concentration of ink, the thickness and adsorptivity of the paper, and the flexibility of the brush, an artist is free to produce an infinite variety of styles and forms [43]. This chapter introduces a novel brush stroke generation scheme using Genetic Algorithm (GA) under a pre-defined brush stroke model. For now, we let the brush stroke be defined only by (x, y, pressure) commands. While all degree of freedom will be attempted in the future, nevertheless, such kind of information will still give us certain perspectives about robot manipulator kinematics control for full stroke painting. This artistic stroke generation also provides future exploration about the extension of our project main theme to full stroke painting incorporating with a real brushsince, and different GA's painting scheme would be executed and evaluated iteratively until the best painting solution outcome.



(a)



(b)

Figure 6.2. (a) 121 control points from 23 groups formed in the sequence of line segments, (b) A bundle of Bezier curves represent various painting trajectories

6.1 Brush trajectory representation

The study of Bezier curves falls under the general topic of curve fitting. However, these curves really do not have many scientific purposes. That is, given some data points, a scientist would not use Bezier curves to approximate a function definition for the data. Rather, Bezier curves have more of an artistic purpose. Originally used by car designers to create pleasant looking curves, these methods are now used by graphics artists in many fields where the generation of curved shapes is a necessity. In this case, it provides us a simple representation model for the brush stroke painting trajectory. We describe the trajectory as a sequence of 2D discrete control points shown in Figure 3.11. In Figure 6.2(a), those control points which are grouped into various length line segments drawn manually determine a Bezier spine curve, and a set of pre-defined brush template centered along to the curve to model a brush. For fully automatic process, we would like to simulate the stroke rendering in finding the painting scheme by just given the sample stroke image. Thus, it is envisioned that the sequence of line segments inside the stroke for Bezier curve control point constraints are necessary to be generated rather than manually pre-defined.

Similar to section 3.4, we choose Bezier curve to represent the painting trajectory rather than B-spline or spline curve based on its original properties as mentioned before. By varying the control point, different Bezier curves will be formed as the painting trajectory which varies with different combination of control points depicted in Figure 6.2(b). Totally, there are astronomical sums of trajectory combinations (1.9508×10^{16} 2D trajectories) can be achieved in this case. The complexity of this representation is low rather than using the connected pixels to represent it.

6.2 Brush stroke modeling

In addition to brush trajectory, the brush stroke model is another vital factor to achieve good painting result. Recently, Nelson S.-H. Chu et. la. [41]-[42] developed a 3D brush model consisting of a brush skeleton and surface. Such brush model, shown in Figure 6.3, is capable to simulate the brush flattening and bristle spreading due to brush bending and lateral friction exerted by the paper surface. By simulating the deformation of Chinese brushes and the ink decomposition during the painting process, a virtual painting can be generated with different effects.

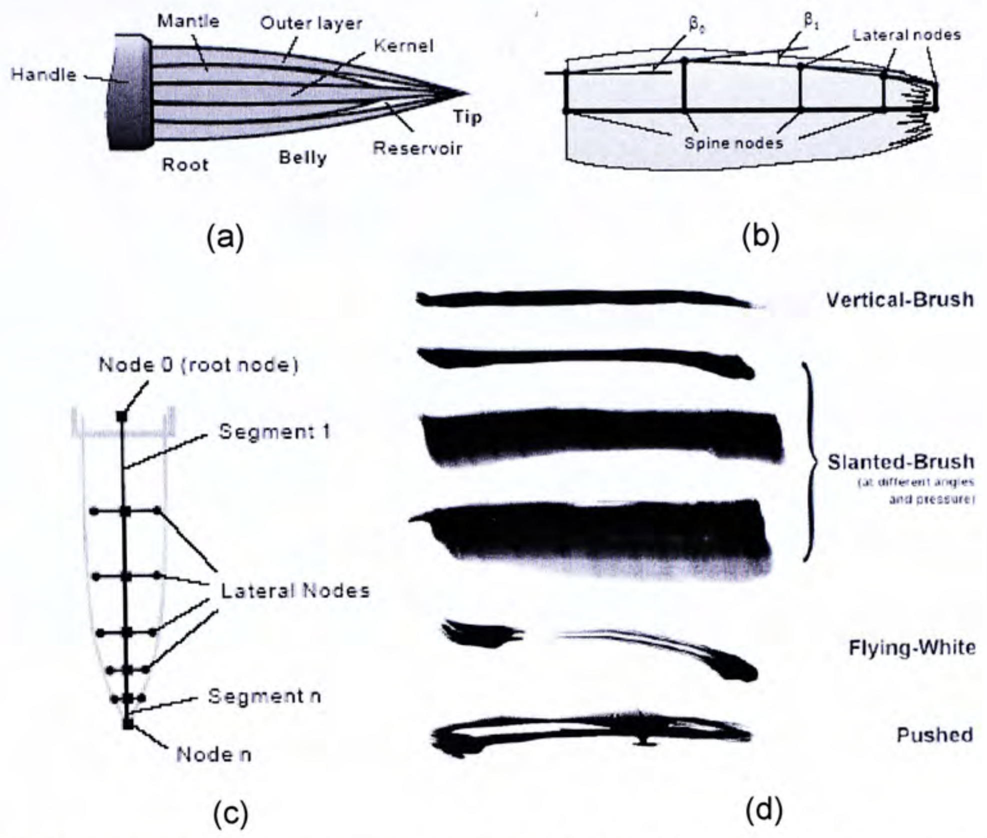


Figure 6.3. (a) Anatomy of a typical brush, (b) Blending string of lateral notes, (c) Geometric model of a brush tuft, (d) Different sample strokes are rendered by Nelson's brush model (Source: [41])

In Chapter 7, we will describe a preliminary experiment which tries to obtain stroke model using a real brush tuft. As part of the overall study, we would like to use a very simple brush 2D template model here to generate the brush trajectory.

In our case, brush pen models can be simplified as brush 2D template. By following the basic painting phenomenon, greater the writing pressure, larger the footprint painted on the paper. The 2D brush templates shown in Figure 6.4 are pre-defined. A totally 50 different sizes are stored in

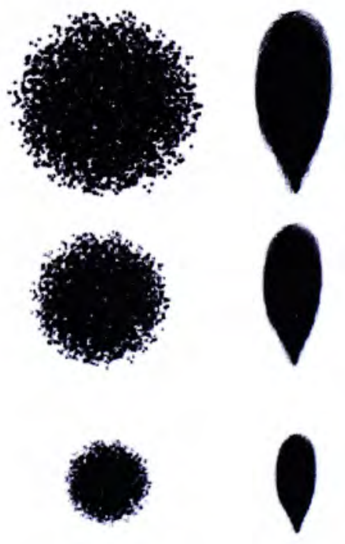
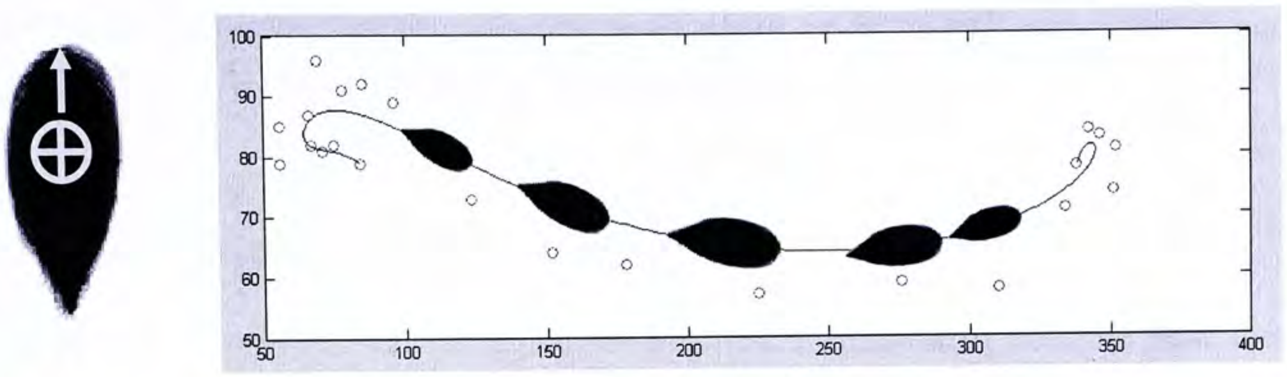


Figure 6.4. Two different kinds of 2D brush template under different painting pressure descending from up to down.

our template library. Figure 6.5 shows that the brush templates in different sizes as according to the pressure applied along the Bezier curve trajectory, assuming that the size of the template is proportional to the pressure evaluated.



(a) (b)

Figure 6.5. (a) The centroid and front direction of the template are defined, (b) The brush templates chopped tangentially along the Bezier curve under different pressure applied

6.3 Stroke simulation using GA

Unlike line stroke drawing, there is no trivial solution for full stroke painting. In what follows, we proposed GA to determine the (x, y) trajectory and z-direction pressure for a given brush stroke serves to simplify the searching process rather than studying the geometry of the brush and sample stroke given. Furthermore, one can use various kinds of 2D brush template and study how they would achieve the corresponding stroke generation. At the same time, one may judge how different brush templates would affect the chromosome resulting artistic quality.

(i) GA Representation for stroke generation

In GA representation, the chromosome can be regarded as the brush painting solution that picks $C_{i=1,\dots,m}$ as the Bezier curve control point genes and $P_{i=1,\dots,n}$ tangential pressure genes, see Figure 6.6.

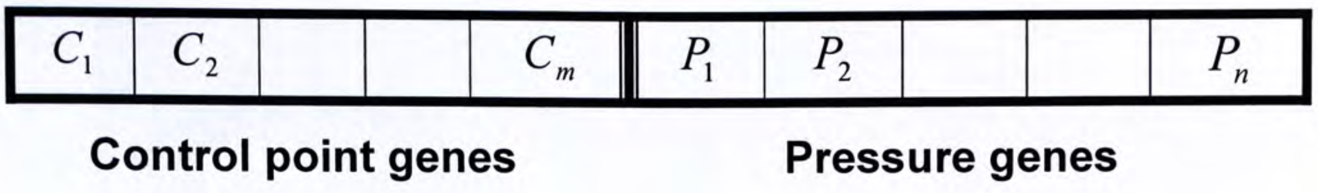


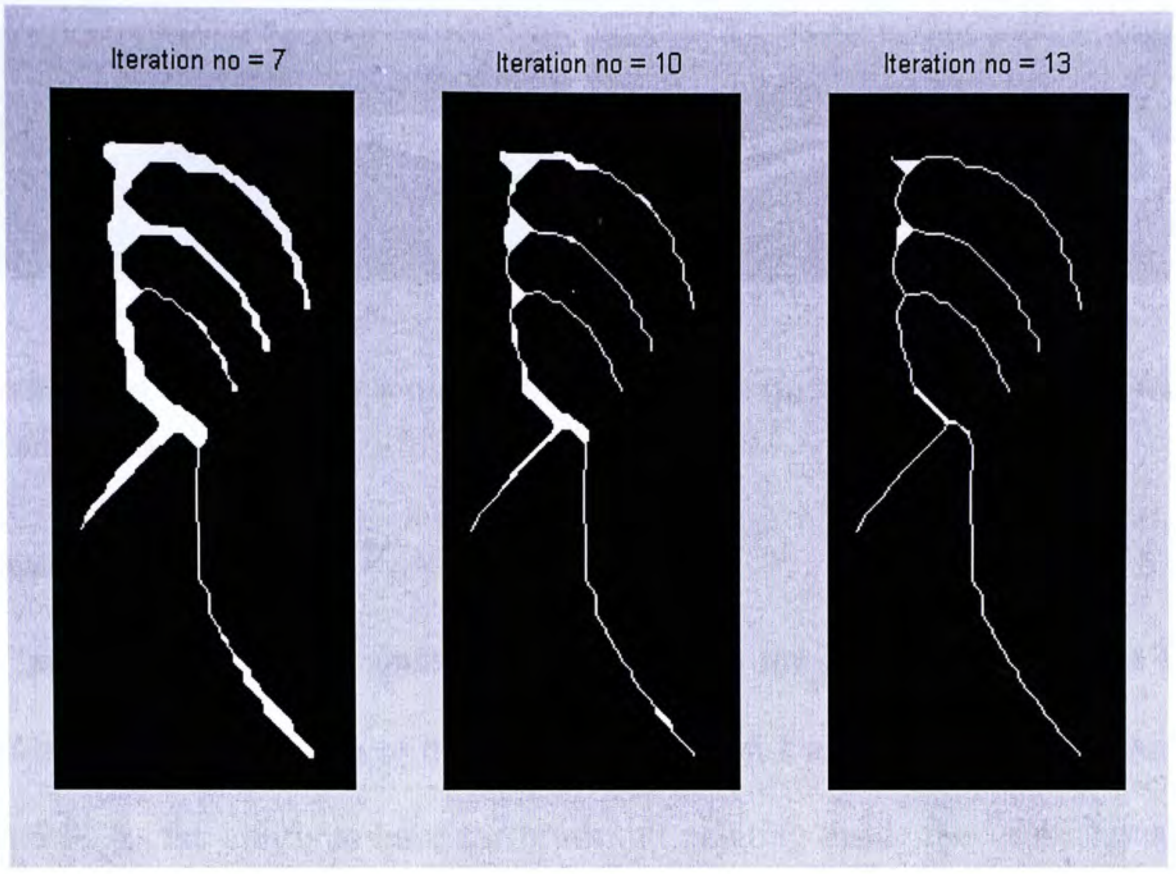
Figure 6.6. Chromosome structure of the painting plan which represented by Bezier curve control point genes and painting pressure genes, where m is the total control point group number and n is the total number of pressure applied.

(ii) Objective and Fitness Evaluation

Similar to section 4.3, the objective function is defined to measure how the individuals performed in stroke generation. Under our scheme, the stroke generated should clearly resemble i.e., fully painted with in bound the sample stroke given, the sample stroke in our study is shown in Figure 6.7. It is in Running Script style "行書", which is one of the major categories of Chinese calligraphy.

Figure 6.7 The sample stroke "一" the Chinese character number "one" in Running Script style "行書"

The principle of the objective function is thus designed according to the above painting scheme. In the case of a minimization problem, the most fitted individual will have the lowest numerical value of the associated objective function. There are three objectives in our consideration as presented in the following. The fitness function transformation will be a non-linear one similar to (Eqn 4.5).

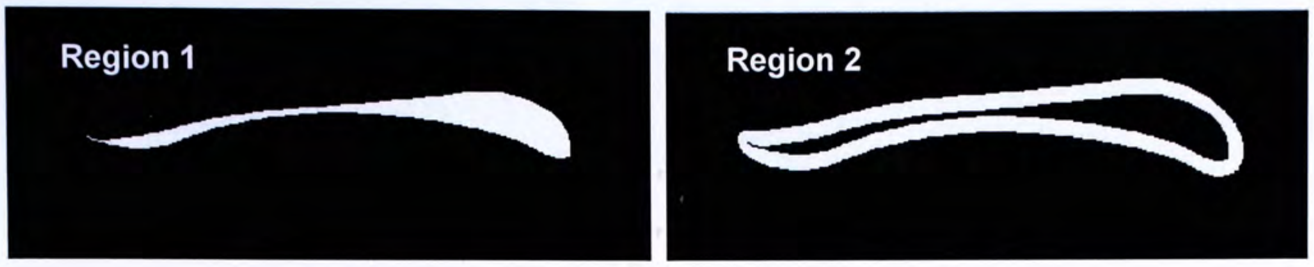


(a) (b) (c)

Figure 6.8. The image of Chinese character "見" expressed in Seal Character "篆書" undergoing the iterative thinning process, (a) The remaining pixels after 7 iterations. (b) after 10 iterations, (c) after 13 iterations.

Objective 1: Painting inside the stroke boundary.

Figure 6.8 depicts the thinning algorithm which is the procedure to iteratively remove of region boundary pixels. Removal is repeated until a pixel set with maximum thickness of one or two is obtained [21]. Thus, the more the iterative removal, the thinner the image skeleton. By using such thinning property, we can divide the sample stroke into two regions automatically. This is shown in Figure 6.9(a), where the sample stroke is thinned iteratively until the remaining region is reduced to 35% of its original.



(a) (b)

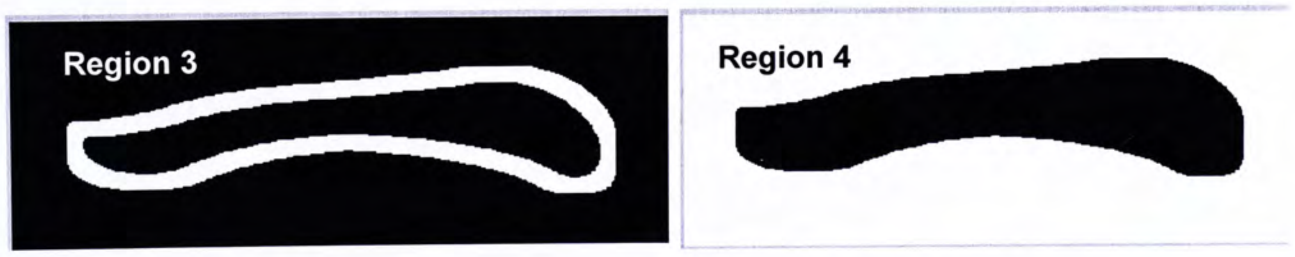
Figure 6.9. The sample stroke is divided into two regions: (a) inner region, (b) outer region, those are inside the stroke

Minimize objective 1: $ObjV_1 = w_1 p_1 + w_2 p_2$ (Eqn 5.1),

where p_i is the number of unpainted pixels within the region i and w_i is the corresponding cost weighting of the region. The region 1 and region 2 are shown in Figure 6.9. As the aim is to have the brush tuft painting inside the stroke, usually we would set $w_1 > w_2$ during the GA evolution.

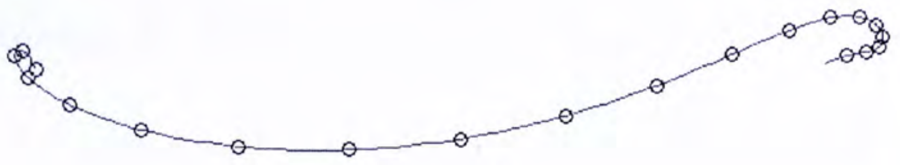
Objective 2: Not painting out of stroke boundary

Contrast to the thinning algorithm, dilation is a morphological operation to expand image object. The iterative procedure effectively sticks a pixel "layer" on the image object. Figure 6.10 shows the sample stroke being dilated iteratively until the added "layer" constitutes 85% area of the sample stroke.

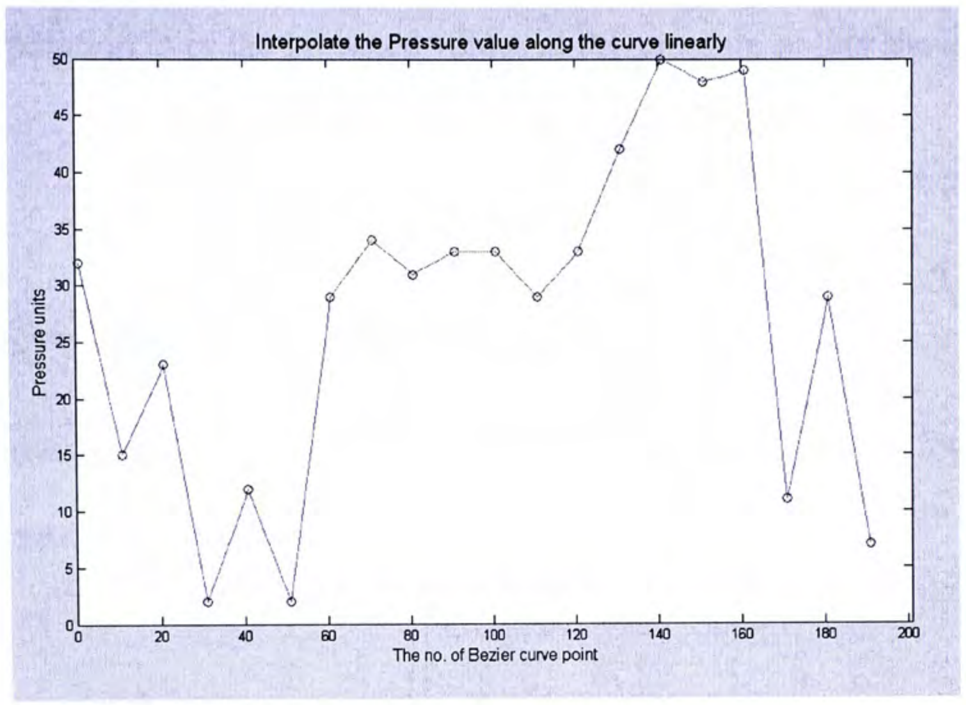


(a) (b)

Figure 6.10. There are two regions colored outside the sample stroke: (a) skin region, (b) out-of-skin region.



(a)



(b)

Figure 6.11. (a) The circles indicate the pressure points distributed along the trajectory, (b) Linear Interpolation of presence in before points.

Minimize objective 2: $ObjV_2 = w_3 p_3' + w_4 p_4'$ (Eqn 5.2),

where p_i is the number of painted pixel within the white region i as shown in Figure 6.10. Similarly, we set $w_4 > w_3$ to enforce the desire of not painting far off from the stroke boundary.

Objective 3: Painting with smooth pressure change.

In real calligraphy painting, the force applied on the brush pen usually increases or decrease gently for smooth painting. It is thus necessary to design a penalty scheme for handling the pressure change along the trajectory.

Minimize Objective 3: $ObjV_3 = w_5 \sum_{i=1}^{\max-1} \left| pres_{i+1} - pres_i \right|$ (Eqn 5.3),

where $pres_i$ is the pressure levels applied on the i^{th} pressure point as depicted in Figure 6.11(b). Note that the actual applied pressure over the Bezier curve is the linear interpolation over the presence values at the pressure points shown in Figure 6.11(a).

6.4 Evolutionary computing results

The following genetic parameters and operations are adopted for the stroke generation problem at hand.

1) No. of control point groups defined: m, 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=3 pixels, see the example of Figure 6.12.
2) No. of control points in group i: n_cp_i
3) Total no. of control points inside the stroke: cpn
4) No. of pressure level applied: There are n=20 pressure levels ranging from 1 to 50, i.e., pressure value are defined [$pres_{\min}$, $pres_{\max}$]
5) Stroke region division: the area of region 1, 2 and 3 as relation to that of the sample stroke in percentage $R_{i=1,\dots,3}$=[35, 65, 85]
6) Objective value weighting: [$w_1, w_2, \dots, w_5$]
7) Chromosome length in binary coded:

$$l_{bit} = \sum_{i=1}^{m} \lceil \log_2(n\_cp_i) \rceil + n\lceil \log_2(pres_{\max} - pres_{\min} + 1) \rceil \quad \text{(Eqn 5.4)}$$

8) Crossover method: Multi-point crossover with probability of crossover p_c

9) Mutation method: New individuals are generated by taking the current population and mutating each element with mutation probability $p_m = 0.7/l_{bit}$.

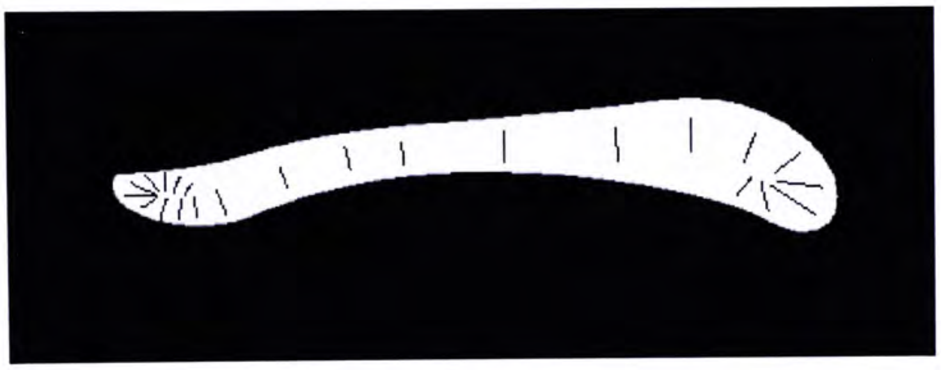
10) Selection method: Stochastic Universal Sampling with population selection p_s.

11) Replacement: Fitness-base reinsertion to current population.

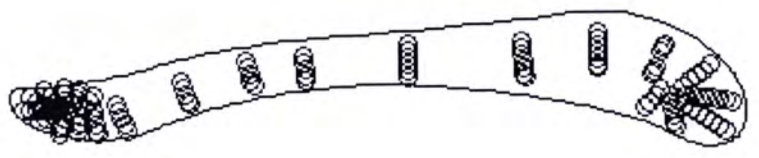
12) Population size N_{ind}: between 40-100, depending on the length of the chromosome.

13) Number of generation past: N_{gen}

14) Initial population: randomly generated within the groups of control point and range of pressure value.



(a)



(b)

Figure 6.12. (a) There are m =23 line segments drawn manually inside the sample stroke, (b) The bubbles circles indicates the control points placed on the line segment.

The GA stroke generation result is performed on a notebook PC with Centrino (1.6G) CPU, 512 MB RAM, using Windows XP. The computation time for generating the stroke solution is roughly 20-50mins. This computation time (averaged 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.

In result 1, a set of circular spray paint templates is adopted and depicted in Figure 6.13(a). The objective cost diagram during the GA evolution is shown in Figure 6.13(b), and Figure 6.14 shows the best resulting strokes for each 10 generation past during the evolution. In result 2, the water drops templates are used as depicted in Figure 6.15(a). The objective cost diagram and the resulting strokes respectively are shown in Figure 6.15(b) and Figure 6.16. By comparing two experimental GA results, water drop template performed better and the final best outcome indicated with a painting trajectory is depicted as Figure 6.17.

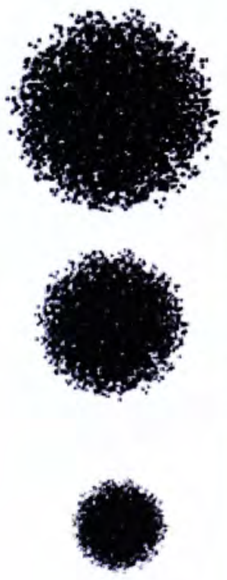
For further demonstration, we would like to replicate a Chinese calligraphy character by a very famous ancient calligrapher, Wang Xianzhi (王羲之). One character, see Figure 6.18(a), is chosen from his well known masterpiece called Lan ting xu (蘭亭序). Figure 6.18(b) shows the stroke elements extracted manually using image editor software and the corresponding stroke control points are set as depicted in Figure 6.18(c). In this time, another brush template shown in Figure 6.19(a) is adopted which is more or less combined the two previous templates used such that there is spraying effect on its boundary. The evolutionary computing results are demonstrated from result 3 to 6 with the input parameters. The evolutionary objective cost diagrams are shown in Figure 6.19(b), Figure 6.21, Figure 6.23 and Figure 6.25 respectively. Also, the strokes evolution processes are respectively shown in Figure 6.20, Figure 6.22, Figure 6.24 and Figure 6.26. Finally, the recombination of evolutionary strokes is shown in Figure 6.27(a), and the corresponding painting trajectories are shown in Figure 6.27(b) as well. Similarly, TSP solver (CE-Method) can be performed in this case for fast-effective calligraphy painting.

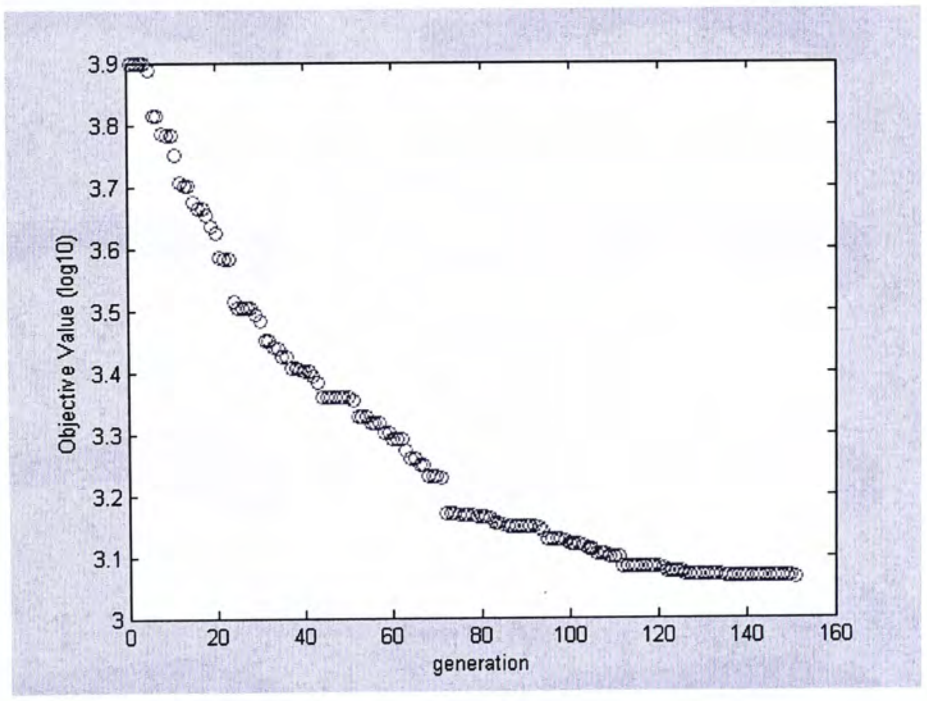
Result 1

The 2D brush template used as Figure 6.13(a).
No. of sub population: N_{ind} **=60**

Max. generations past: N_{gen} **=151**

New individuals created per one reproduction: $N_{ind} \times p_s$ **=60x70%=42**
Chromosome length: **182 (in binary coding)**
Mutation rate: p_m**= 3.8462e-3**
Crossover rate: p_c**=0.7**
Total group no. of control points: m **=23**
Total no. of control points: cpn**=121**
No. of pressure levels: n**=20**
Max. and Min. pressure: $[pres_{\min} \quad pres_{\max}]$**=[1 50]**
Objective value weighting: w_i**= [3 1 1 3 3]**
No. of point interpolated in the Bezier curve: **200**
No. of choice of Bezier curve trajectories: **1.9508e+16**
Time usage: **25mins 43s**





(a) (b)

Figure 6.13. (a) A set of 50 circular spray paint templates, the max and min sizes are 100×100 and 2×2, (b) The diagram shows the objective cost (in log scale) of the best individual during evolution

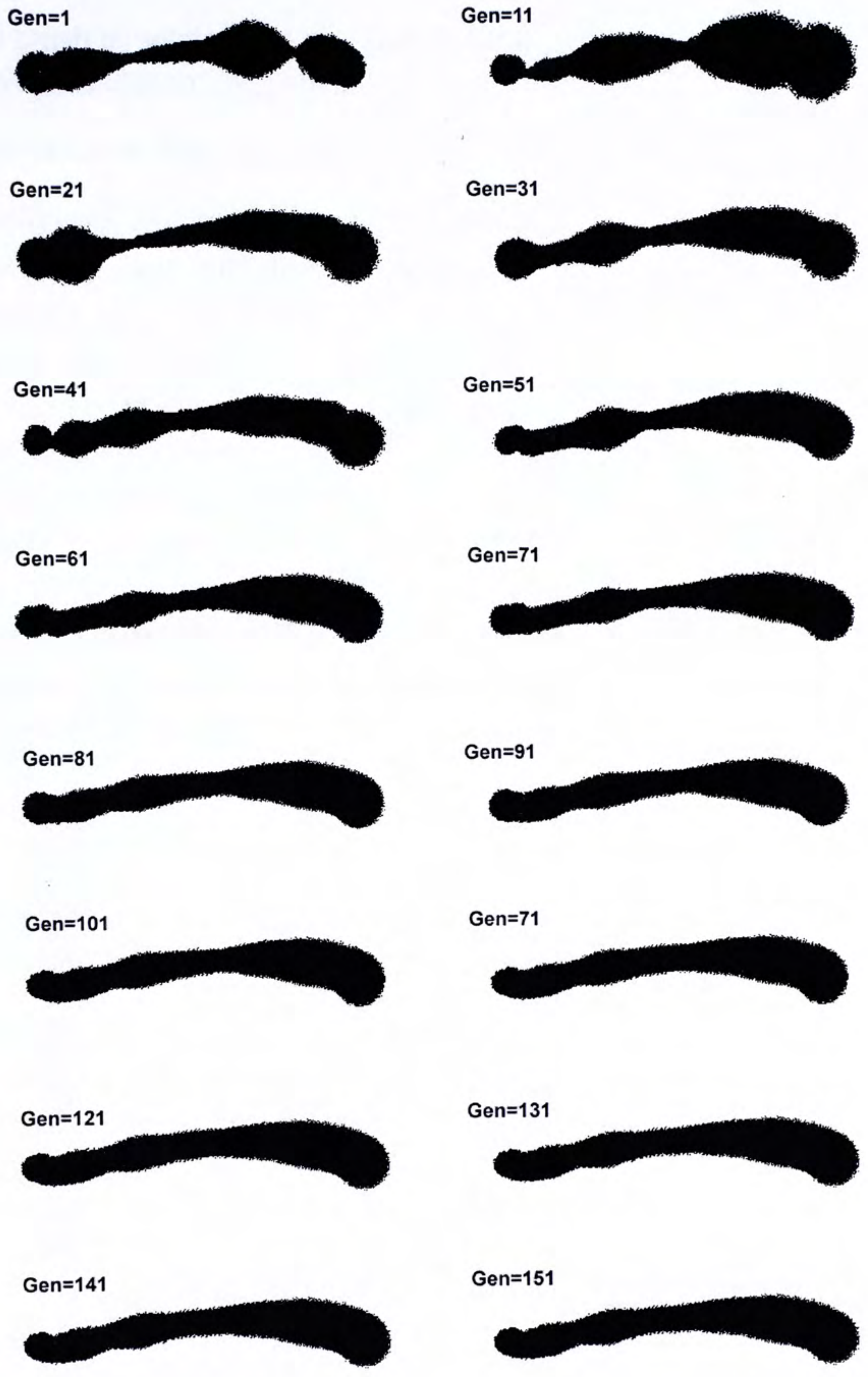


Figure 6.14. A stroke painting plan is being modified during evolution and the best individual is shown for each 10 generation past.

Result 2

The 2D brush template used as Figure 6.15(a).

No. of sub population: N_{ind} **=60**

Max. generations past: N_{gen} **=151**

New individuals created per one reproduction: $N_{ind} \times p_s$ **=60x70%=42**

Chromosome length: **182 (in binary coding)**

Mutation rate: p_m **= 3.8462e-3**

Crossover rate: p_c **=0.7**

Total group no. of control points: m **=23**

Total no. of control points: cpn **=121**

No. of pressure levels: n **=20**

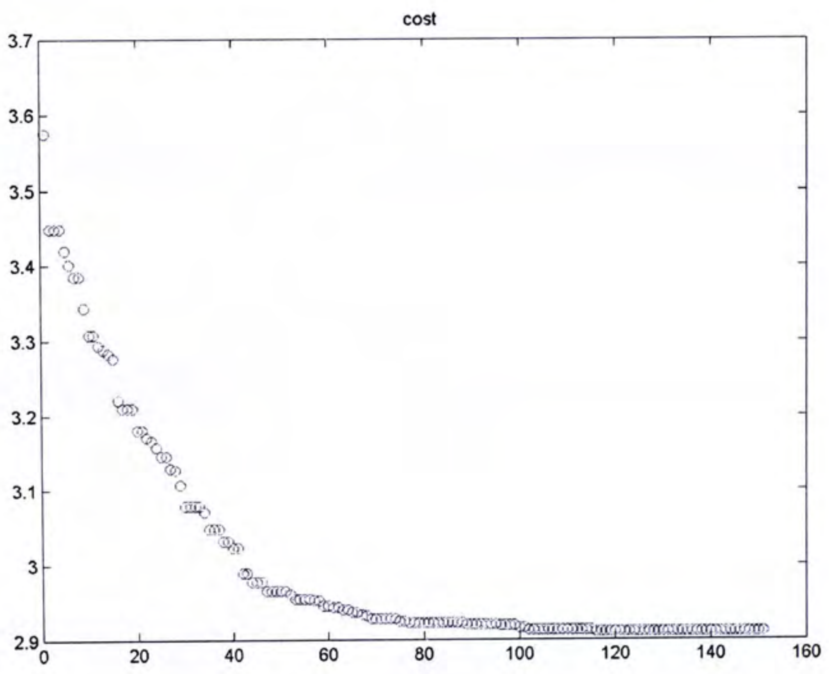
Max. and Min. pressure: $[pres_{\min} \quad pres_{\max}]$**=[1 50]**

Objective value weighting: w_i**= [3 1 1 3 3]**

No. of point interpolated in the Bezier curve: **200**

No. of choice of Bezier curve trajectories: **1.9508e+16**

Time usage: **25mins 10s**



(a) (b)

Figure 6.15. (a) A set of 50 water drop templates, the max and min sizes are 100×50 and 2×1, (b) The diagram shows the objective cost (in log scale) of the best individual during evolution

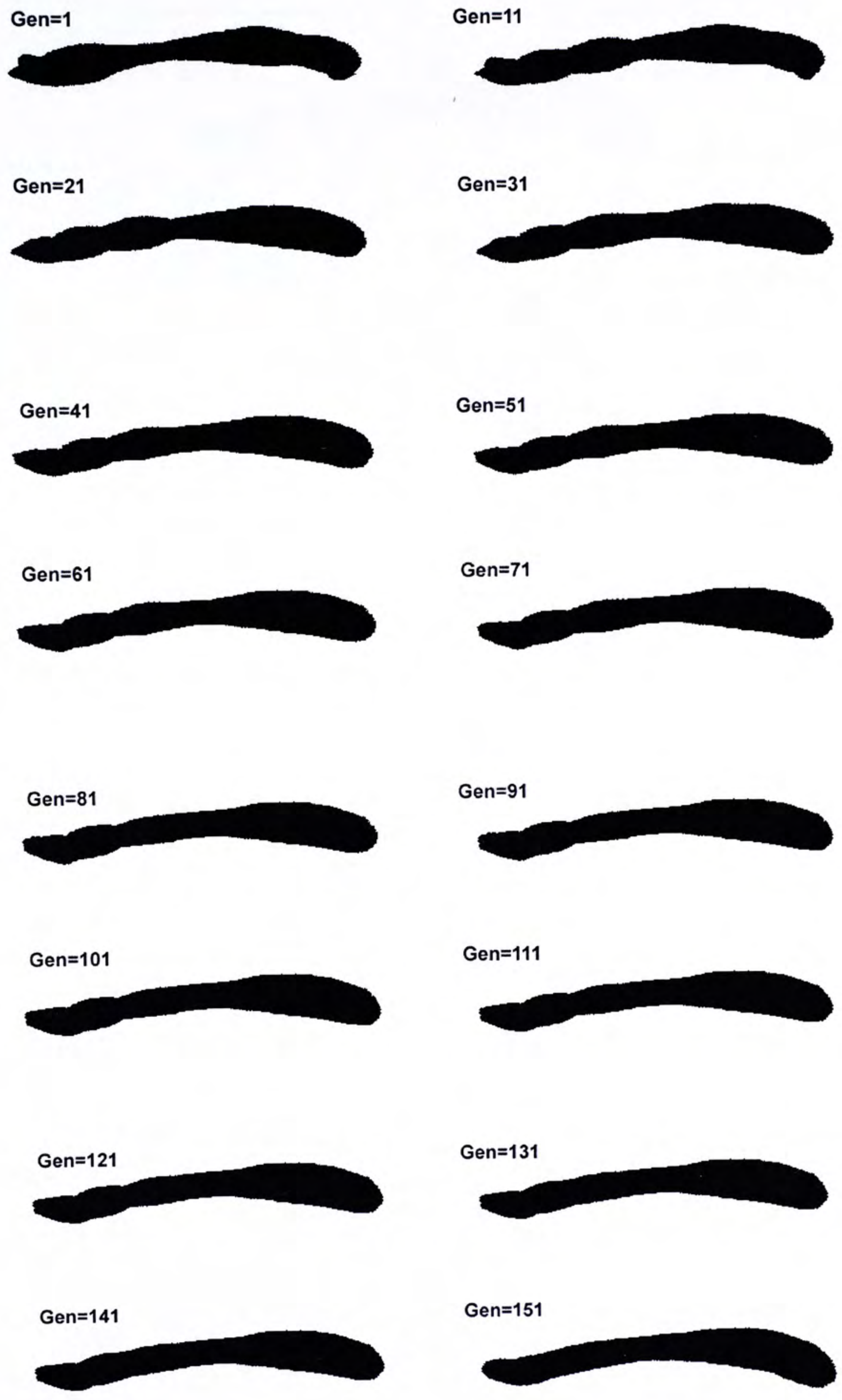


Figure 6.16. Better evolutionary result using water drop template

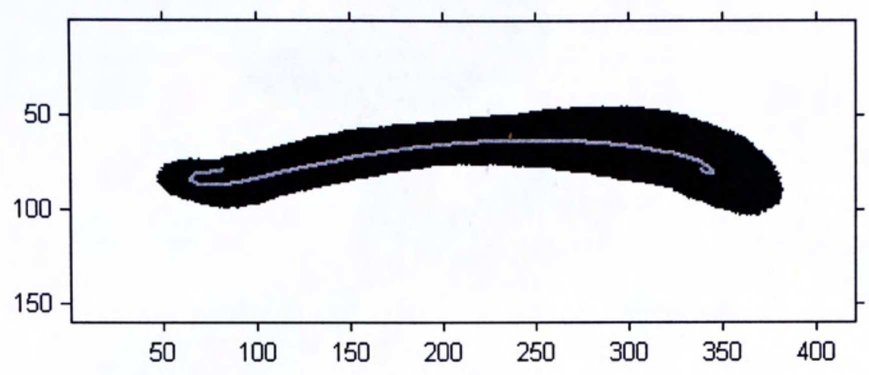
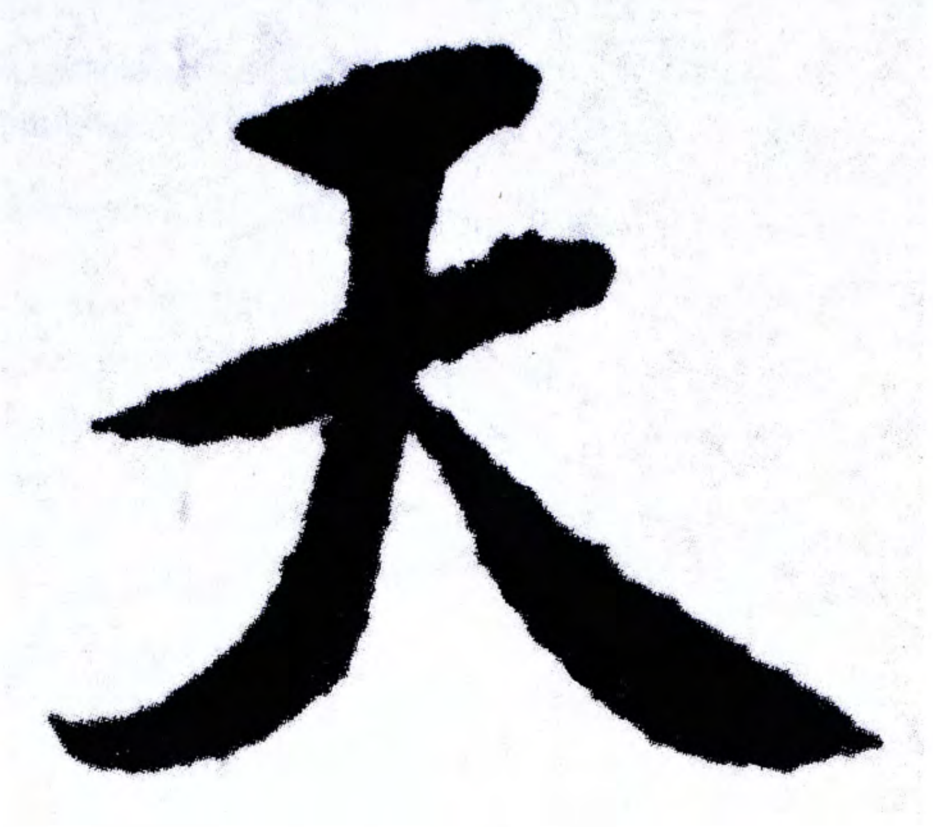
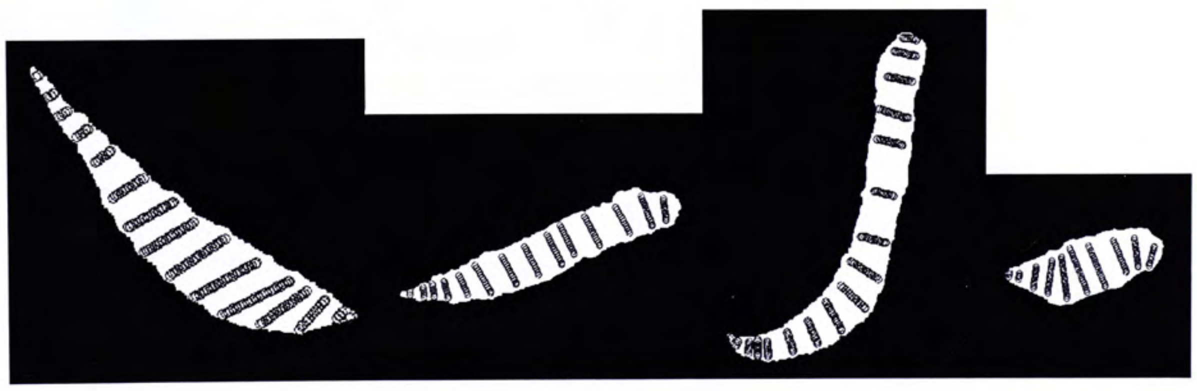


Figure 6.17. The best GA stroke painting scheme using water drop template



(a)

(b)



(c)

Figure 6.18. (a) A Chinese calligraphy character “天” from Lan ting xu “蘭亭序” (source: [22]), (b) Four stroke elements extracted, (c) The control points set inside the corresponding strokes

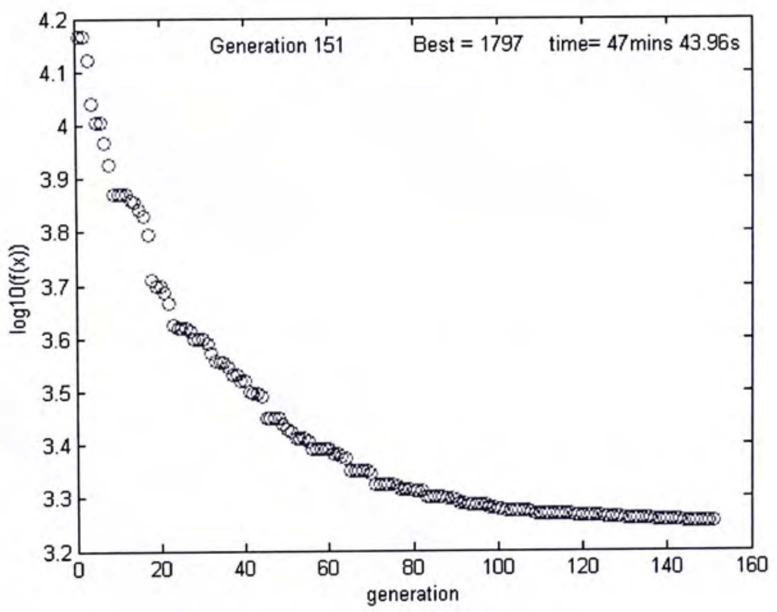
Result 3

The 2D brush template used as Figure 6.19(a).
No. of sub population: N_{ind} **=60**

Max. generations past: N_{gen} **=151**

New individuals created per one reproduction: $N_{ind} \times p_s$ **=60x70%=42**
Chromosome length: **172 (in binary coding)**
Mutation rate: p_m **= 4.0698e-3**
Crossover rate: p_c **=0.7**
Total group no. of control points: m **=17**
Total no. of control points: cpn **=179**
No. of pressure levels: n **=20**
Max. and Min. pressure: $[pres_{\min} \quad pres_{\max}]$**=[1 50]**
Objective value weighting: w_i **= [3 1 1 3 1]**
No. of point interpolated in the Bezier curve: **200**
No. of choice of Bezier curve trajectories: **1.0904e+14**
Time usage: **47mins 44s**

(a)



(b)

Figure 6.19. (a) Water drop templates with spraying effect on the boundary, (b) The diagram shows the objective cost (in log scale) of the best individual during evolution of result 3

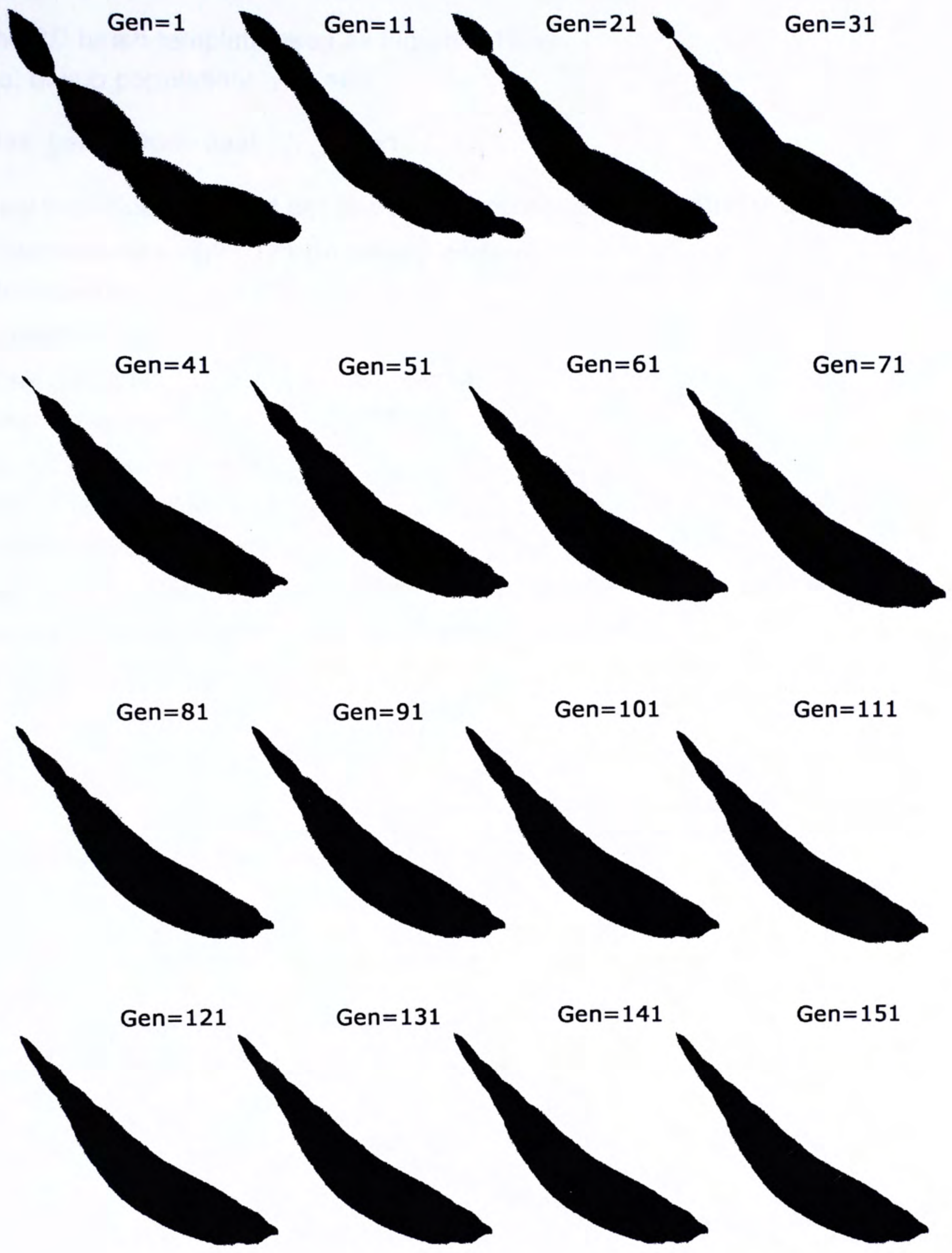


Figure 6.20. Stroke evolutionary process of result 3

Result 4

The 2D brush template used as Figure 6.19(a).

No. of sub population: N_{ind} **=60**

Max. generations past: N_{gen} **=151**

New individuals created per one reproduction: $N_{ind} \times p_s$ **=60x70%=42**

Chromosome length: **171 (in binary coding)**

Mutation rate: p_m**= 4.0936e-3**

Crossover rate: p_c**=0.7**

Total group no. of control points: m **=14**

Total no. of control points: cpn **=150**

No. of pressure levels: n **=20**

Max. and Min. pressure: $[pres_{\min} \quad pres_{\max}]$**=[1 50]**

Objective value weighting: w_i**= [3 1 1 3 1]**

No. of point interpolated in the Bezier curve: **200**

No. of choice of Bezier curve trajectories: **8.4224e+13**

Time usage: **37mins 54s**

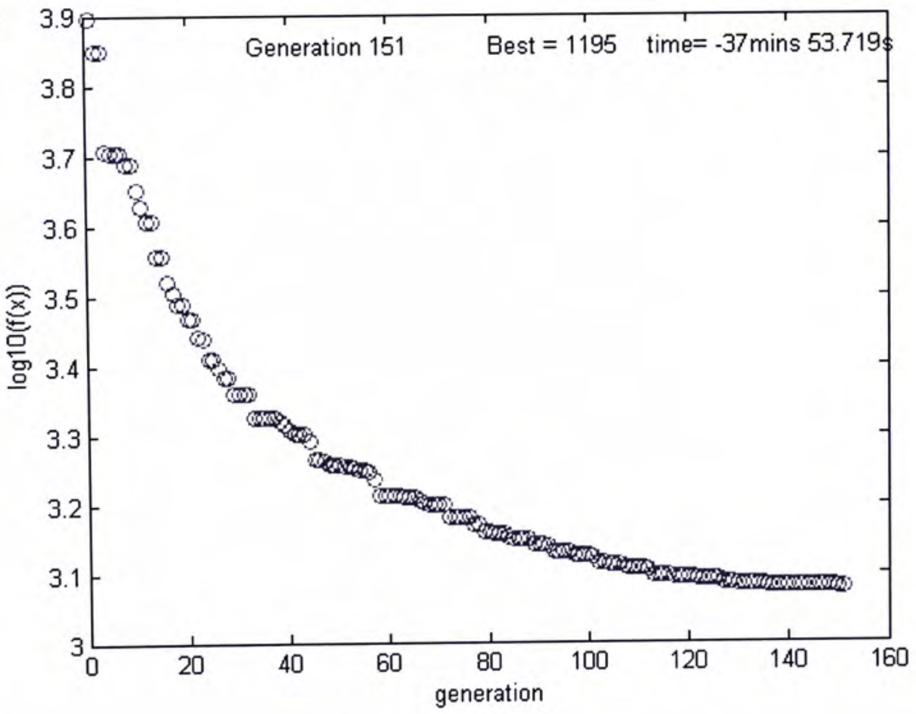


Figure 6.21. The diagram shows the objective cost (in log scale) of result 4

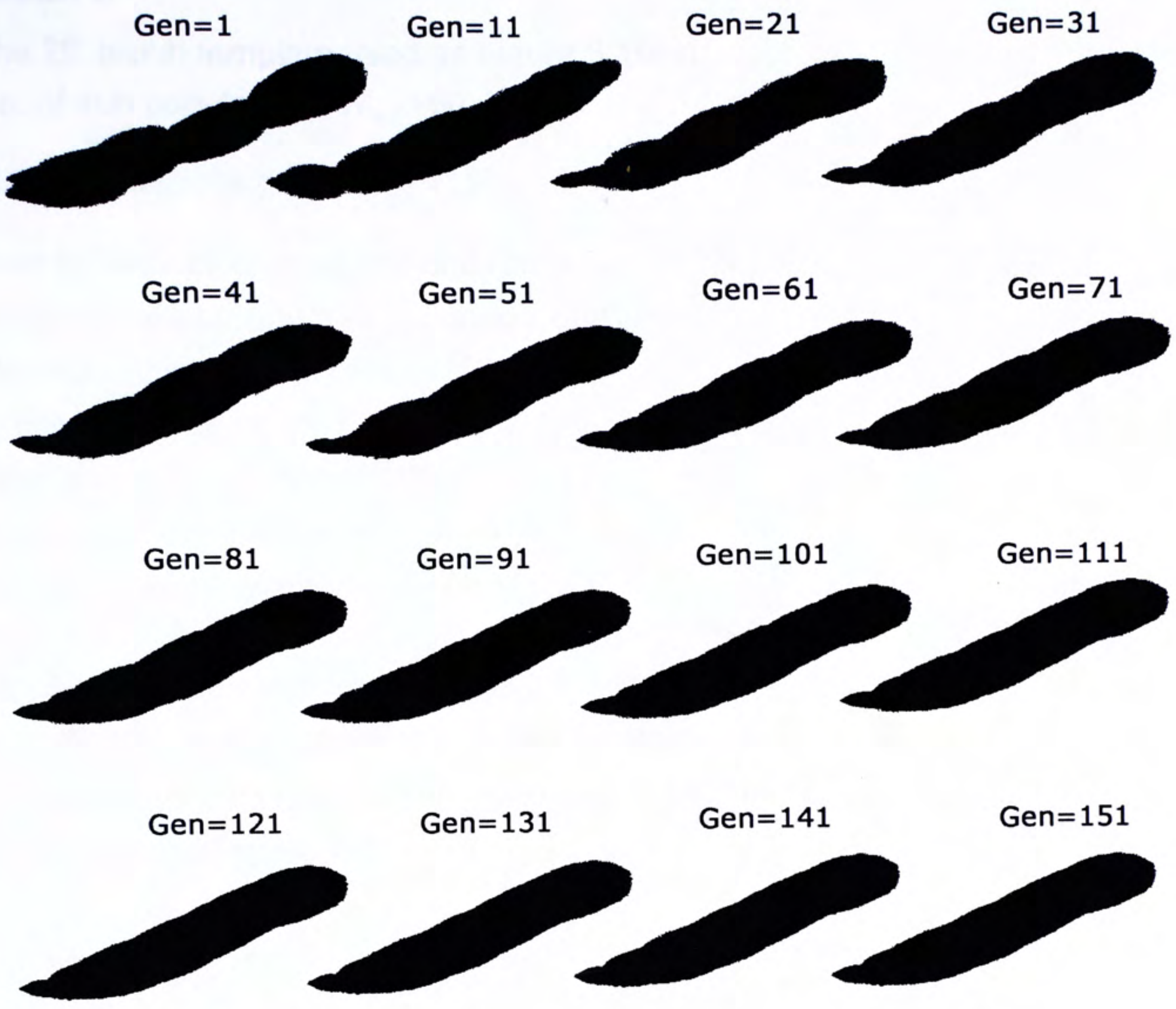


Figure 6.22. Stroke evolutionary process of result 4

Result 5

The 2D brush template used as Figure 6.19(a).
No. of sub population: N_{ind} **=60**

Max. generations past: N_{gen} **=151**

New individuals created per one reproduction: $N_{ind} \times p_s$**=60x70%=42**
Chromosome length: **224 (in binary coding)**
Mutation rate: p_m= **3.125e-3**
Crossover rate: p_c**=0.7**
Total group no. of control points: m **=17**
Total no. of control points: cpn**=179**
No. of pressure levels: n**=27**
Max. and Min. pressure: $[pres_{\min} \quad pres_{\max}]$**=[1 50]**
Objective value weighting: w_i= **[3 1 1 3 1]**
No. of point interpolated in the Bezier curve: **200**
No. of choice of Bezier curve trajectories: **7.1824e+16**
Time usage: **28mins 29s**

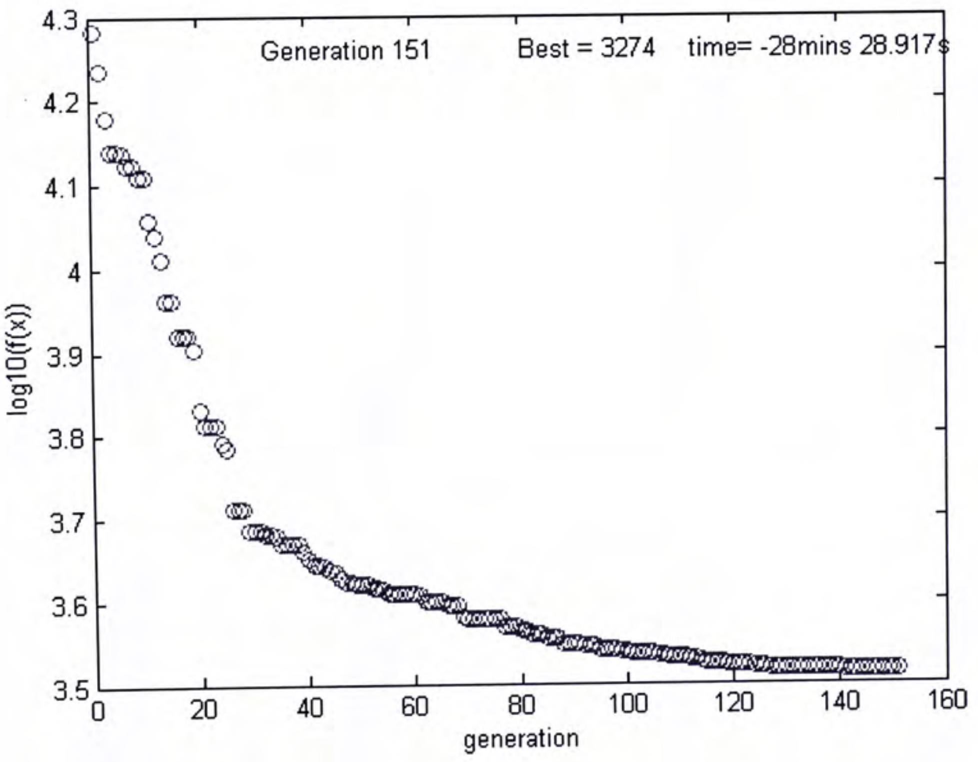


Figure 6.23. The diagram shows the objective cost (in log scale) of result 5

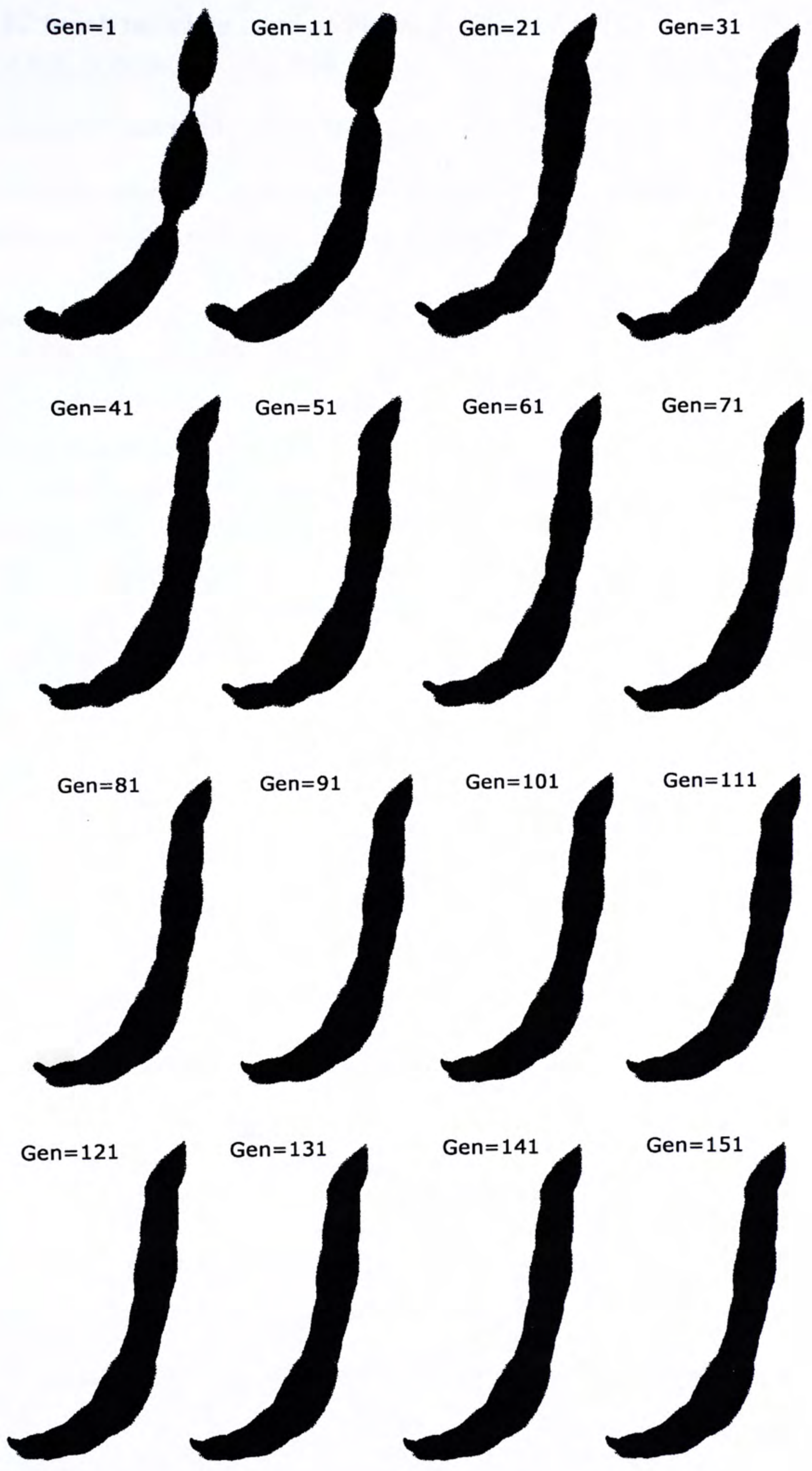


Figure 6.24. Stroke evolutionary process of result 5

Result 6

The 2D brush template used as Figure 6.19(a).

No. of sub population: N_{ind} **=60**

Max. generations past: N_{gen} **=151**

New individuals created per one reproduction: $N_{ind} \times p_s$ **=60x70%=42**

Chromosome length: **224 (in binary coding)**

Mutation rate: p_m**= 3.8043e-3**

Crossover rate: p_c**=0.7**

Total group no. of control points: m **=10**

Total no. of control points: cpn **=132**

No. of pressure levels: n **=24**

Max. and Min. pressure: $[pres_{\min} \quad pres_{\max}]$**=[1 50]**

Objective value weighting: w_i**= [3 1 1 3 1]**

No. of point interpolated in the Bezier curve: **200**

No. of choice of Bezier curve trajectories: **5.0035e+10**

Time usage: **36mins 53s**

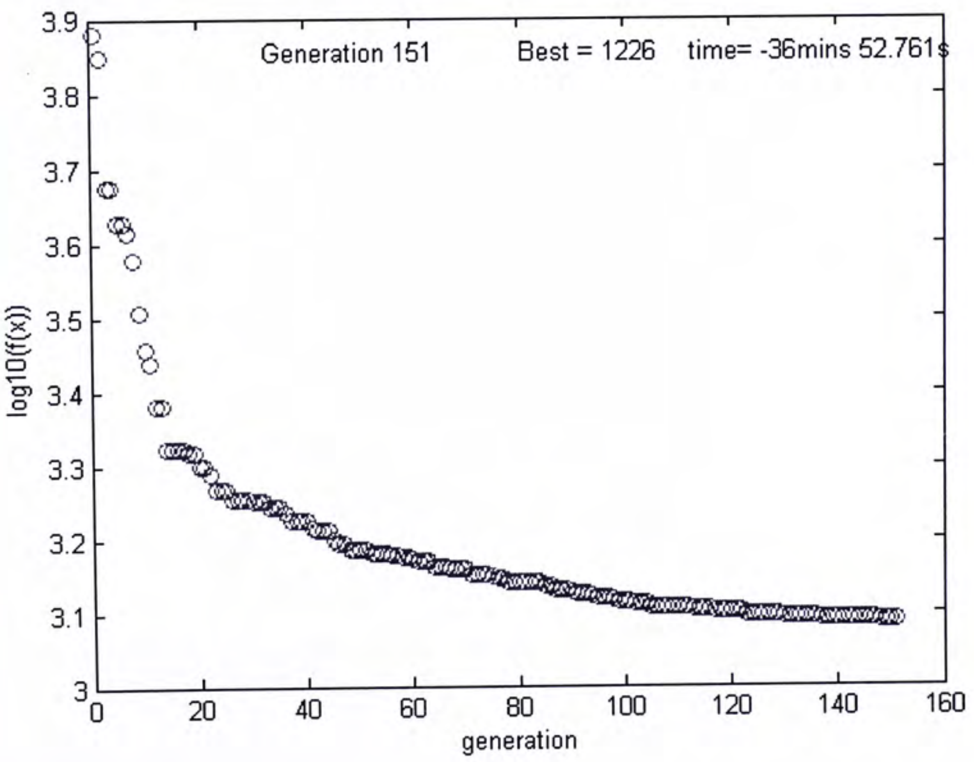


Figure 6.25. The diagram shows the objective cost (in log scale) of result 6

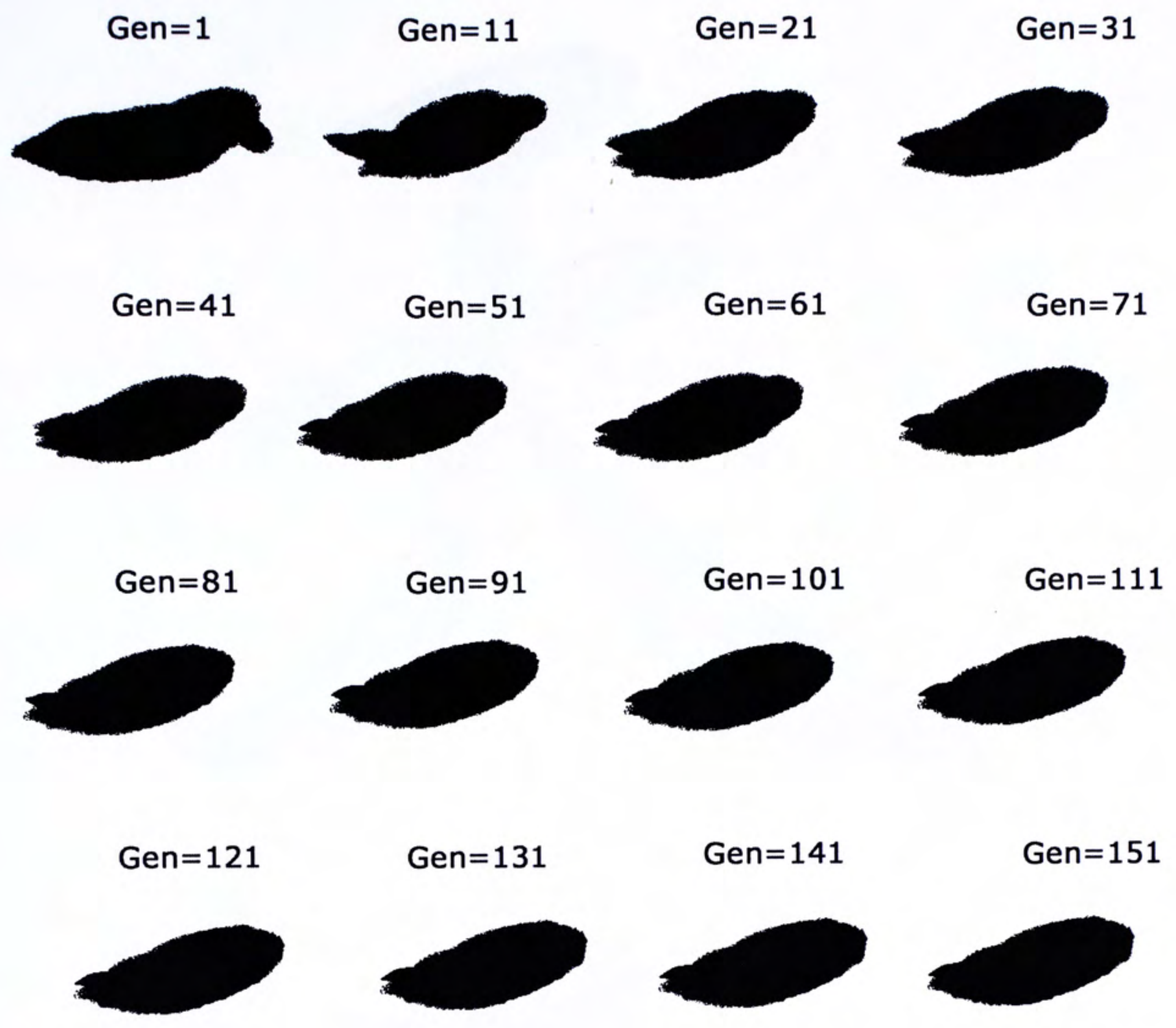
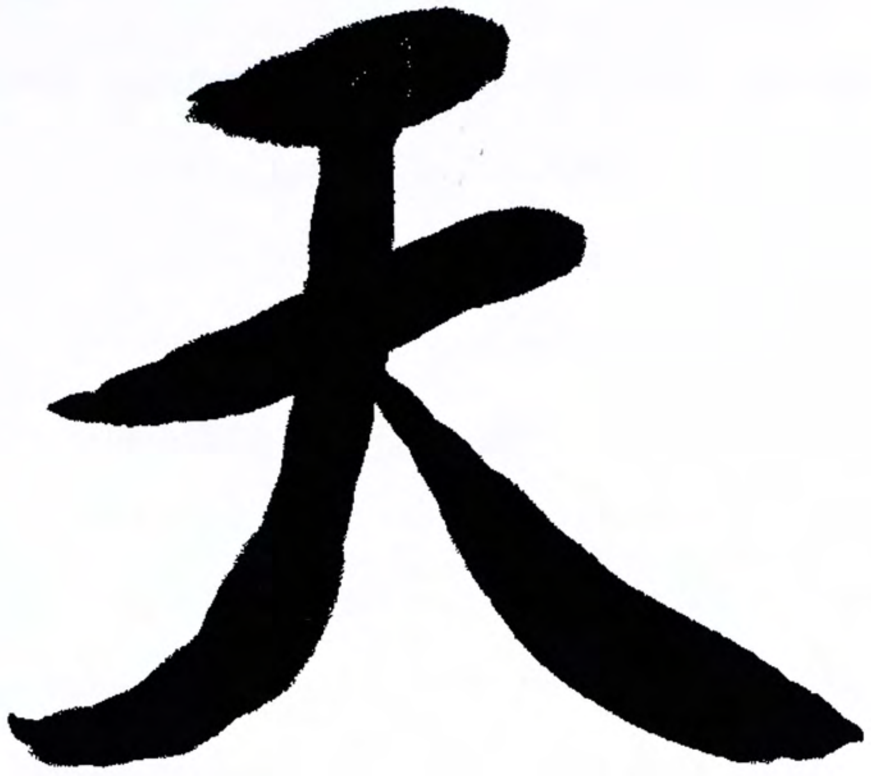
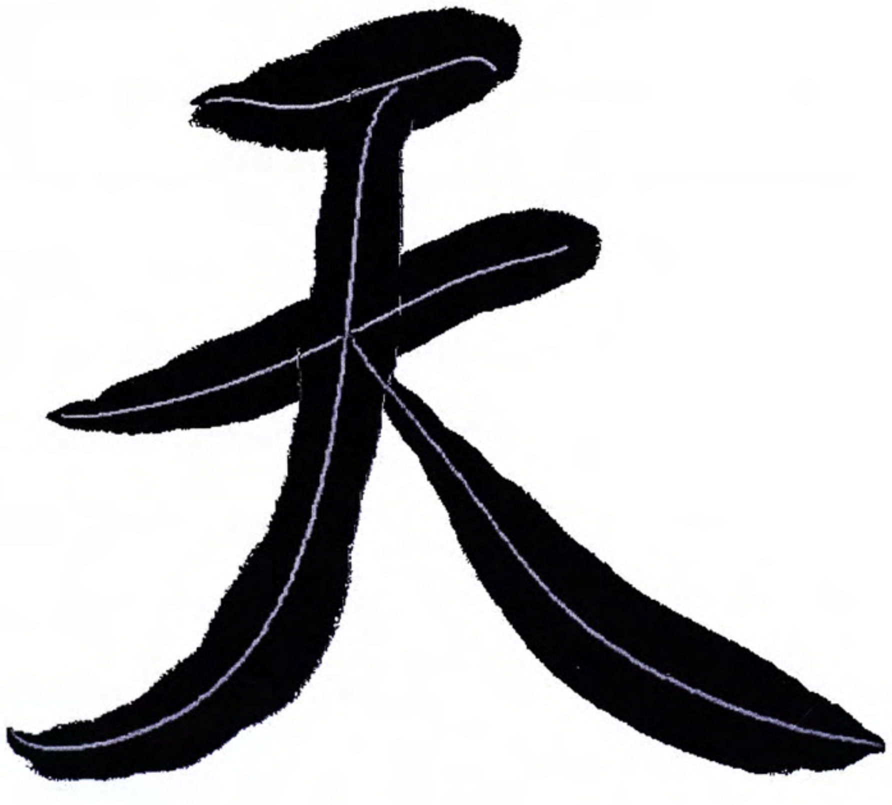


Figure 6.26. Stroke evolutionary process of result 6



(a)



(b)

Figure 6.27. (a) The calligraphy character replication formed by the four evolutionary strokes in result 3-6, (b) Indicated with the painting trajectories

6.5 Chapter summary

Good painting trajectory and stroke thickness control are essential for high artistic quality in Chinese calligraphy. In this chapter, a novel method of stroke painting plan generation using GA is presented. We decretize the brush model as 2D brush template of different sizes and represent the painting trajectory by the control points of Bezier curve. Depending on the pressure applied, templates of different size would be applied along the brush trajectory to trace out the stroke. Using GA to determine a brush stroke solution enhances the flexibility and freedom for obtaining desirable match with the given sample, rather than having to analyze the geometric information. The results are promising which show that GA is sufficient to generate a full stroke by using the appropriate brush template, and the calligraphy can be replicated successfully in simulation. The performance of GA so far is quite dependent on some of the input parameters. Sometimes, its computational results and time may not be as good as we would expect. It is envisioned that heristic Genetic Algorithms (HGA) will be proposed to enhance the computational performance by the new heuristic crossover and mutation operators which based upon the domain-specific characteristics of the sample stroke and brush model.

Hopefully, we can use this kind of technique to replicate the masterpieces by some famous passed away calligraphers e.g. Wang Xianzhi (王羲之), all their writing style will be collected and recovered in the computational sense. Such evolutionary painting idea will be entended to execution in our robot drawing platform that will be dicussed in the next chapter.

7. BRUSH STROKE FOOTPRINT CHARACTERIZATION

Unlike western painting style, the spiritual depiction in Chinese calligraphy is to large extent expressed by the brush stroke rather than the outward appearance of the painted subjects. It is thus worthy to study the Chinese brush model more deeply.

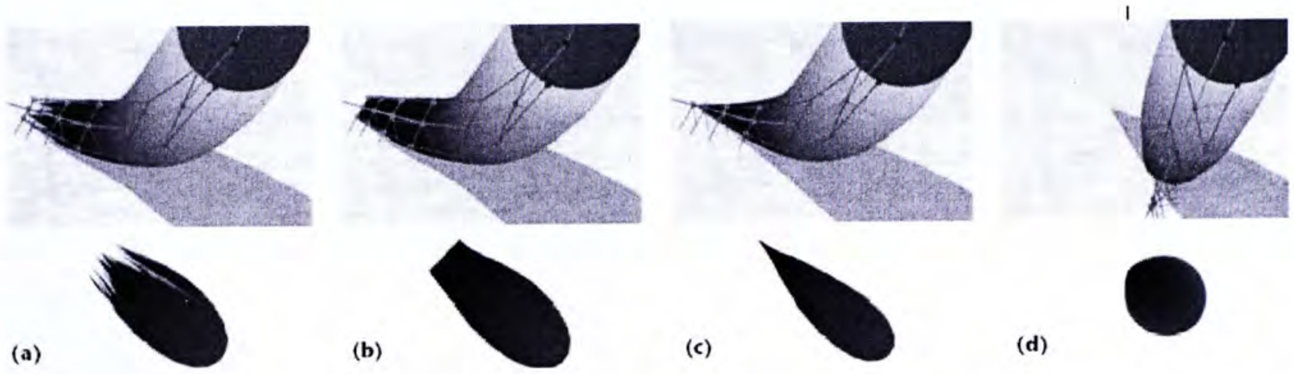


Figure 7.1. (a) A fully functioning tuft pressed against virtual paper, (b) Depicted result with the texture-based bristle-splitting effect turned off, (c). Recent with lateral spreading furthered removed, (d) Model do not handle collision with paper (Source: [42])

Specially, Xu. et la. [38] and Lee [39] worked on the modeling aspects of the computer generated artistic Chinese calligraphy in simulation, and Wong et la. [40] on a virtual brush stroke model. In the recent works of Nelson S.-H. Chu et. la. [41]-[42], a 3D virtual brush model is developed to produce stroke rendering during brush painting. The instantaneous tuft footprints under different painting condition are shown in Figure 7.1. The work so far, however, did not deal with the actual production of Chinese artistic stroke. In what follows, we describe an experiment to determine experimentally a realistic brush model. The experimental setup for this investigation is described in chapter 2. Basically, it consists of a video camera, 5 degree of freedoms robot manipulation, a brush pen with high elasticity tuft and a flat transparent container filled with a blue liquid instead of drawing on the paper. It involves the real-time capturing and data analysis of the brush footprint using the newly developed capabilities in the platform. This chapter provides the future exploration in visual-based decision for full stroke painting and perspective on the brush stroke characterization based on some footprint image properties. Hopefully, the brush model can be established first through an object-oriented analysis such that the variation of footprint profile, a 5-dimensional brush motion, and also the ink-depositing process in real paper painting can be modeled in a hierarchical structure with well-defined object-oriented operations and attributes. Besides, the future development of GA-based full stroke painting system is also introduced.

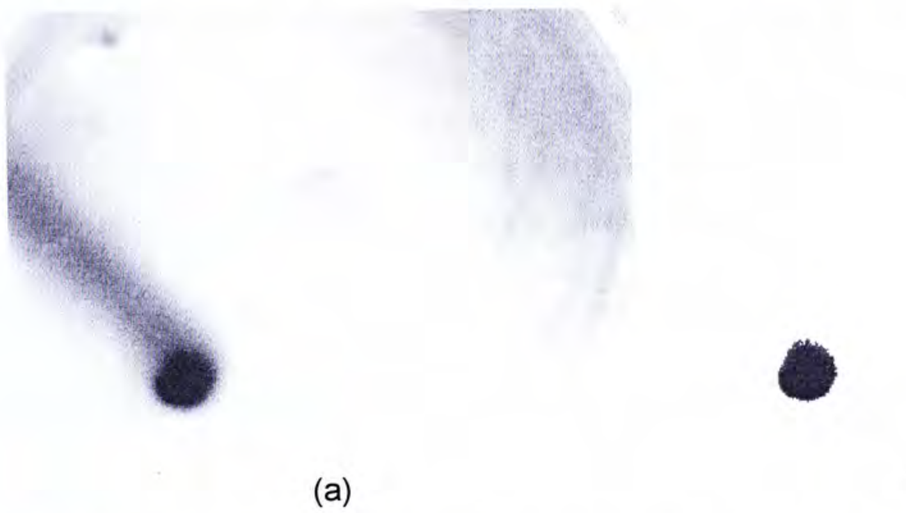


Figure 7.2. (a) The video captured frame shows the instantaneous footprint profile during painting, (b) Only the footprint region remained after segmentation performed.

7.1 Footprint video capturing

By using the video camera shown in Figure 2.8(a), it can capture 20-30 frames per second. The instantaneous footprints in one stroke are captured in video file during painting. For our purpose, those video frames will be indexed with the corresponding instant time and brush motion commands. After the process of footprint characterization, the footprint profile will vary as a function of robot manipulation. Figure 7.2(a) shows a video frame captured during stroke painting, and then a simple image processing such as footprint region segmentation is performed; see Figure 7.2(b). The appropriate setting of color, contrast and brightness are adjusted to ease this segmentation process.

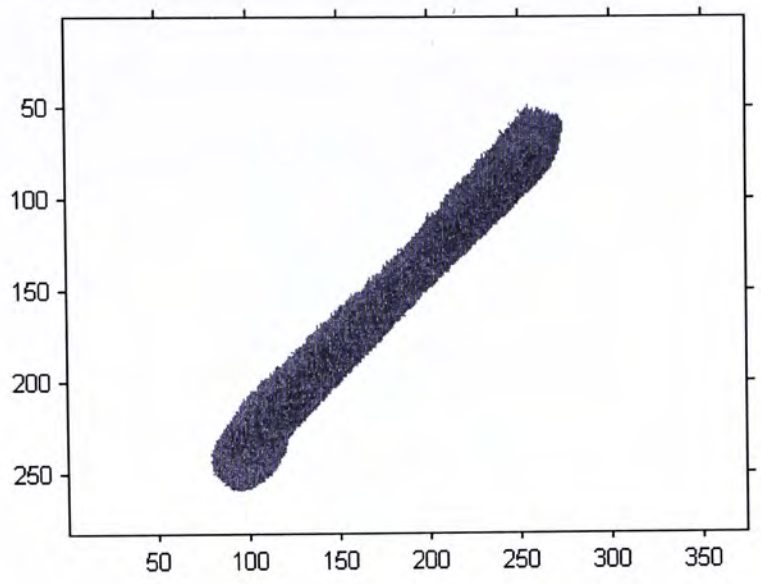


Figure 7.3. The full stroke form achieved by taking the union of all instantaneous footprint images during painting

Even though the brush tuft does not contain any ink and the captured image is an instantaneous form of the footprint, we can take the union of these footprints along the video frames to acquire the would-be full "painting" stroke, as shown in Figure 7.3.

7.2 Footprint image property

By varying only the x-, y- and z-translation of the brush pen motion, a wide variety of brush footprints are produced. The motion of the changing footprint can be captured by a video camera. Our challenge here is to characterize the instantaneous footprint image and try to derive its relations with the kinematics control of the robot. As a start, the following properties are studied as representatives of the resulting footprint.

A. Footprint area

The first property is the area, which constitutes the actual number of pixels in the footprint region. The area measured depends on the brush surface exerting on the paper. When the artist exerts the brush pen more toward the paper, greater tuft bending will enlarge the area value. Figure 7.4 shows the corresponding area produced in the stroke depicted in Figure 7.3 as a function of the video frame number.

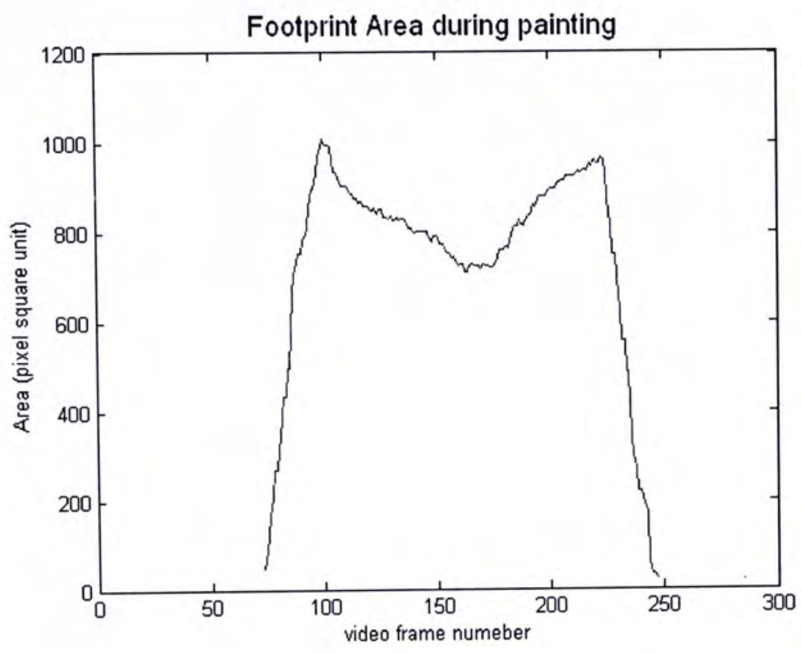


Figure 7.4. The footprint region area is changing due to the painting manipulation

B. Footprint centroid

The centroid of the image region provides information on the separation between the center of the footprint and the commanded (x, y) coordinate at any time instant.

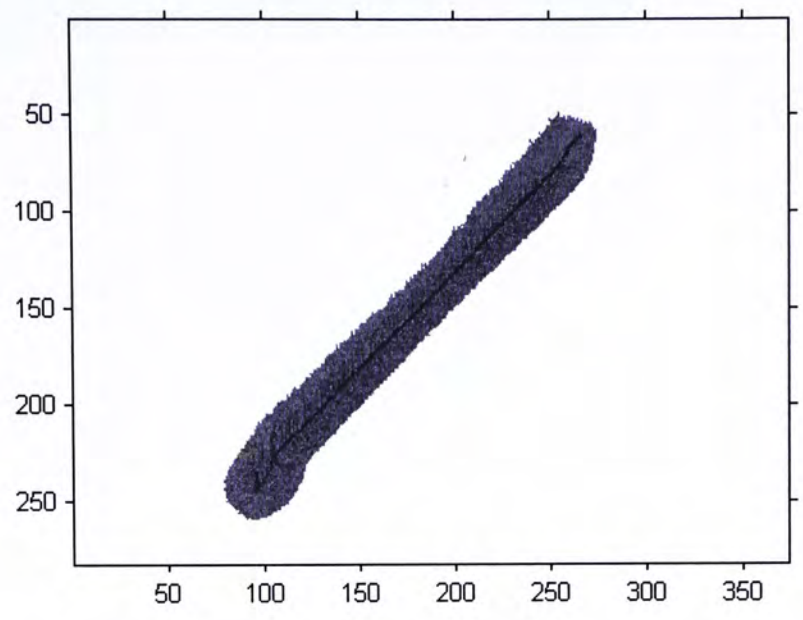
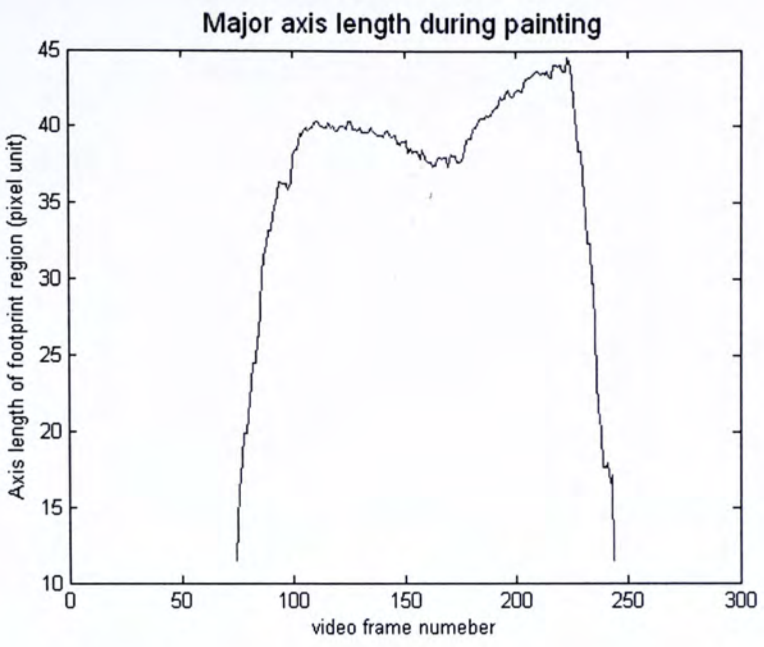


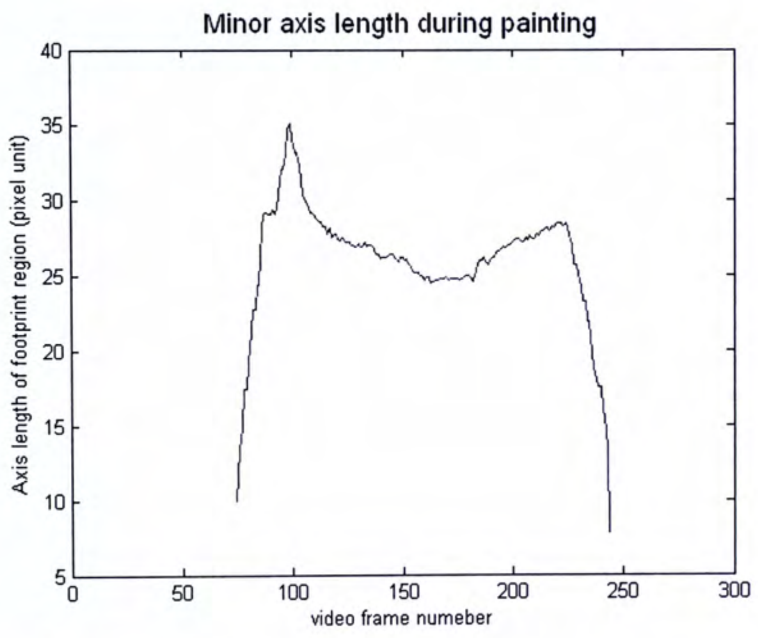
Figure 7.5. The centroids of all instantaneous footprint regions are plotted inside the stroke.

C. Major and minor length

Major and minor axes are the length (in pixels) of the major axis and minor axis of the approximate ellipse that has the same second-moments as the footprint region. Upon determination of the major length a and minor length b, the eccentricity is defined as $e = \sqrt{1 - \frac{b^2}{a^2}}$ which the property can be investigate as a function of velocity and acceleration during painting. Figure 7.6.(a) and Figure 7.6.(b) show the major and minor length respectively during painting the stroke depicted in Figure 7.3.



(a)



(b)

Figure 7.6. The minor axis length (a) and major axis length (b) are changing during the painting manipulation.

D. Orientation

Orientation is the angle (in degrees from -90 to 90) between the x-axis and the major axis of the elliptical approximation of the footprint. By computing the changing rate of orientation angle, the brush tuft plasticity can be reflected. Figure 7.7 shows the diagram of footprint orientation angle that is changing during the painting. We can observe the orientation of the approximation ellipse is relatively steady during intermediary brush running.

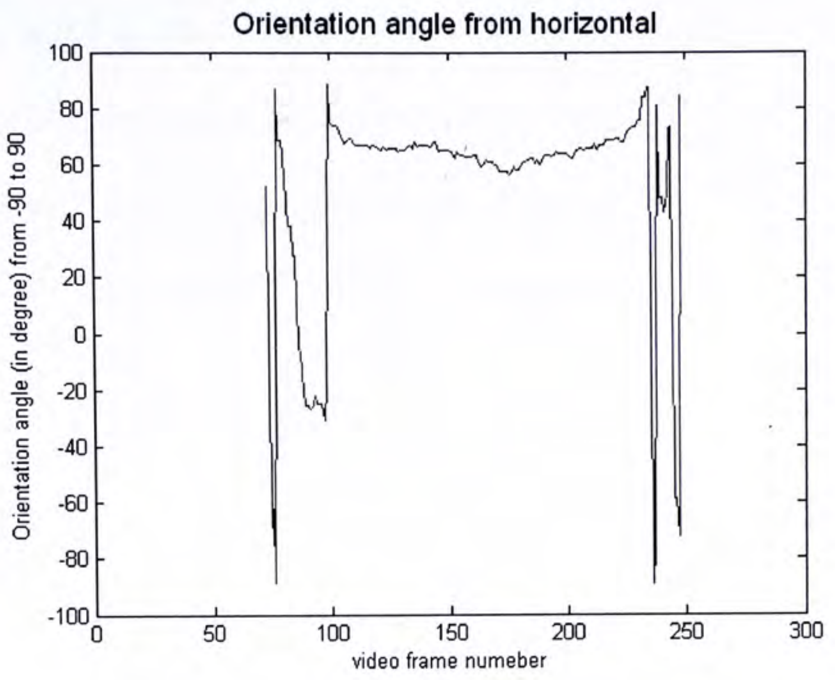


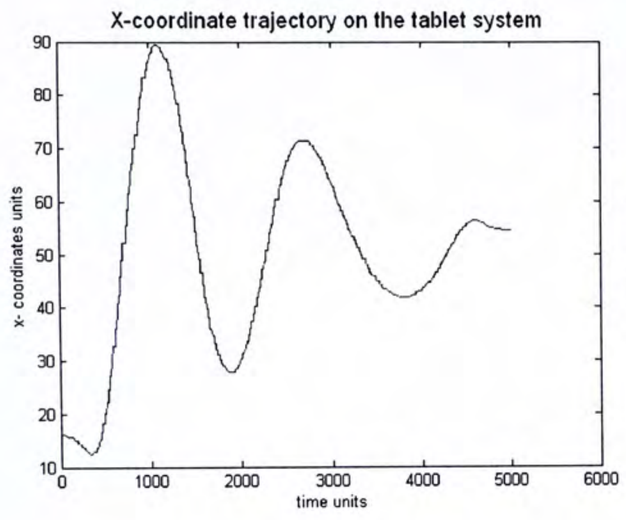
Figure 7.7. The footprint orientation angle from horizontal.

7.3 Experimental results

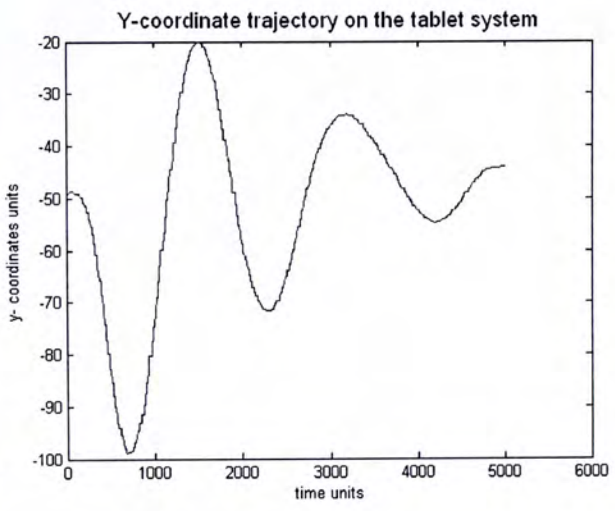
In this section, we will show the characterization results on two strokes painting in different form. To create the variation of stroke form user-friendly, we use the *Intuos*^{c}2 12 by 12 Tablet and pen system as shown in Figure 2.3 to capture the painting motion. Since full stroke characterization was utilized as starting point, the x- and y- translation painting motion as well as the writing pressure are just captured.

Result 1

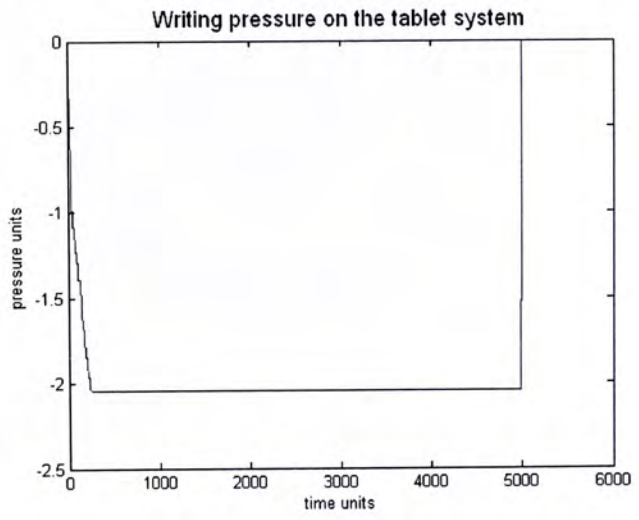
A spiral form painting trajectory is input through the tablet system. The (x, y) trajectory against time is shown in Figure 7.8(a) and Figure 7.8(b). As the writing pressure is captured, the z-axis painting command is mapped from 1 pressure unit to 1mm and the diagram of Figure 7.8(c) shows the writing pressure (limited ranging from 0 to -2 pressure units) measure on the tablet system. Upon the x-, y-, z-trajectory input, the would-be stroke painting stroke in spiral form is shown in Figure 7.9(a), and Figure 7.9(b) indicates all the instantaneous footprint centroid. After the footprint characterization process, the function of footprint area, major axis length, minor axis length and orientation angle are shown in Figure 7.10(a), Figure 7.10(b), Figure 7.10(c) and Figure 7.10(d) respectively.



(a)

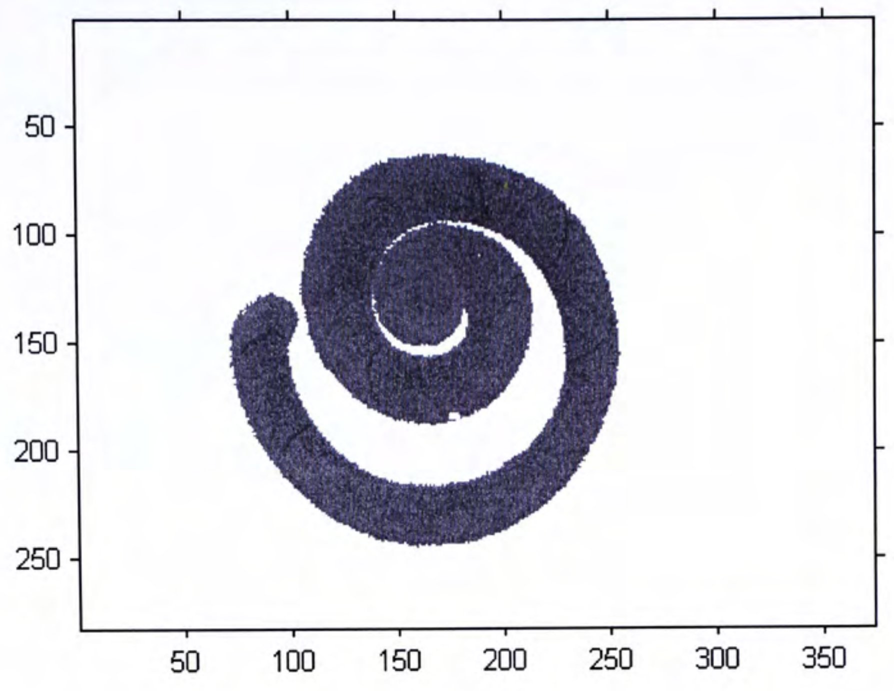


(b)

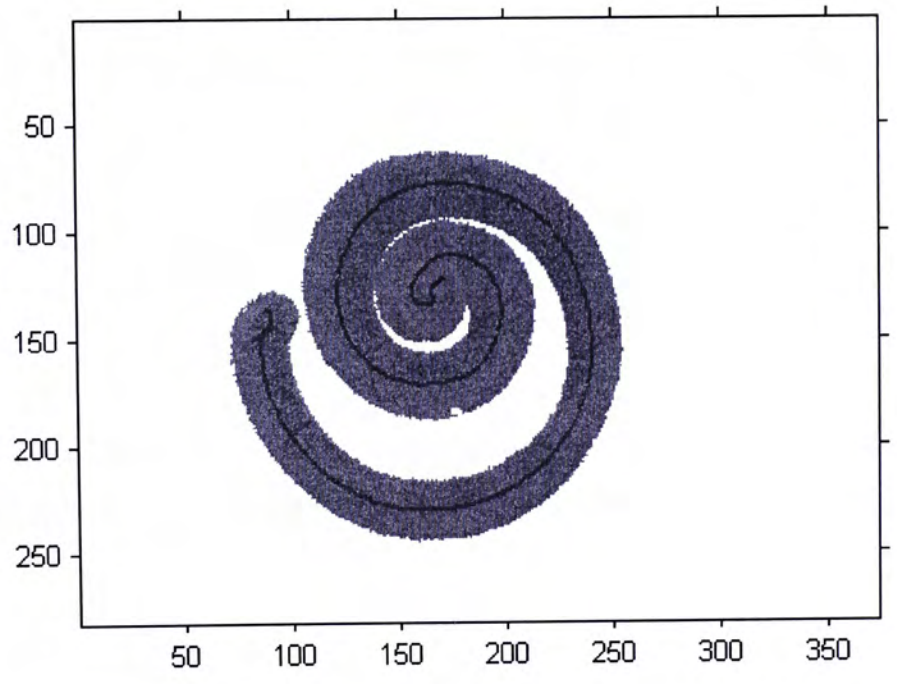


(c)

Figure 7.8. (a) X-coordinate trajectory diagram against time, (b) Y-coordinate trajectory diagram against time, (c) Pressure measure during writing



(a)



(b)

Figure 7.9. (a) The would-be full "painting" spiral stroke form, (b) with instantaneous footprint centroid indication

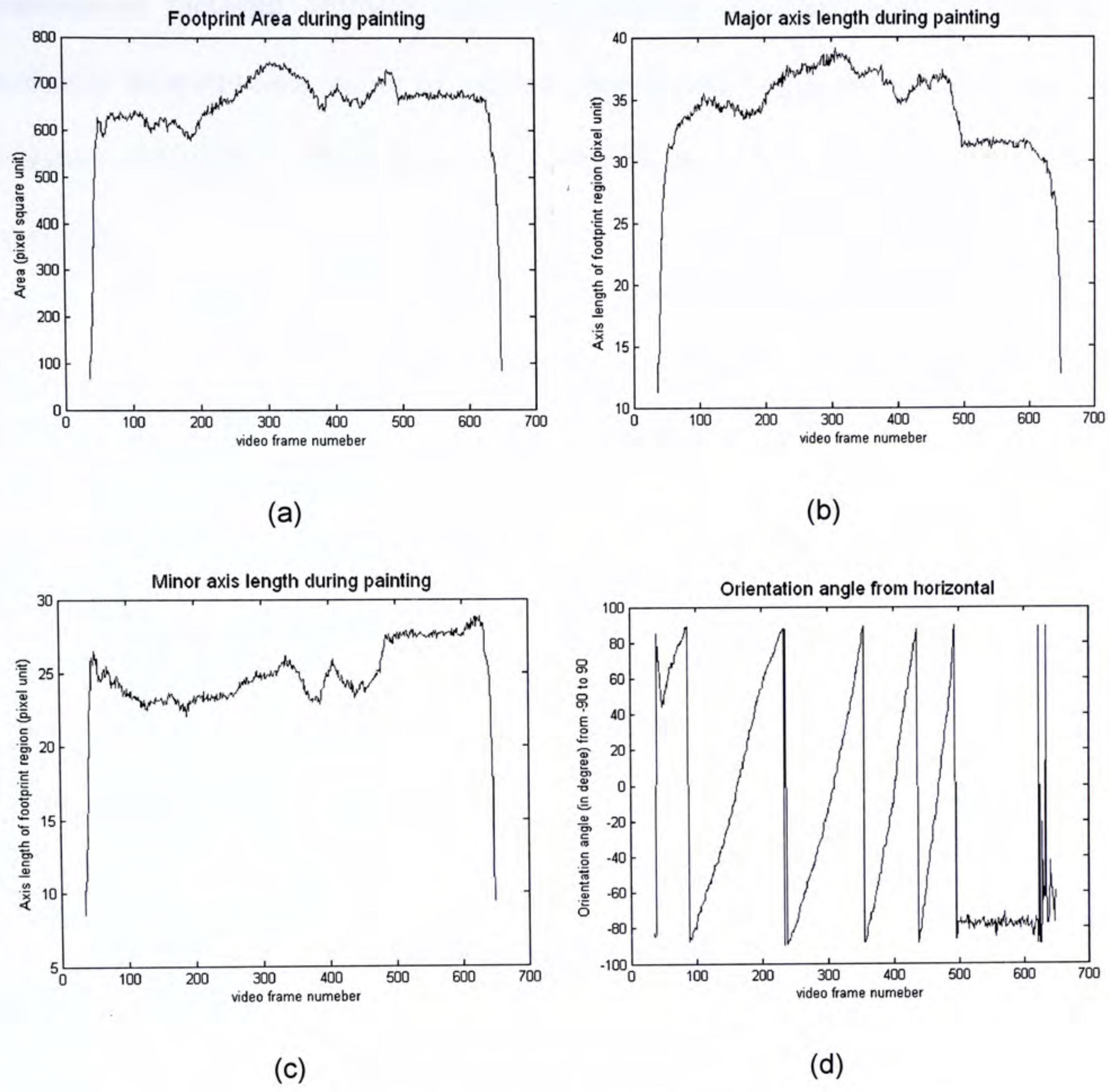


Figure 7.10(a). The diagram of footprint area during painting. (b). The diagram of major axis length. (c). The diagram of minor axis length diagram. (d). The diagram of orientation

Result 2

A spring form painting trajectory is input through the tablet system. The (x, y) trajectory against time is shown in Figure 7.11(a) and Figure 7.11(b). Same as the precious result, the z-axis painting command is mapped from 1 pressure unit to 1mm and the diagram of Figure 7.11(c) shows the writing pressure measure on the tablet system. Upon the x-, y-, z- trajectory input, the would-be stroke painting stroke in spring form is shown in Figure 7.12(a), and Figure 7.12(b) indicates all the instantaneous footprint centroid. After the footprint characterization process, the function of footprint area, major axis length, minor axis length and orientation angle are shown in Figure 7.13(a), Figure 7.13(b), Figure 7.13(c) and Figure 7.13(d) respectively.

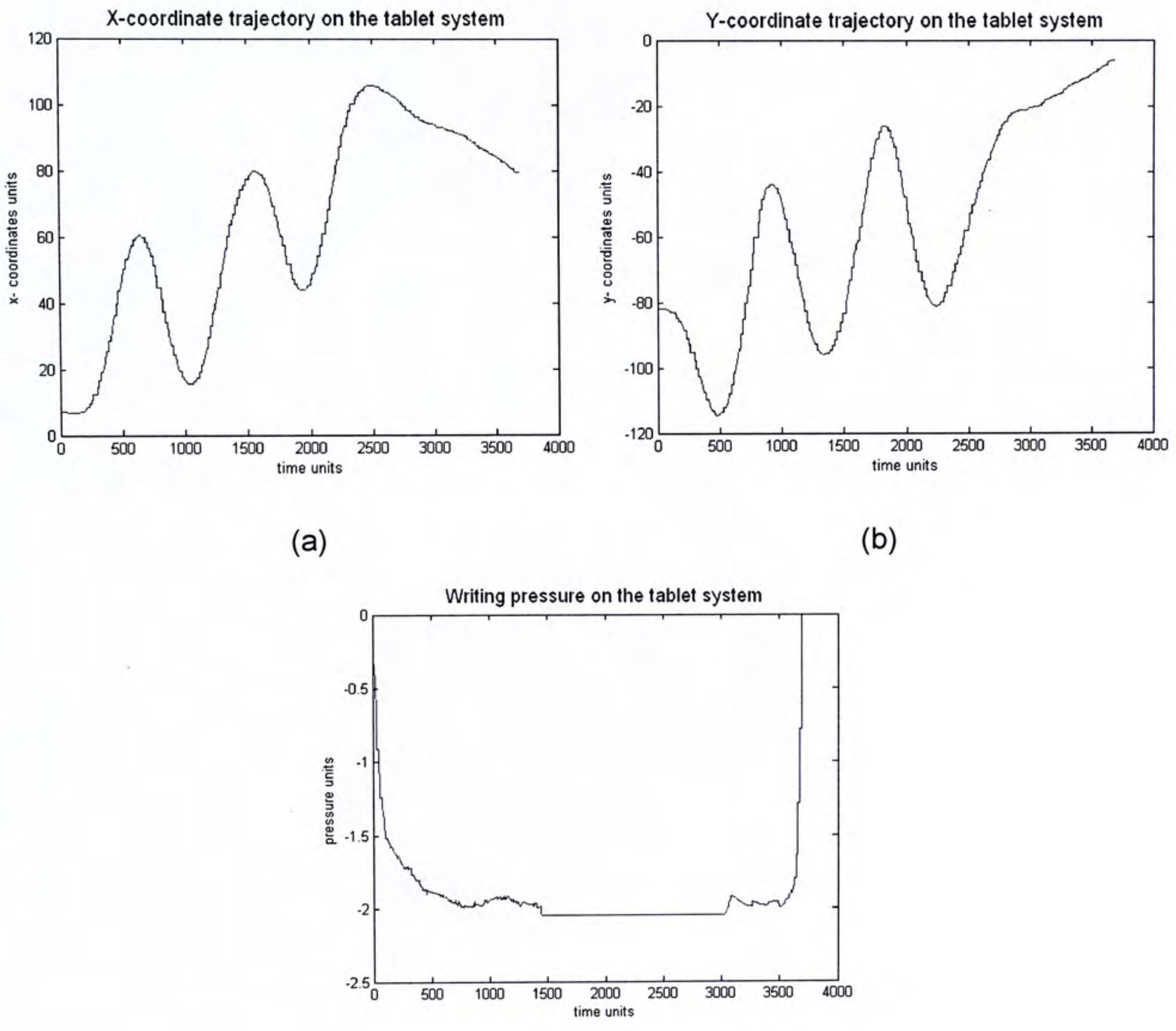
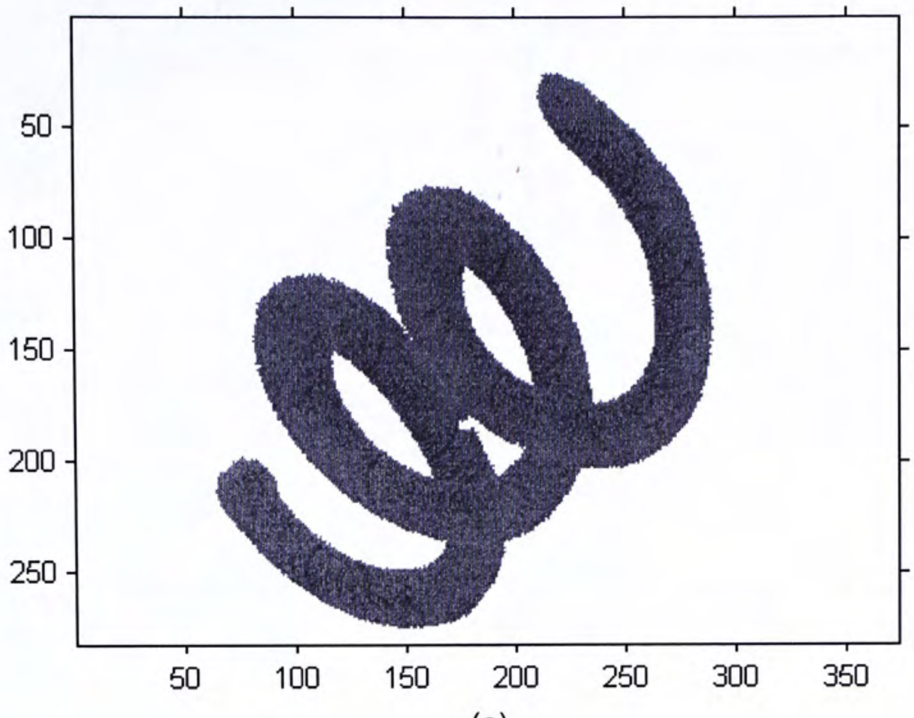
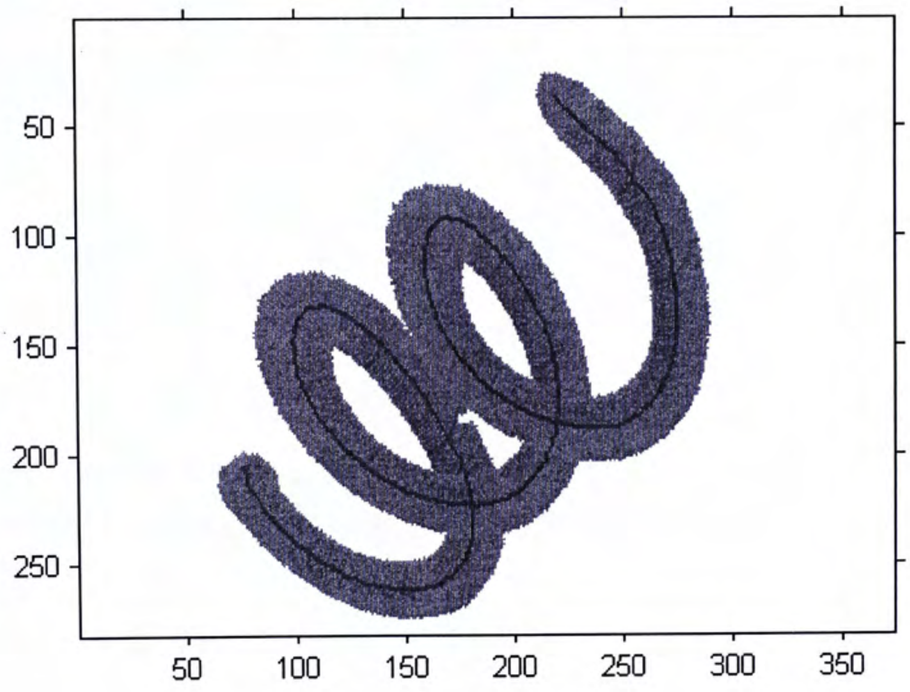


Figure 7.11. (a) X-coordinate trajectory diagram against time, (b) Y-coordinate trajectory diagram against time, (c) Pressure measure during writing

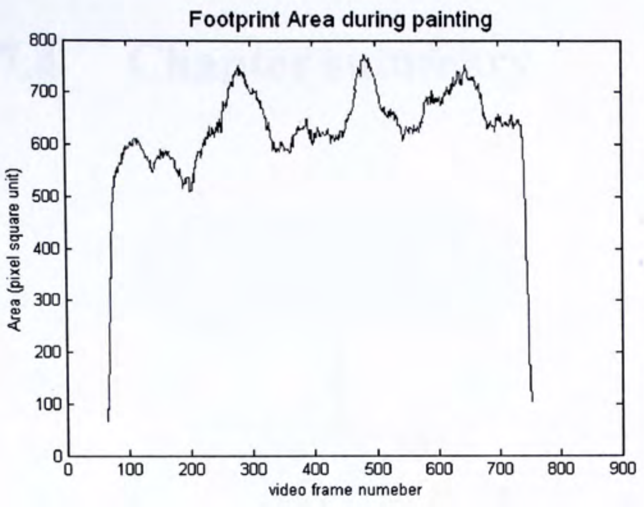


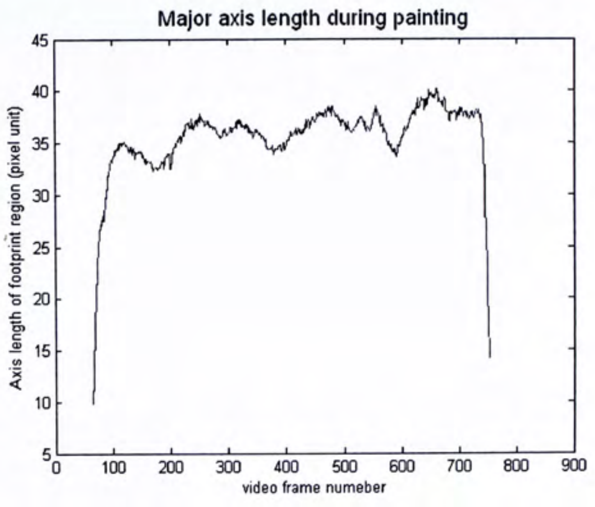
(a)



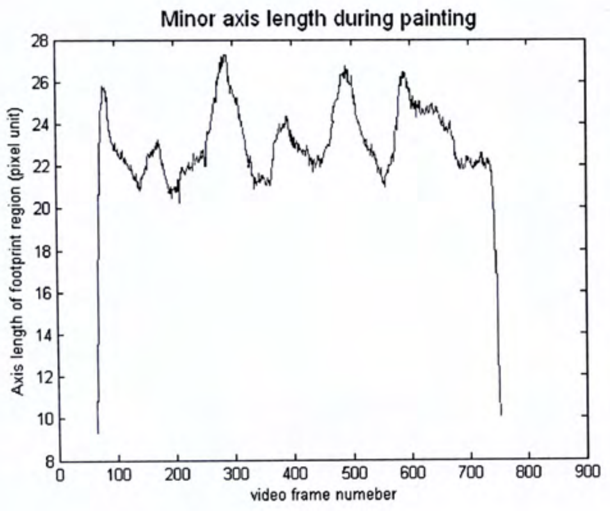
(b)

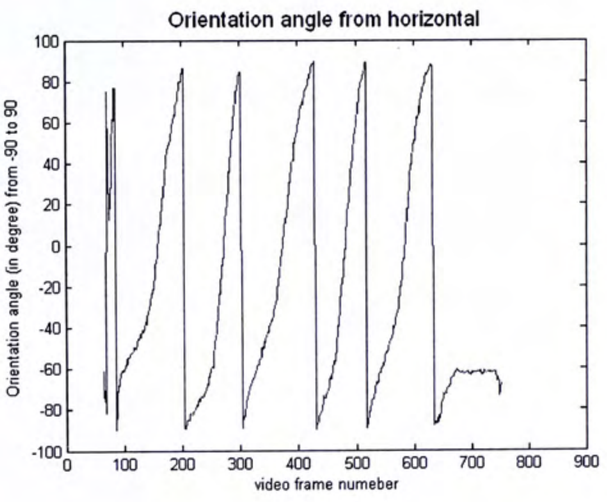
Figure 7.12 (a) The would-be full “painting” spring stroke form, (b) with instantaneous footprint centroid indication





(a) (b)





(c) (d)

Figure 7.13. (a) The diagram of footprint area during painting, (b) The diagram of major axis length, (c) The diagram of minor axis length diagram, (d) The diagram of orientation

7.4 Chapter summary

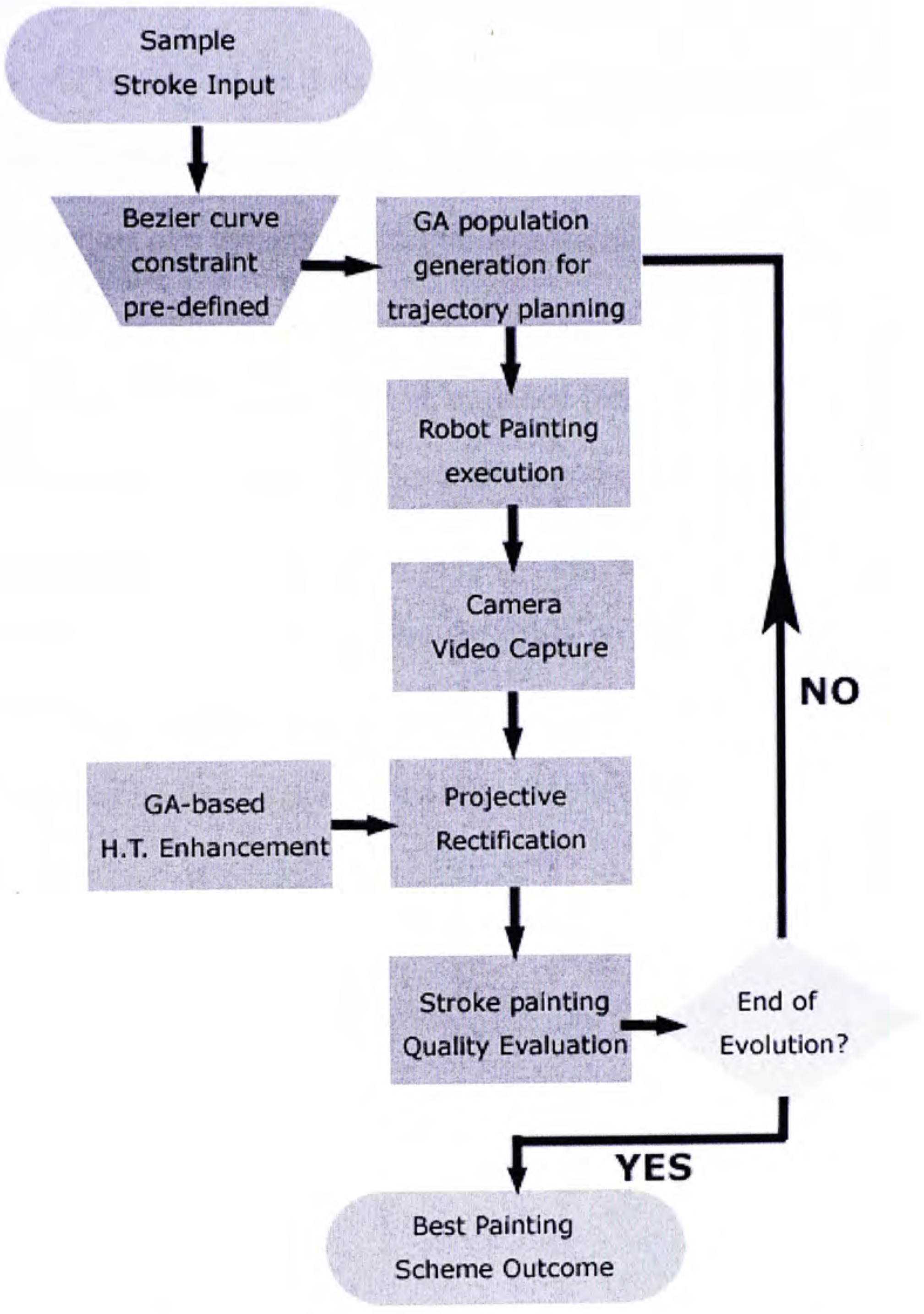


Figure 7.14. The process flow chart of visual-based full stroke execution using GA in finding the best replication painting scheme

By analyzing the brush stroke footprint, one hopes to obtain the realistic dynamic model of the brush pen motion. The camera system and the robot platform here provide us with a good environment (such as clean footprint image segmentation from the video captured and accurate brush manipulation) to study about this possibility. Here, footprint region area, centroid, major and minor axis length and orientation, those variety image region properties are introduced as basic representatives of the much complicated footprint resulted. These property parameters will be studied for their approximation relation to the brush movement. It is envisioned that the well known universal approximation theorem may be able to learn the behaviors of a dynamic model through training the input-output relations.

In our future work, we would like to develop the GA-based full stroke painting system in which the full strokes will be executed incorporating a real brush without its model knowledge. It extends the idea of visual-based iterative process that using GA painting to find the best painting scheme specific to given stroke input, except that the strokes are now generated by the robot on the paint container of Figure 2.8(b) or real paper rather than simulation. Thus, all GA individual's objective cost will be evaluated by those visial capabilities. The procedure as envisioned is shown in Figure 7.14.

8. CONCLUSIONS AND FUTURE WORKS

This chapter presents the conclusions of this thesis. Main ideas on the goals and motivation, to the approach we adopt, and to the implementation of the drawing system and experimental results are described in summary. Possible areas for future research are also outlined.

We have developed an intelligent drawing system which operated based on visual information, much like a child learns to write and draw by sight. The results demonstrate that capabilities developed can be successful in pinpointing the proper branch points of a line drawing over that generated merely by skeleton-based vectorization (using Bezier curve interpolation) and TSP solver (CE-Method). Branch point decision is important as it is related to the order of stroke formation and execution, which plays a vital role in Chinese calligraphy. The present iterative drawing approach is able to automatically recover the stroke thickness and connectivity, as well as compensating the offset effect due to the bristles' deformation. To facilitate this process, a vision feedback mechanism is installed which allows a captured image to be rectified in full plane view as the original input image. This high accuracy rectification is implemented automatically via GA-based homography transformation. The iterative drawing process as developed is shown in Figure 8.1.

For future exploration in full stroke calligraphy replication and execution, GA-based stroke generation is to be studied. All painting factors such as trajectory and pressure applied are to be regarded as chromosome. Upon the objective assigned, an appropriate painting scheme is to be found using the evolutionary computing. This is one topic for future direction. A second topic is the determination of a realistic brush tuft model. Such model is to be established experimentally by another camera system which captures the brush tuft "footprint" without ink depositing. By "footprint" image analysis, we hope to obtain the relationship between the robot brush kinematics control and the tuft deformation. An understanding on how the brush kinematics would affect the tuft "footprint" formation will be of great importance in our future work.

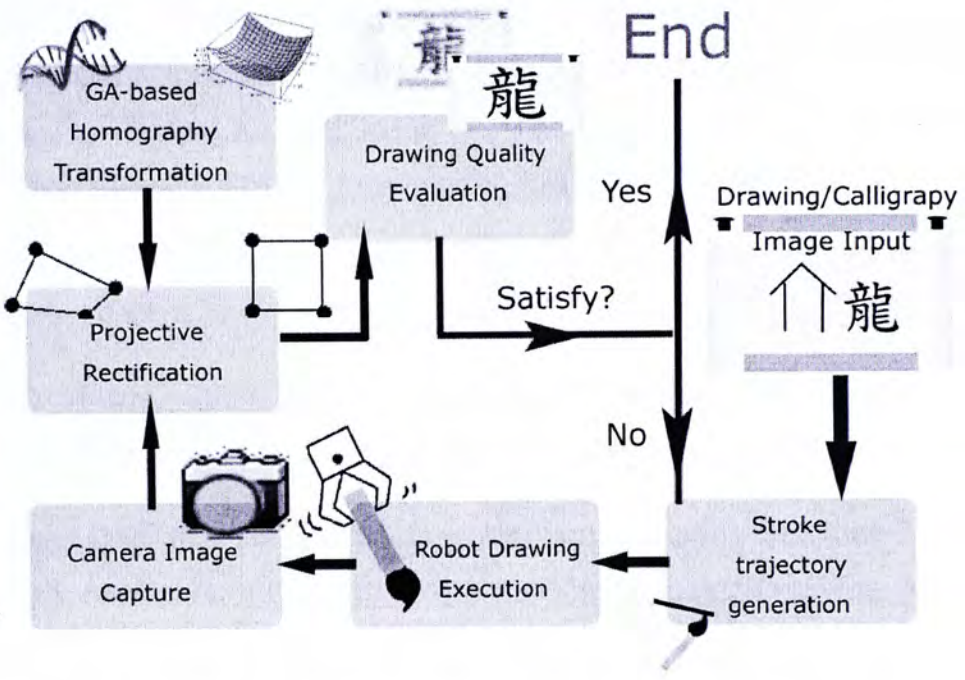

End
GA-based Homography Transformation
Drawing Quality Evaluation
Yes
Drawing/Calligrapy Image Input
Projective Rectification
Satisfy?
No
Camera Image Capture
Robot Drawing Execution
Stroke trajectory generation

Figure 8.1. Schematic diagram of iterative drawing process

In the present research, various algorithms have been developed and several new approaches have been proposed for tackling relevant technical problem. We demonstrated the effectiveness and practiccality of our proposed method on the acquisition and imitation of human artistry and the promising results. For future work, we will work on using 5 DOFs brush pen manipulation, combining with high accuracy visual feedback capabilies, appropriate brush and ink diffusion model, and GA evolutionary computing. We aim to execute full stroke calligraphy on paper with the objective of imitating the personalized drawing and calligraphy style of some famous artists.

BIBLIOGRAPHY

[1] Pagliarini, L., and Lund, H.H., *Art, Robots, and Evolution as a Tool for Creativity*, Creative Evolutionary Systems, P. Bentley and D. Corne (eds), Morgan Kaufman, 2001.

[2] Morgan, H., Harold Cohen's Aaron, The Robot as an Artist., [Online]. Available: http://www.scinetphotos.com/aaron.html

[3] Srikaew A., Cambron, M.E., Northrup S., Peters II, R.A., Wilkes, D.M., and Kawamura K., *Humanoid Drawing Robot*, IASTED International Conference on Robotics and Manufacturing, Banff, Canada, July 26-31, 1998.

[4] Morris, D., Phelps, K., and Joshi, N., The drawing teleoperated robot. [Online]. Available: http://techhouse.brown.edu/~neel/drawing_telerobot

[5] The annual international art exhibition for robotic art and art-making robots, The Robot Talent Show: ArtBots. [Online]. Available: http://artbots.org/2005/

[6] Leonel Moura, 2004 ArtBots participant: ArtSBot (Art Symbiotic roBots). [Online]. Available: http://artbots.org/2004/participants/ArtSBot/

[7] Fernando Orellana, 2003 ArtBots participant: Drawing Machine 3.1415926 v.2. [Online]. Available: http://artbots.org/2003/participants/Drawing_Machine/

[8] Jonah Brucker-Cohen, 2004 ArtBots participant: DrawBot Modding Contest and Workshop. [Online]. Available: http://artbots.org/2004/participants/DrawBot/

[9] McCorduck, P., *Aaron's Code: Meta-art, Artificial Intelligence and the work of Harold Cohen*, W.H. Freeman & Co., New York, New York Copyright 1991.

[10] P. Monaghan, "An art professor uses artificial intelligence to create a computer that draws and paints," *The Chronicle of Higher Education*, pp. 27-28, May 1997.

[11] Tao Guoliang, Wang Xuanyin, "Research on Pneumatic-Servo Calligraphy Robot", *5th International Conference on Fluid Power Transmission and Control (ICFP2001)*, 3-5 April, 2001, Hangzhou, China, 3_8.pdf.

[12] Fenghui Yao, Guifeng Shao, Ryoichi Takaue, Akikazu Tamaki, "Development of a Chinese Character Calligraphy Robot", Proceedings of the 9th IEEE Internatinal Conference on Mechatronics and Machine Vision in Practice, 2002 Chiang Mai, Tailand, Sept 10-12, 2002.

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

[14] 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.

[15] Dordevic, G.S., Rasic, M., Kostic, D., and Potkonjak, V., *Representation of Robot Motion Control Skill,* IEEE Transaction on Systems, Man, and Cybernetics – Part C: Systems and Applications, 30(2), pp. 219-237, May 2000.

[16] Song, J.Y., Xu, Y.S., Nechyba, M.C., and Yam, Y., *Transfer of Human Control Strategy Based on Similarity Measure,* Proceedings of 1999 IEEE international Conference on Robotics and Automation (ICRA'99), Detroit, Michigan, USA, May 10-15, 1999, pp.3134-3139.

[17] Nancy Doyle, Nancy Doyle Fine Art, Exercise 1 – Contour Drawing. [Online]. Available: http://www.ndoylefineart.com/drawexercise1.html

[18] Xiangyun Ye; Cheriet, M.; Suen, C.Y., "Stroke-model-based character extraction from gray-level document images", *Image Processing*, IEEE Transactions on , Volume: 10 , Issue: 8 , Aug. 2001,Pages:1152 – 1161

[19] Peters, R.A., II., "A new algorithm for image noise reduction using mathematical morphology", *Image Processing*, IEEE Transactions on , Volume: 4 , Issue: 5 , May 1995, Pages:554 - 568

[20] Mozer, M.C. and Smolensky, P., *Skeletonization: A technique for trimming the fat from a network via relevance assessment*, in Advances in Neural Information Processing Systems. Tourezky D.S., Ed. Vol. 1, Denver, CO. Morgan Kaufmann, pp. 107--115.

[21] 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* , Volume: 4 , 25-29 Aug. 1996, Pages:457 - 461 vol.4

[22] 墓誌銘 / [解說者中田勇次郎], *"書道藝術"*, 東京 ： 二玄社, 1967

[23] A. N. Kolesnlkov, V.V.Belekhov and I. O. Chalenko, "Vectorization of Raster Images", *Applied Problem of Pattern Recognition Image Analysis, and Signal Processing,* Volume 6, No. 4, 1996, pp. 786-794.

[24] Liang, S., Ahmadi, M., Shridhar, M, "Segmentation of interference marks using morphological approach", *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on,* Volume 2, 14-16 Aug. 1995 Page(s):1042 - 1046 vol.2

[25] John Y. Chiang, S. C. Tue and Y. C. Leu, "A New Algorithm For Line Image Vectorization", *Pattern Recognition*, Volume 31, Issue 10, October 1998, Pages 1541-1549

[26] Jiqiang Song, Min Cai, Michael R. Lyu, "Graphics Recognition from Binary Images: One Step or Two Steps", *16 th International Conference on Pattern Recognition (ICPR'02)*, Volume 3, August 2002, page(s): 135- 138

[27] Paul Bourke. (1996, Dec.). *Bezier curve*. [Online]. Available: http://astronomy.swin.edu.au/~pbourke/curves/bezier/

[28] Zhang Tianquan, 張天泉, *"Zhongguo wen zi hua 中國文字畫"*, 台北市：新文豐出版公司, 民國68[1979]

[29] 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.

[30] 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

[31] Liebowitz, D. and Zisserman, A., "Metric Rectification for Perspective Images of Planes", In Proceedings of the *Conference on Computer Vision and Pattern Recognition*, pages 482-488, June, 1998.

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

[33] Richard Hartley and Andrew Zisserman, *Multiple View Geometry in Computer Vision Second Edition*, Cambridge University Press, March 2004.

[34] K.W. Kwok , Y. Yam, K.W. Lo, "GA-based homography Transformation for Vision Rectification in Robot Drawing System", Accepted for publication in Proceeding of *the 44th IEEE Conference on Decision and Control and European Control Conference ECC 2005*

[35] Chin-Teng Lin , C. S. George Lee, *Neural fuzzy systems: a neuro-fuzzy synergism to intelligent systems*, Prentice-Hall, Inc., Upper Saddle River, NJ, 1996

[36] 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.

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

[38] Songhua Xu, Francis C.M. Lau, William K. Cheung, Yunhe Pan, "Automatic Generation of Artistic Chinese Calligraphy," *IEEE Computer Society*, pp32-39, 2005.

[39] J. Lee., "Simulating Oriental Black-ink Painting", *IEEE Computer Graphics and Applications*, 19(3), pp. 74-81, May/June 1999.

[40] Helena T.F. Wong, Horace H.S. Ip, "Virtual brush: a model-based synthesis of Chinese calligraphy," *Computers & Graphics*, 24, 2000. pp99-113

[41] 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.*

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

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