

# Image processing and reconstruction of cultured neuron skeletons

Donggang Yu<sup>a</sup>, Tuan D. Pham<sup>b,\*</sup>, Hong Yan<sup>c,d</sup>, Jesse S. Jin<sup>a</sup>, Suhuai Luo<sup>a</sup> and Denis I. Crane<sup>e,f</sup>

<sup>a</sup>*School of Design, Communication and Information Technology, The University of Newcastle, Callaghan, NSW 2308, Australia*

<sup>b</sup>*ADFA School of Information Technology and Electrical Engineering, The University of New South Wales, Canberra, ACT 2600, Australia*

<sup>c</sup>*Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong*

<sup>d</sup>*School of Electrical and Information Engineering, University of Sydney, NSW 2006, Australia*

<sup>e</sup>*School of Biomolecular and Biomedical Science*

<sup>f</sup>*Eskitis Institute for Cell and Molecular Therapies, Griffith University, Nathan, Qld 4111, Australia*

**Abstract.** One approach to investigating neural death is through systematic studies of the changing morphology of cultured brain neurons in response to cellular challenges. Image segmentation and neuron skeleton reconstruction methods developed to date to analyze such changes have been limited by the low contrast of cells. In this paper we present new algorithms that successfully circumvent these problems. The binary method is based on logical analysis of grey and distance difference of images. The spurious regions are detected and removed through use of a hierarchical window filter. The skeletons of binary cell images are extracted. The extension direction and connection points of broken cell skeletons are automatically determined, and broken neural skeletons are reconstructed. The spurious strokes are deleted based on cell prior knowledge. The reconstructed skeletons are processed furthermore by filling holes, smoothing and extracting new skeletons. The final constructed neuron skeletons are analyzed and calculated to find the length and morphology of skeleton branches automatically. The efficacy of the developed algorithms is demonstrated here through a test of cultured brain neurons from newborn mice.

**Keywords:** Neuron cell image, image segmentation, grey and distance difference, filtering window, neuron skeleton, skeleton reconstruction, skeleton branch

## 1. Introduction

The molecular pathogenesis of neurodegeneration is poorly understood. These challenges include high energy demand, the length of the neural axon and dendrites – and resulting susceptibility to cytoskeletal transport defects – and their high metabolic rate, which together with their relative paucity of antioxidant capacity, makes them highly susceptible to damage caused by reactive oxygen species (ROS) [1–3].

A number of methods have been developed for analysis of neuron extraction and imaging. These include

an algorithm for fast, automatic extraction of neurite structures based on soma segmentation, seed point detection, recursive center line detection, and 2D-curve smoothing [4], an automated neurite analysis method for extracting single and connected centerlines along neurites [5], and an interactive technique for the tracing and quantification of elongated image structures [6].

In order to binary object images from poor quality images, it is essential to threshold the image reliably. Although many thresholding techniques, such as global and local thresholding algorithms, multi thresholding methods [7–10] and unimodal thresholding [11] have been developed in the past, it is still difficult to deal with images with very low quality. Such a sample cell image is shown in Fig. 1(1), and its histogram is shown

---

\*Corresponding author. E-mail: t.pham@adfa.edu.au.

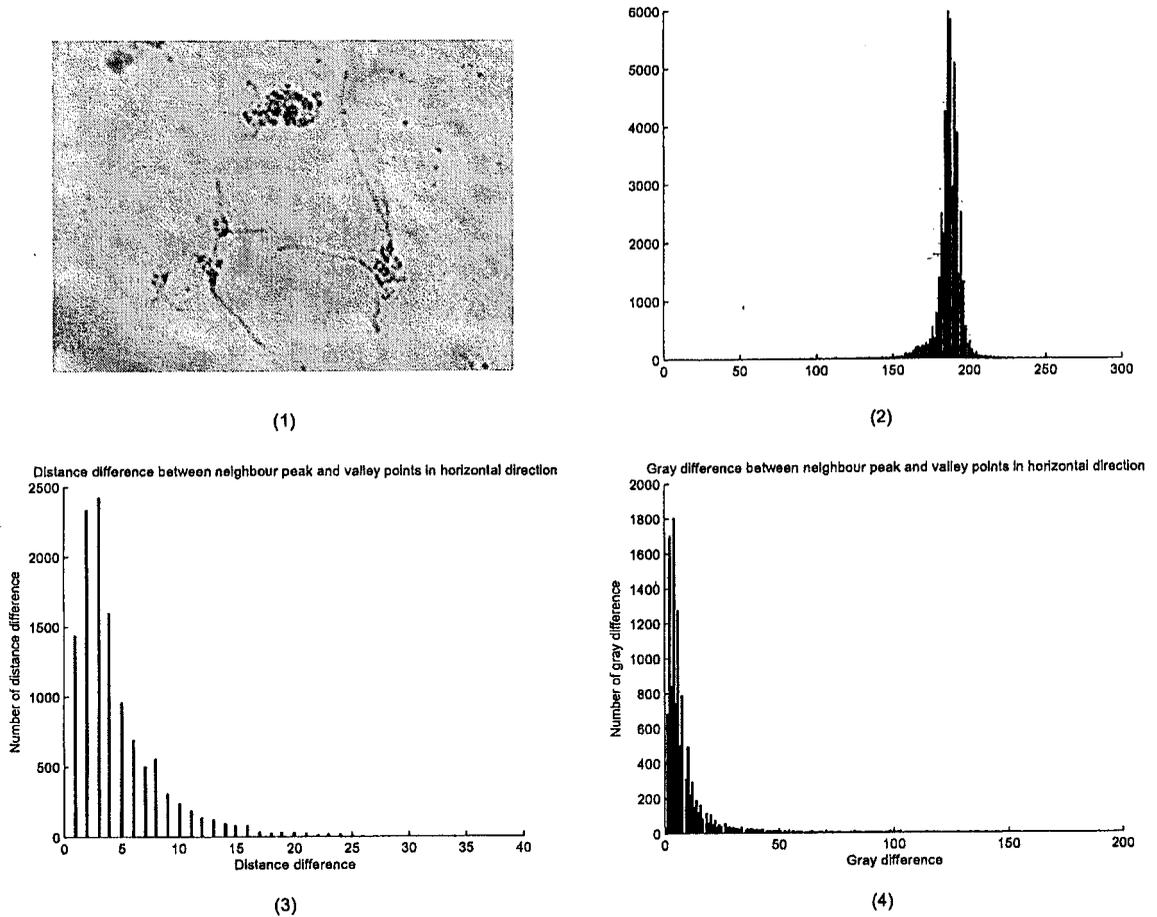


Fig. 1. The typical image obtained from monitoring a live, unstained, neuron in culture ( $T = 180$  min.) and its histogram, and the distance and grey difference histogram of image in Fig. 1(1) in the horizontal direction. (1) The typical neuron image. (2) The histogram of the neuron image. (3) Distance difference. (4) Grey difference.

in Fig. 1(2). We can see that its histogram is unimodal because of poor background.

Information taken from images of neuron cells being grown in culture with oxidative agents allows life science researchers to compare changes in neurons. It is clear that image analysis and recognition are useful tool to help our study of the neuron degeneration in a human disorder called Zellweger syndrome. In morphological terms, we expect to see this initial deterioration as the contraction, and eventually loss, of processes of neurons grown in culture. Therefore, not only segment neuron cell images, but also it is important to reconstruct broken neuron cell skeletons caused by segmentation of neuron cell images with more poor background than that of some neuron images in the previous research methods [1–3].

In this paper, preprocessing procedure, segmentation of neuron images with poor background, is introduced in Section 2. In Section 3, the reconstruction of cell skeletons is developed. We then conclude our analyzer in the final section.

The contribution of the paper is that new segmentation method is firstly used to segment the cultured neuron cell images, novel reconstruction and analysis of neuron skeleton are firstly developed and the neuron skeleton length (neural axons) is automatically analyzed and calculated.

## 2. Preprocessing

Although many segmentation methods [7–10] were developed to binary grey images, but it is still difficult

to deal with images with poor back ground. In this paper, the novel methods logical level technique with difference analysis of grey region and filtering window with contained condition are used to segment these neuronal images [12].

### 2.1. Logical level technique

Logical level technique are developed to be used to segment document images by Kamel, Zhao [13], Yang and Yan [14]. After analyzing integrated function algorithm [15], Kamel and Zhao proposed Logical level technique. The basis idea is comparing the gray level of the processed pixel or its smoothed gray level with some local averages in the neighborhoods, and the comparison results are regarded as derivatives. Therefore, pixel labeling, detection and extraction using the derivatives, the logical bound on the ordered sequences and the window width range can be adopted. This technique processes each pixel by simultaneously comparing its gray level or its smoothed gray level with four local averages in the selected window region. Suppose selected window is  $W$ . The window region is  $(2W+1)^2$ . Let the start point of the image be upper-left and  $f(i, j)$  be grey intensity of coordinates  $(x, j)$ , and it is eight neighboring.

Suppose each neighbor point  $(x, y)$  is the center of region  $(2W+1)^2$ , then the average grey intensity  $lp(k)$  of region  $(2W+1)^2$  is

$$lp(n) = \frac{\sum_{-W \leq m \leq W} \sum_{-W \leq n \leq W} f(x+m, y+n)}{(2W+1)^2} \quad (1)$$

where if  $k=0, x=i$  and  $y=j+1$ ;  $k=1, x=i-1$  and  $y=j+1$ ;  $k=2, x=i-1$  and  $y=j$ ;  $k=3, x=i-1$  and  $y=j-1$ ;  $k=4, x=i$  and  $y=j-1$ ;  $k=5, x=i+1$  and  $y=j-1$ ;  $k=6, x=i+1$  and  $y=j$ ;  $k=7, x=i+1$  and  $y=j+1$ . Therefore grey region difference ( $llp(k)$ ) between  $lp(k)$  and  $f(i, j)$  can be found

$$llp(k) = lp(k) - f(i, j) \geq T \quad k=0, 1, \dots, 7 \quad (2)$$

where  $T$  is predetermined parameter.

The logical level technique is

$$b(i, j) = \begin{cases} 1 & \text{if } (llp(0) \wedge llp(4)) \vee (llp(2) \wedge llp(6)) \vee (llp(1) \wedge llp(5)) \vee (llp(3) \wedge llp(7)) \text{ is true} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where "1" represents object and "0" to represent background in the resulting binary image.

### 2.2. Innovative logical level technique with difference analysis of grey region

We can find that logical level technique need two key parameters, window " $W$ " and threshold " $T$ ". However their predetermination is difficult, and no efficient method can be found from logical level technique. In fact, the essence of image segmentation is the determination of grey regions. Therefore, we approach new method to determine parameters, window " $W$ " and threshold " $T$ " automatically based on the analysis grey regions. Grey region can be defined as the region between each pair of neighbor grey peak and valley points in horizontal and vertical direction. Mathematically, the peak and valley points of image grey histogram are the points which make the first order derivative of image grey function equal to zero. Therefore, each row the peak and valley point sets,  $P_h, V_h$ , and each column peak and valley point sets,  $P_v$  and  $V_v$  can be found.

Grey regions can be calculated based on found peak and valley point sets,  $P_h, V_h, P_v$  and  $V_v$ . For each grey region two parameters are calculated. One is grey difference between each pair of peak and valley points, which can be represented as  $H_g(m), m=1, 2, \dots, k$ , where  $k$  is region number for all rows of an image. Another one is distance difference between each pair of peak and valley points, which can be represented as  $H_d(m), m=1, 2, \dots, k$ . Furthermore, one new data set of grey region in which the number of points that have same grey difference and region distance is found. It can be represented with  $H_{dg}(m), m=1, 2, \dots, kn$ .

Sort  $H_{dg}(m), m=1, 2, \dots, kn$  based on  $H_{dg}(m)$  get a decreasing data set,  $H_{dgd}(m), m=1, 2, \dots, kn$ . Therefore  $H_{dgd}(0)$  is the first number of grey regions with same grey and distance difference, and it is largest. If first  $tk$  groups are summed

$$S_{tk} = \sum_{m=1}^{tk} (H_{dgd}(m)). \quad (4)$$

Parameter  $tk$  is selected to meet  $(S_{tk}/k)$  is approximately equal to 30%, where  $k$  is region number for all rows of an image. For example, for the image in Fig. 1(1)  $tk=10, kn=931, (S_{tk}=3125$  and  $k=11961$ . We can see only 10 groups of grey region contain 3125 grey regions which is approximately equal to 30% of  $k=11961$ . Therefore, here 10 groups of grey region represent major property of region distribution of the image. Based on this idea parameter  $W$  and threshold  $T$  can be determined.  $W$  and  $T$  are selected as mean region distance and region grey difference of  $tk$  groups of grey region respectively. That is

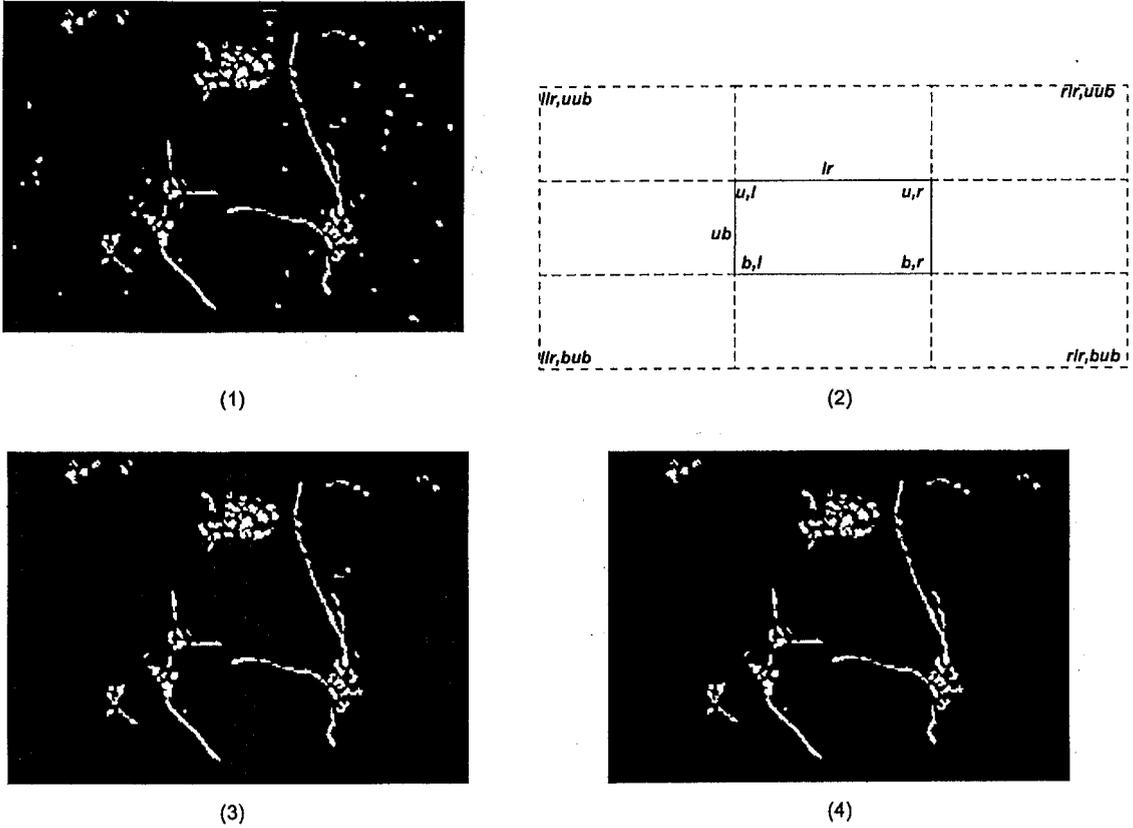


Fig. 2. The separation result of the image in Fig. 1, filter window and the processing results using filter windows with one object and two objects respectively. (1) The separation result. (2) Filtering window. (3) Using filtering window with one object. (4) Using filtering window with two objects.

$$W_h = \frac{\sum_{m=1}^{tk} (H_{dd})(m)}{tk} \tag{5}$$

$$T_h = \frac{\sum_{m=1}^{tk} (H_{gd})(m)}{tk} \tag{6}$$

where  $(H_{dd})(m)$  and  $(H_{gd})(m)$  is region distance and grey difference of each group of  $tk$  groups in horizontal direction respectively.

Similarly, the peak and valley points, related analysis parameters,  $W_v$  (window parameter in vertical direction) and  $T_v$  (thresholding parameter in vertical direction) in vertical direction of images can be found. The final window parameter is  $W = (W_h + W_v)/2 + 2$ , where being plus 2 is because of considering the influence of single point noise, and thresholding parameter is  $T = T_h + T_v)/2$ . The distance and grey difference histograms of image in Fig. 1(1) in the horizontal is shown in Fig. 1(3-4). Based on the found parameters,  $W$ ,  $T$  and logical thresholding algorithm, the images in Fig. 1(1) can be binarized, and shown in Fig. 2(1).

### 2.3. Filtering window

As average smoothed grey and grey difference information of image window is used, the algorithm can binarized poor quality. However, if a region meets the conditions of binarization, it can be selected as an object image. For example some spurious regions are made in Fig. 2(1). Usually valuable object image has big size, and one part of object image is close to its neighbor parts. Spurious region is usually isolated little region. For example, neuron images belong to such a case. Therefore, we can detect and remove some spurious region based the above idea.

The algorithm can be described as follows:

- (1) Find all regions of binarization image, which can be represented as  $R(k)$ ,  $k = 1, \dots, rn$ , where  $rn$  is the number of regions and  $R(k)$  is  $k$ -th region's area size.

|                                                                                |                                                                          |                                                                                |                                                                                |                                                                          |                                                                                |                                                                                |                                                                          |                                                                                |                                                                                |                                                                          |                                                                                |
|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|
| $\begin{matrix} (i-1,j-1) \\ x \\ (i-1,j) \\ 1 \\ (i-1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i,j-1) \\ x \\ (i,j) \\ 1 \\ (i,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ 0 \\ (i+1,j) \\ x \\ (i+1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ x \\ (i-1,j) \\ 1 \\ (i-1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i,j-1) \\ 1 \\ (i,j) \\ 1 \\ (i,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ x \\ (i+1,j) \\ x \\ (i+1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ 0 \\ (i-1,j) \\ x \\ (i-1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i,j-1) \\ x \\ (i,j) \\ 1 \\ (i,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ x \\ (i+1,j) \\ 1 \\ (i+1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ x \\ (i-1,j) \\ x \\ (i-1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i,j-1) \\ 1 \\ (i,j) \\ 1 \\ (i,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ x \\ (i+1,j) \\ 1 \\ (i+1,j+1) \\ x \end{matrix}$ |
| $\begin{matrix} (i-1,j-1) \\ 0 \\ (i-1,j) \\ 1 \\ (i-1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i,j-1) \\ 0 \\ (i,j) \\ 0 \\ (i,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ 0 \\ (i+1,j) \\ 1 \\ (i+1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ 0 \\ (i-1,j) \\ 1 \\ (i-1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i,j-1) \\ 1 \\ (i,j) \\ 0 \\ (i,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ 0 \\ (i+1,j) \\ 0 \\ (i+1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ 0 \\ (i-1,j) \\ 1 \\ (i-1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i,j-1) \\ 1 \\ (i,j) \\ 0 \\ (i,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ 0 \\ (i+1,j) \\ 0 \\ (i+1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ 0 \\ (i-1,j) \\ 0 \\ (i-1,j+1) \\ 0 \end{matrix}$ | $\begin{matrix} (i,j-1) \\ 1 \\ (i,j) \\ 0 \\ (i,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ 0 \\ (i+1,j) \\ 1 \\ (i+1,j+1) \\ 0 \end{matrix}$ |

(1) The models for determining and deleting corner points.

|                                                                                |                                                                          |                                                                                |                                                                                |                                                                          |                                                                                |                                                                                |                                                                          |                                                                                |                                                                                |                                                                          |                                                                                |
|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|
| $\begin{matrix} (i-1,j-1) \\ x \\ (i-1,j) \\ 1 \\ (i-1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i,j-1) \\ x \\ (i,j) \\ 1 \\ (i,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ 1 \\ (i+1,j) \\ x \\ (i+1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ x \\ (i-1,j) \\ 1 \\ (i-1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i,j-1) \\ 1 \\ (i,j) \\ 1 \\ (i,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ x \\ (i+1,j) \\ x \\ (i+1,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ 1 \\ (i-1,j) \\ x \\ (i-1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i,j-1) \\ x \\ (i,j) \\ 1 \\ (i,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ x \\ (i+1,j) \\ 1 \\ (i+1,j+1) \\ x \end{matrix}$ | $\begin{matrix} (i-1,j-1) \\ x \\ (i-1,j) \\ x \\ (i-1,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i,j-1) \\ 1 \\ (i,j) \\ 1 \\ (i,j+1) \\ 1 \end{matrix}$ | $\begin{matrix} (i+1,j-1) \\ x \\ (i+1,j) \\ 1 \\ (i+1,j+1) \\ x \end{matrix}$ |
|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------|

(2) The models for saving corner points.

Fig. 3. The pattern models for the smoothing skeleton.

- (2) Sort  $R(k), k = 1, \dots, rn$  based their area size in increasing order, and it is represented as  $SR(k), k = 1, \dots, rn$ .
- (3) From the first region: (3.1) find minimum rectangle which can cover the region, which is call as filtering window; (3.2) find a new region which consists of the region found in Step (3.1) and its eight neighbor regions; (3.3) detect whether there is the point of another object region in the new region, and then remove the processed region if not, keep the processed region if yes. Detection region is shown in Fig. 2(2), where  $(l, u), (r, u), (l, b)$  and  $(r, b)$  are the coordinates of four corners of found minimum rectangle respectively,  $lr = r - l$  and  $ub = b - u$  are sizes of the rectangle respectively, and  $llr = l - lr, rlr = r + lr, uub = u - ub$  and  $bub = b + ub$ .

Based on the above algorithm, the images in Fig. 2(1) can be processed and shown in Fig. 2(3). Hierarchical processing can be done based on new filtering window with two neighbor regions. The processing method can be described similar to above algorithm. Based on the above procedure, the images in Fig. 2(3) can be processed and shown in Fig. 2(4).

### 3. Reconstruction and extraction of neuron skeletons

After segmentation of neuron images, there are some broken neuron strokes (see Fig. 2(4)). In order to study the length of the neural axon and dendrites, it is necessary to reconstruct broken neuron skeletons and remove some spurious skeletons which do not belong to neurons. The skeletons of segmented image can be extracted by a fast parallel thinning algorithm [16].

#### 3.1. Smoothing of neuron skeletons

Generally, skeletons of binary images are not smooth because there are some spurious points, which make it difficult to extract skeleton features for reconstruction [17]. Therefore, it is useful to remove spurious points by smoothing skeletons. Also, the connection of skeleton should be retained. The skeleton can further be smoothed based on some patterns as shown in Fig. 3(1), where "0" represents background pixel, "1" represents a skeleton pixel and "x" stands for "don't care" point. In this way, some "corner" points of the

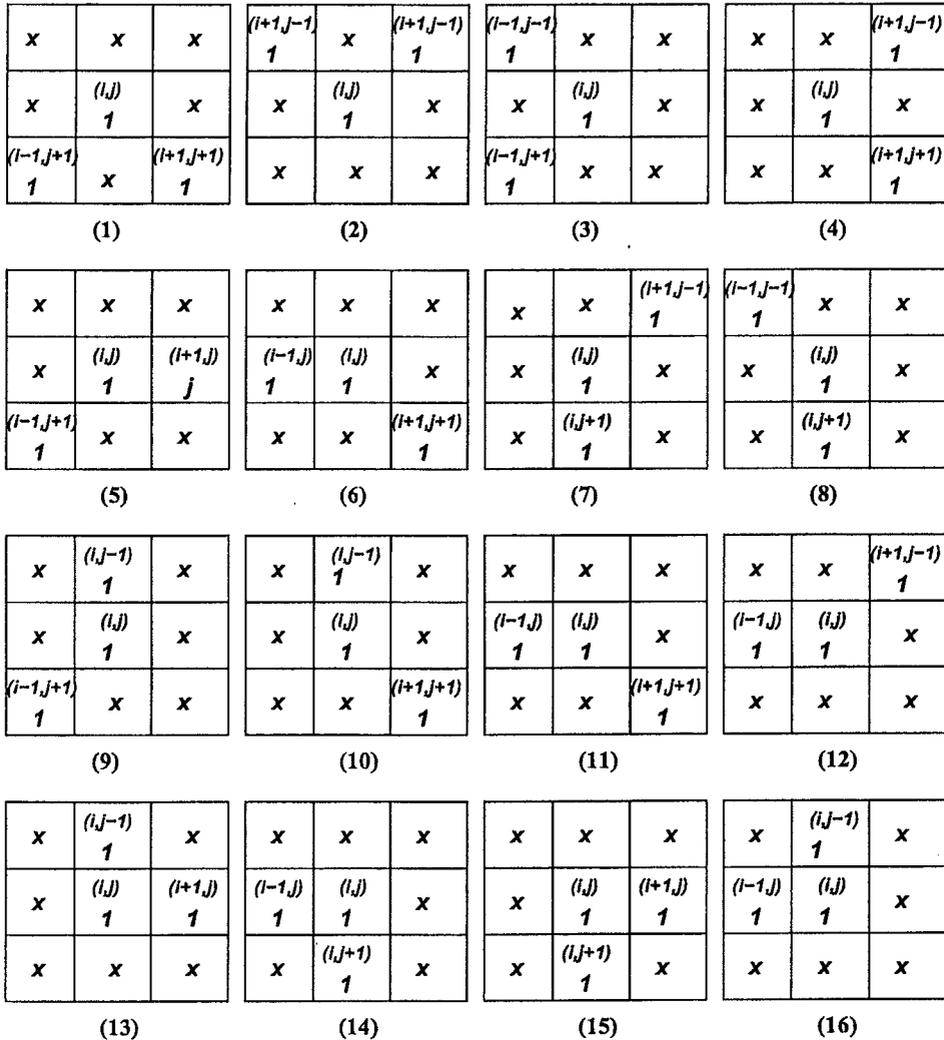


Fig. 4. The pattern models of "junction" points.

skeletons are removed. In order to retain the continuity of the skeletons, if the pixels of the skeleton belong to one of the patterns in Fig. 3(2), they are saved.

There are three types of skeleton point. If there is only one neighbor point for one skeleton point, then the skeleton point is called as "end" point. If there are three neighbor points for one skeleton point, then the skeleton point is called as "junction" point. Another smoothing procedure is as follows. If a "junction" point is belong to one pattern in Fig. 4, where the coordinates of two neighbor points of the "junction" point are given and "x" stands for "don't care" point, then let one neighbor point of a "junction" point as a center point of eight-neighboring. For example, a "junction" point is

belong to one pattern in Fig. 4(1) and its neighbor point  $(i-1, j+1)$  is the center point of eight-neighboring shown in Fig. 5(1). If its eight-neighbor points are all zero except its neighbor "junction" point (see Fig. 5(1)) then the neighbor point is spurious, and deleted. For the pattern in Fig. 4(13), one neighbor point  $(i, j-1)$  is the center point of eight-neighboring shown in Fig. 5(2). If its eight-neighbor points are all zero except its neighbor "junction" point and another neighbor point of the "junction" point, then the neighbor point is spurious, and deleted.

Based the above algorithms, the skeleton of cell image in Fig. 2(4) is show in Fig. 6(1), and its smoothed



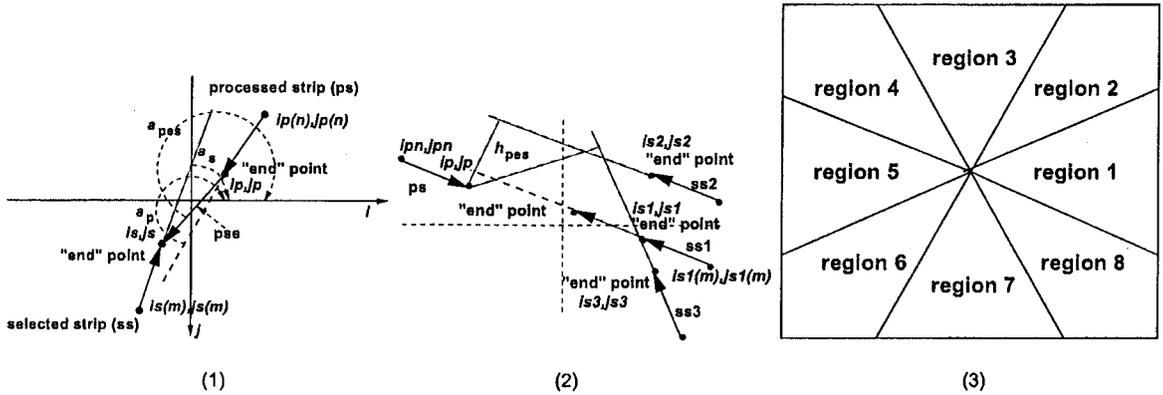


Fig. 7. The selection of connection strokes and moving strip of skeleton. (1) The angles and angle difference of strips of broken points. (2) Moving strip of skeleton. (3) Moving regions.

(( $ip(n), jp(n)$ )) to the first point ( $ip, jp$ ) and from (( $is(m), js(m)$ )) to ( $is, js$ ) and  $i$  axis respectively (see Fig. 7(1)); (b) find the direction angle between the connection line  $pse$  (from the "end" point of  $ps$  to that of  $ss$ ) and  $i$  axis,  $a_{pes}$ ; (c) determine the reverse angle of the angle  $a_s$ , representing as  $a_{sr}$ , and calculate the angle differences,  $ad_{pps} = |a_p - a_{pes}|$  and  $ad_{srps} = |a_{sr} - a_{pes}|$ . (d) find the total angle difference,  $t_{ad} = ad_{pps} + ad_{srps}$ . The calculation of all angles is demonstrate in Fig. 7(1).

- (4) The strip  $ss$  extends in the direction of the angle  $a_s$  by one strip distance, all distances between the "end" point of the strip  $ps$  and each point in the strip  $ss$  are found, and then the minimum distance,  $d_{smpse}$ , is determined. One example is shown in Fig. 7(2) where the strip  $ss1$  is extended.
- (5) Find the distance  $h_{pes}$  which is between the "end" point in  $ps$  and the line which is determined by strip  $ss$ . One example is demonstrated in Fig. 7(2) where the distance  $h_{pes}$  is the distance between the "end" point in  $ps$  and the line which is determined by the strip  $ss2$ .

The selected broken points are based on the above feature parameters. If the "end" points  $pe$  and  $se$  are selected as broken points, their related parameters should meet following conditions: (a)  $d_{smpse} < d_{ps}$ , and that means the strip  $ss$  is close to the strip  $ps$  after the strip  $ss$ 's moving in the direction of the angle  $a_s$ ; (b)  $t_{ad} < 50^\circ$  and  $h_{pes} < 7$ , or  $t_{ad} < 70^\circ$  and  $h_{pes} < 5$ , and that means the strip  $ss$  is not offset from the strip  $ps$  too much, and the less  $t_{ad}$ , the bigger  $h_{pes}$  is permitted; (c) In eight regions (shown in Fig. 7(3)), angles

$a_p$  and  $a_s$  meet following condition: (1)  $0^\circ \leq a_s < 22.5^\circ$  or  $337.5^\circ \leq a_s < 360^\circ$  and  $135^\circ \leq a_p < 225^\circ$ , (2)  $22.5^\circ \leq a_s < 67.5^\circ$  and  $180^\circ \leq a_p < 270^\circ$ , (3)  $67.5^\circ \leq a_s < 112.5^\circ$  and  $225^\circ \leq a_p < 315^\circ$ , (4)  $112.5^\circ \leq a_s < 157.5^\circ$  and  $270^\circ \leq a_p < 360^\circ$ , (5)  $157.5^\circ \leq a_s < 202.5^\circ$  and  $0^\circ \leq a_p < 45^\circ$ , (6)  $202.5^\circ \leq a_s < 247.5^\circ$  and  $0^\circ \leq a_p < 90^\circ$ , (7)  $247.5^\circ \leq a_s < 292.5^\circ$  and  $45^\circ \leq a_p < 135^\circ$ , (8)  $292.5^\circ \leq a_s < 337.5^\circ$  and  $90^\circ \leq a_p < 180^\circ$ , and that means the angle direction of the strip  $ps$  is opposite to that of the strip  $ss$ .

Based on the above algorithm, a pair of broken points can be determined, and then all interpolated points between the selected "end" points can be made. For example, the "end" points, ( $ip, jp$ ) and ( $is1, js1$ ) Fig. 7(2), are selected as broken points, and the image skeletons in Fig. 6 can be reconstructed and shown in Fig. 8(1) based on the above algorithm.

It is possible that there are some small gaps in the constructed skeletons. Therefore, it is necessary to find all gaps in which the distance between a pair of points is less than 6, and then fill all gaps. After the above procedure, the processing result of Fig. 8(1) is shown in Fig. 8(2).

The image in Fig. 8(2) are skeletonized and shown in Fig. 8(3). Based on the prior knowledge of neural cell image, there is one "junction" point at least in a cell image skeleton. Therefore, if there is no any "junction" point in an image skeleton, then the image skeleton is spurious, and it is deleted. Based on the algorithm, the image in Fig. 8(3) is further processed, and the extracted neuron skeleton is shown in Fig. 8(4).

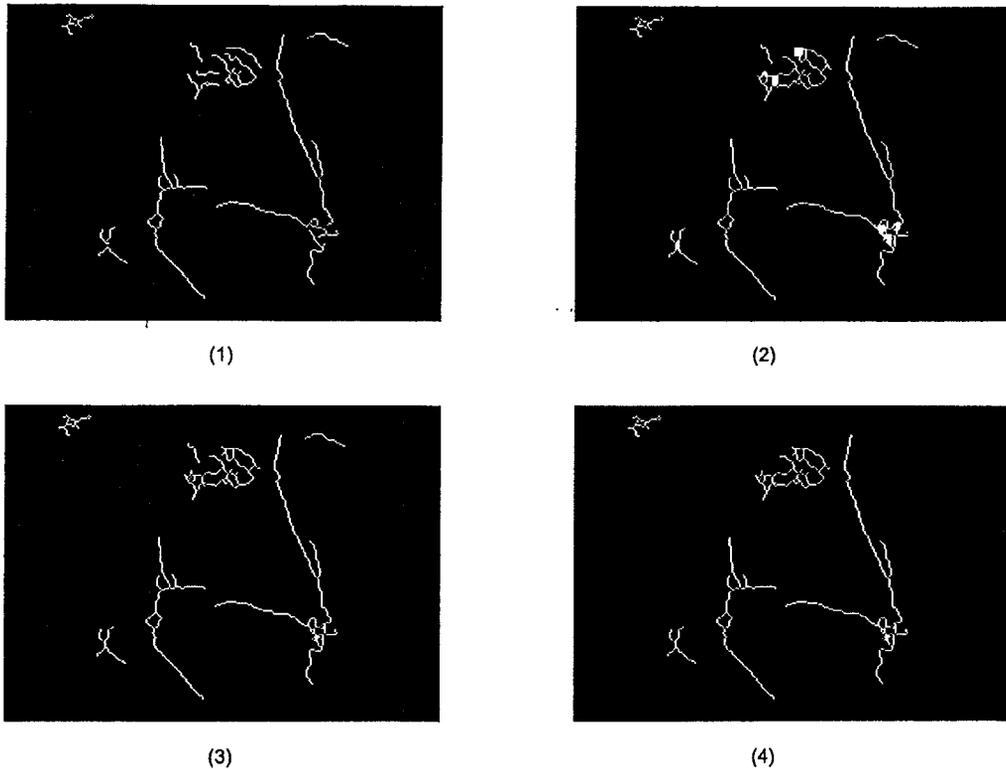


Fig. 8. The skeleton reconstruction, small gap connection, the reconstructed skeleton and extracted skeletons. (1) The skeleton reconstruction. (2) Small gap connection. (3) The reconstructed skeleton. (4) Extracted neuron skeletons (after removing spurious skeletons).



Fig. 9. All holes of the image in Fig. 8(4) are filled, and new skeletons are extracted. (1) The image in which all holes are filled. (2) Extracted skeletons of the image in Fig. 9(1).

#### 4.2. Analysis and calculation of reconstructed neuron skeletons

Each region skeleton consists of some branches (lines), and the neuron axon skeleton should be a branch. In order to find the length and morphology

of neuron skeleton axons, the reconstructed skeletons are processed furthermore. Firstly, all holes of the reconstructed skeletons are filled based a filling algorithm [18]. For example, all holes of the extracted skeletons in Fig. 8(4) are filled shown in Fig. 9(1). Furthermore, the skeletons of the image in Fig. 9(1) are

Table 1  
The data sets (chain codes and  $x$  and  $y$  coordinates) of the first branch (line) of skeleton 5.

|              |     |     |     |     |     |     |     |     |     |     |     |     |
|--------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Point number | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  | 11  | 12  |
| x            | 139 | 140 | 141 | 142 | 143 | 144 | 145 | 146 | 147 | 148 | 149 | 150 |
| y            | 135 | 134 | 134 | 134 | 133 | 133 | 133 | 132 | 132 | 132 | 132 | 132 |
| code         | 1   | 1   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
| Point number | 13  | 14  | 15  | 16  | 17  | 18  | 19  | 20  | 21  | 22  | 23  | 24  |
| x            | 151 | 152 | 153 | 154 | 155 | 156 | 157 | 158 | 159 | 160 | 161 | 162 |
| y            | 132 | 132 | 133 | 133 | 133 | 133 | 133 | 133 | 133 | 134 | 134 | 134 |
| code         | 0   | 0   | 7   | 0   | 0   | 0   | 0   | 0   | 0   | 7   | 0   | 0   |
| Point number | 25  | 26  | 27  | 28  | 29  | 30  | 31  | 32  | 33  | 34  | 35  | 36  |
| x            | 163 | 164 | 165 | 166 | 167 | 168 | 169 | 170 | 171 | 172 | 173 | 174 |
| y            | 135 | 136 | 136 | 136 | 137 | 137 | 137 | 138 | 138 | 138 | 138 | 138 |
| code         | 7   | 7   | 0   | 0   | 7   | 0   | 0   | 7   | 0   | 0   | 0   | 0   |
| Point number | 37  | 38  | 39  | 40  | 41  | 42  | 43  | 44  | 45  | 46  | 47  | 48  |
| x            | 175 | 176 | 177 | 178 | 179 | 180 | 181 | 182 | 183 | 184 | 185 | 186 |
| y            | 138 | 139 | 140 | 140 | 140 | 140 | 140 | 140 | 140 | 140 | 140 | 140 |
| code         | 0   | 7   | 7   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
| Point number | 49  | 50  | 51  | 52  | 53  | 54  | 55  | 56  | 57  | 58  | 59  | 60  |
| x            | 187 | 188 | 189 | 190 | 191 | 192 | 193 | 194 | 195 | 196 | 197 | 197 |
| y            | 141 | 142 | 142 | 142 | 143 | 144 | 145 | 146 | 147 | 147 | 148 | 148 |
| code         | 7   | 7   | 0   | 0   | 7   | 7   | 7   | 7   | 7   | 0   | 7   | 0   |
| Point number | 61  | 62  | 63  | 64  | 65  | 66  | 67  | 68  | 69  | 70  | 71  | 72  |
| x            | 199 | 200 | 201 | 202 |     |     |     |     |     |     |     |     |
| y            | 148 | 148 | 147 | 148 |     |     |     |     |     |     |     |     |
| code         | 0   | 0   | 1   | 7   |     |     |     |     |     |     |     |     |

extracted and shown in Fig. 9(2). The skeleton of the image need be smoothed by the algorithm described in Section 3.1. The smoothed skeletons are shown in Fig. 10 for the image in Fig. 9(2). Also, all “end” and “junction” points need be extracted for the smoothed skeletons. One branch of skeleton is defined as a line between a “end” point and its neighboring “junction” point. The chain code set of branch  $k$  of a skeleton is represented as:

$$C_k = \{c_0, c_1 \dots c_i, \dots c_{n-1}, c_n\} \quad (7)$$

where  $i$  is the index of the line point. The difference code,  $d_i$ , is defined as:

$$d_i = c_{i+1} - c_i. \quad (8)$$

If we calculate the difference code of two neighboring points between one “end” point and its neighboring “junction” point of smoothed skeleton, then  $|d_i|$  equals 0 or 1. If there is no “junction” point on the smoothed skeleton,  $|d_i|$  between two “end” points of skeleton equals 0 or 1. Therefore, the branch can be linearized, and the structural points of the branch can be extracted based on the above property [17].

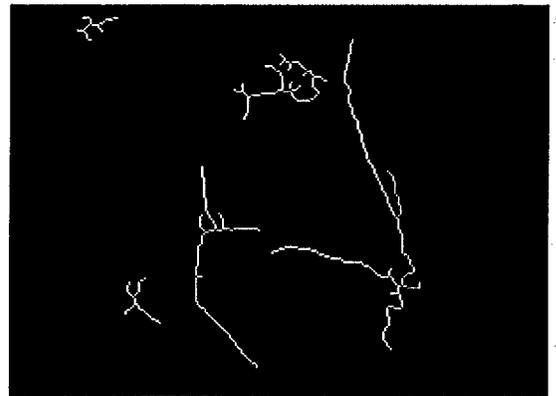


Fig. 10. The smoothing result of the skeletons in Fig. 9(2).

#### 4.3. Calculation and analysis of neuron skeletons

For the smoothed skeletons, the line and chain codes of skeleton branches can be found based on the following algorithm of neighboring points. The algorithm is as follows.

- (1) The  $x$  and  $y$  coordinates of first point of the line is those of a “end” point of one region. Search

Table 2  
The data sets (chain codes and  $x$  and  $y$  coordinates) of the second branch (line) of skeleton 5.

|              |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|--------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Point number | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  | 11  | 12  |     |     |
| x            | 181 | 181 | 180 | 180 | 180 | 180 | 180 | 180 | 179 | 179 | 179 | 179 |     |     |
| y            | 20  | 21  | 22  | 23  | 24  | 25  | 26  | 27  | 28  | 29  | 30  | 31  |     |     |
| code         | 6   | 6   | 5   | 6   | 6   | 6   | 6   | 6   | 5   | 6   | 6   | 6   |     |     |
| Point number | 13  | 14  | 15  | 16  | 17  | 18  | 19  | 20  | 21  | 22  | 23  | 24  |     |     |
| x            | 179 | 178 | 178 | 178 | 178 | 177 | 177 | 177 | 177 | 177 | 177 | 177 |     |     |
| y            | 32  | 33  | 34  | 35  | 36  | 37  | 38  | 39  | 40  | 41  | 42  | 43  |     |     |
| code         | 6   | 5   | 6   | 6   | 6   | 5   | 6   | 6   | 6   | 6   | 6   | 6   |     |     |
| Point number | 25  | 26  | 27  | 28  | 29  | 30  | 31  | 32  | 33  | 34  | 35  | 36  |     |     |
| x            | 178 | 179 | 179 | 180 | 180 | 180 | 179 | 178 | 179 | 179 | 179 | 180 |     |     |
| y            | 44  | 45  | 46  | 47  | 48  | 49  | 50  | 51  | 52  | 53  | 54  | 55  |     |     |
| code         | 7   | 7   | 6   | 7   | 6   | 6   | 5   | 5   | 6   | 6   | 6   | 7   |     |     |
| Point number | 37  | 38  | 39  | 40  | 41  | 42  | 43  | 44  | 45  | 46  | 47  | 48  |     |     |
| x            | 180 | 181 | 181 | 181 | 181 | 182 | 182 | 183 | 184 | 184 | 184 | 184 |     |     |
| y            | 56  | 57  | 58  | 59  | 60  | 61  | 62  | 63  | 64  | 65  | 66  | 67  |     |     |
| code         | 6   | 7   | 6   | 7   | 6   | 6   | 6   | 6   | 7   | 6   | 6   | 7   |     |     |
| Point number | 49  | 50  | 51  | 52  | 53  | 54  | 55  | 56  | 57  | 58  | 59  | 60  |     |     |
| x            | 184 | 185 | 185 | 186 | 186 | 186 | 186 | 186 | 187 | 187 | 187 | 188 |     |     |
| y            | 68  | 69  | 70  | 71  | 72  | 73  | 74  | 75  | 76  | 77  | 78  | 79  |     |     |
| code         | 6   | 7   | 6   | 7   | 6   | 6   | 6   | 6   | 7   | 6   | 6   | 7   |     |     |
| Point Number | 61  | 62  | 63  | 64  | 65  | 66  | 67  | 68  | 69  | 70  | 71  | 72  |     |     |
| x            | 188 | 189 | 189 | 189 | 189 | 190 | 190 | 191 | 192 | 192 | 193 | 193 |     |     |
| y            | 80  | 81  | 82  | 83  | 84  | 85  | 86  | 87  | 88  | 89  | 90  | 91  |     |     |
| code         | 6   | 7   | 6   | 6   | 6   | 7   | 6   | 7   | 7   | 6   | 7   | 6   |     |     |
| Point Number | 73  | 74  | 75  | 76  | 77  | 78  | 79  | 80  | 81  | 82  | 83  | 84  |     |     |
| x            | 193 | 193 | 194 | 194 | 195 | 196 | 196 | 197 | 197 | 197 | 198 | 199 |     |     |
| y            | 92  | 93  | 94  | 95  | 96  | 97  | 98  | 99  | 100 | 101 | 102 | 103 |     |     |
| code         | 6   | 7   | 6   | 7   | 6   | 7   | 6   | 7   | 6   | 6   | 7   | 7   |     |     |
| Point Number | 85  | 86  | 87  | 88  | 89  | 90  | 91  | 92  | 93  | 94  | 95  | 96  | 97  | 98  |
| x            | 199 | 200 | 200 | 200 | 200 | 201 | 201 | 202 | 202 | 203 | 203 | 204 | 205 | 206 |
| y            | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 | 113 | 114 | 115 | 116 | 117 |
| code         | 6   | 7   | 6   | 6   | 6   | 7   | 6   | 7   | 6   | 7   | 6   | 7   | 7   | 7   |

its chain code. Here, the chain code is eight-neighboring.

- (2) Find the second point of the line based on the neighboring search of the first point of the line.
- (3) Find other points of the line beginning from the second point's neighboring searching. If current point of the line is a "junction" point of the region, then the line following is finish. If the found point is the previous point of the line, then the found point is not new point of the line.

For example, there are five skeleton regions in the image shown in Fig. 10. There are 5, 4, 6, 11 and 7 "end" points in regions 1, 2, 3, 4 and 5 respectively. Therefore, the total number of lines is  $5 + 4 + 6 + 11 + 7 = 33$ . In the skeleton 5, there are 7 lines (branches), and the data sets (chain codes and  $x$  and  $y$  coordinates) of the first and second branches of skeleton 5 are shown in Tables 1 and 2 respectively. The length of the second line of the skeleton 5 is longest for neuron skeleton 5, and it is 98 points. The length of the first

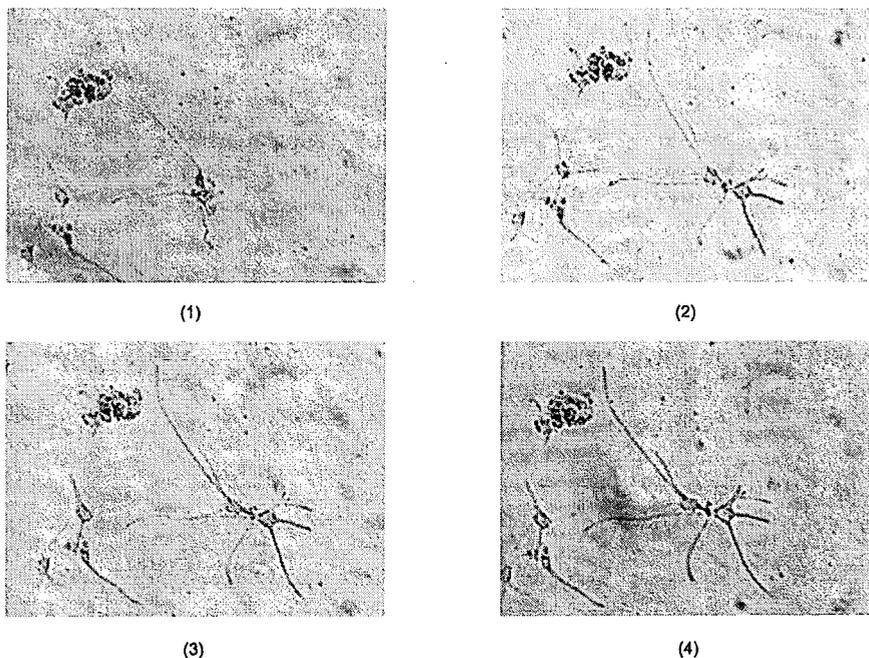


Fig. 11. A typical series of cell images. (1)  $T = 120$  min. (2)  $T = 60$  min. (3)  $T = 30$  min. (4)  $T = 15$  min.

line of the skeleton 5 is 64 points. The lengths of other five lines (lines 3, 4, 5, 6 and 7) are 27, 5, 7, 39, 14 points respectively. Also, the chain code set of the first line of skeleton 5 consists of codes 7, 0 and 1, the number of code 0 is largest, and the number of code 7 is larger than that of code 1. Therefore, its morphology of the first line begin to extend from "end" point to its corresponding "junction" point in the direction of codes 7, 0 and 1 (from the left to the right-down). For the second line, the chain code set of the first line of skeleton 5 consists of codes 6, 5 and 7, the number of code 6 is largest, and the number of code 7 is larger than that of code 5. Therefore, its morphology of the second line begin to extend from "end" point to its corresponding "junction" point in the direction of codes 6, 5 and 7 (from the upper to the right-down). Similarly, the morphology of other lines can be described.

Therefore, it is possible that the length of neural skeleton branch (axon) can be analyzed and calculated based on the above algorithms and prior knowledge of cultured neuron cells.

## 5. Experiments and conclusion

An efficient and new method has been developed for segmentation and reconstruction of cultured neu-

ron skeleton. The segmentation algorithm of neural cell images based on logical thresholding of grey and distance difference analysis and filtering window have been introduced.

The reconstruction of cultured neuron skeleton is based on the prior knowledge of neuron skeleton and the extension analysis (extension direction, distance and the morphological relation between the neighbor neuron skeletons).

Furthermore, the reconstructed skeletons are processed by filling holes, smoothing and extracting new skeletons. The final constructed neuron skeletons are analyzed and calculated to find the length and morphology of skeleton automatically. It is possible that tracing analysis of neuron skeleton of screening neuron images can be done automatically.

The developed algorithm have been applied to a typical series of images obtained from monitoring a live, unstained, neuron in culture. To effect morphological changes, and specifically the retraction of neuron cells, the cultured neurons were treated with  $350 \mu\text{M}$  of the oxidant  $\text{H}_2\text{O}_2$  over a period of 3 hours ( $t = 15, 30, 60, 120$  and  $180$  mins). It is apparent from the raw data (Fig. 11) that the captured images suffer from poor backgrounds. Experiment results show the efficient of our algorithms. The segmentation results of cultured neuron images in Fig. 11 are shown in

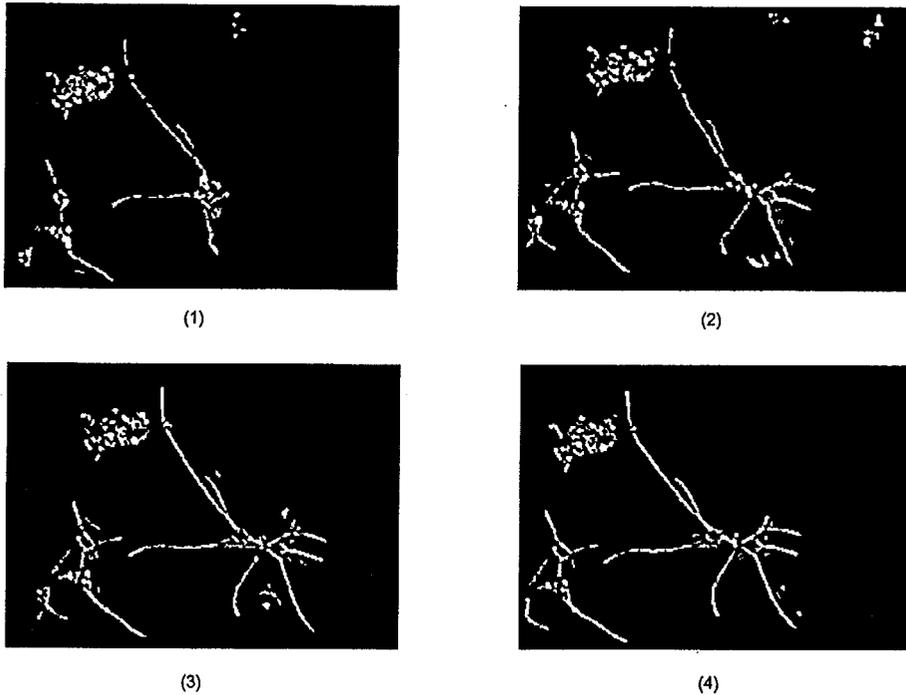


Fig. 12. The segmentation results of the typical series of cell images. (1)  $T = 120$  min. (2)  $T = 60$  min. (3)  $T = 30$  min. (4)  $T = 15$  min.

Fig. 12 based on grey and distance difference analysis, and filter-window. The reconstruction results of neuron skeletons are shown in Fig. 13 based on reconstruction algorithms described in Section 3. Furthermore, all holes of the reconstructed skeletons are filled and shown in Fig. 14, and then their skeletons are extracted and shown in Fig. 15. The skeletons of the images in Fig. 15 are smoothed by the algorithm described in Section 3.1. The smoothed skeletons are shown in Fig. 16. The final reconstructed neuron skeletons can be analyzed, and all data sets of lines (branches) can be calculated automatically based on our methods.

In order to validate the proposed algorithm, we can compare our method with some previous methods [1–3,7–11]. If the previous methods are used, the neural images have to have better background [1–3]. If the segmented neural images are poor, the reconstruction, analysis and calculation of neural image cannot be done.

For example, if three neuron images in Fig. 17 are segmented based on some previous methods [7,9,11], the segmented results are shown in Figs 19, 20 and 21 respectively. It is clear that these segmented neuron skeletons in Figs 19, 20 and 21 cannot be reconstructed by some previous neuron tracing methods [4–6].

However, the segmented neuron skeletons in Fig. 18(2, 3) can be reconstructed and analyzed by our developed algorithms of construction and analysis shown in Figs 15(1) and 10. Therefore, our algorithm is more robust than that by some previous methods.

There is limitation for any method. The proposed algorithm cannot work well for process the neuron image with very poor background which is difficult to be segmented. For example, the neuron image in Fig. 17(1) contains special shadow to make some branches form two parts. After using the proposed method, the reconstruction and analysis result of this neuron skeleton are shown in Fig. 22(6). Another neuron image with very poor background is shown Fig. 22(1), it is segmented by two previous methods [7,11] and our method, and the segmented results are shown in Figs 22(2, 3) and 22(4) respectively. It is clear that neuron skeletons in Fig. 22(2, 3) cannot be reconstructed by any previous methods. Also, the neuron skeletons in Fig. 22(4) are reconstructed by our method shown in Fig. 22(5). It is clear the reconstruction and analysis result are failed as there are some spurious neuron skeleton branches in the processed result, and the proposed method works not well in this case. However, our method is more robust than that of some previous methods based on the above comparison.

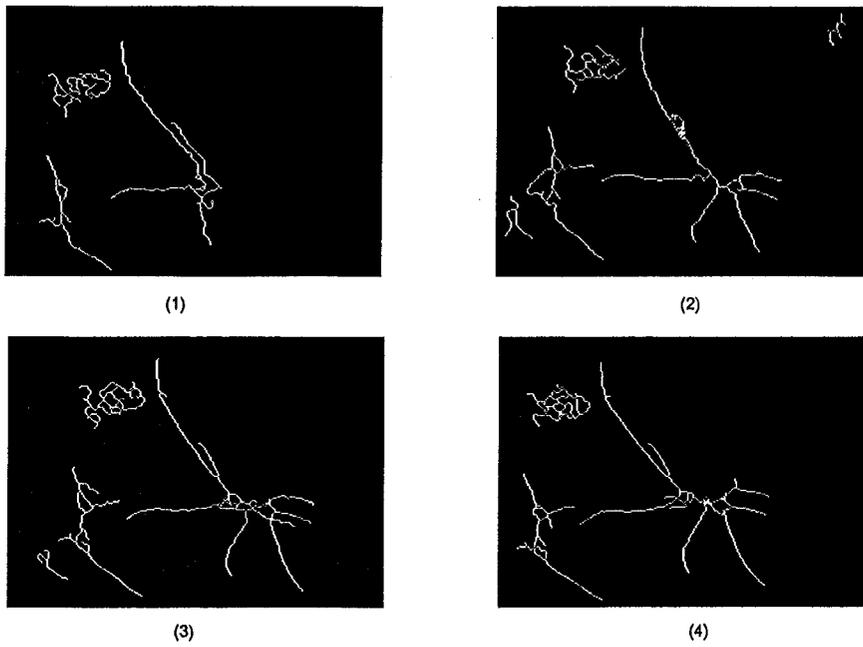


Fig. 13. The separation and skeleton reconstruction results of typical series of cell images (Fig. 12).

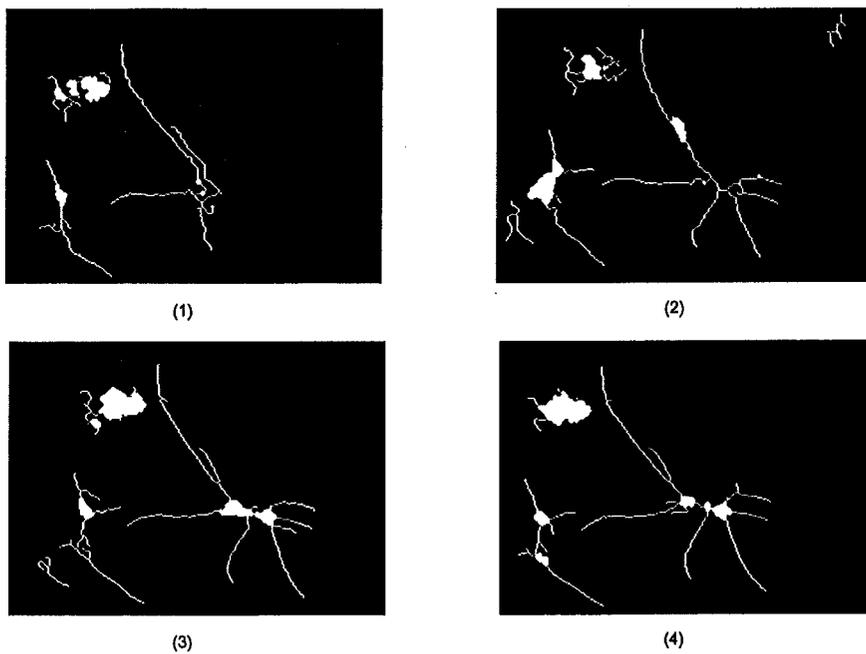


Fig. 14. The processing results of skeleton images (Fig. 13) by filling holes.

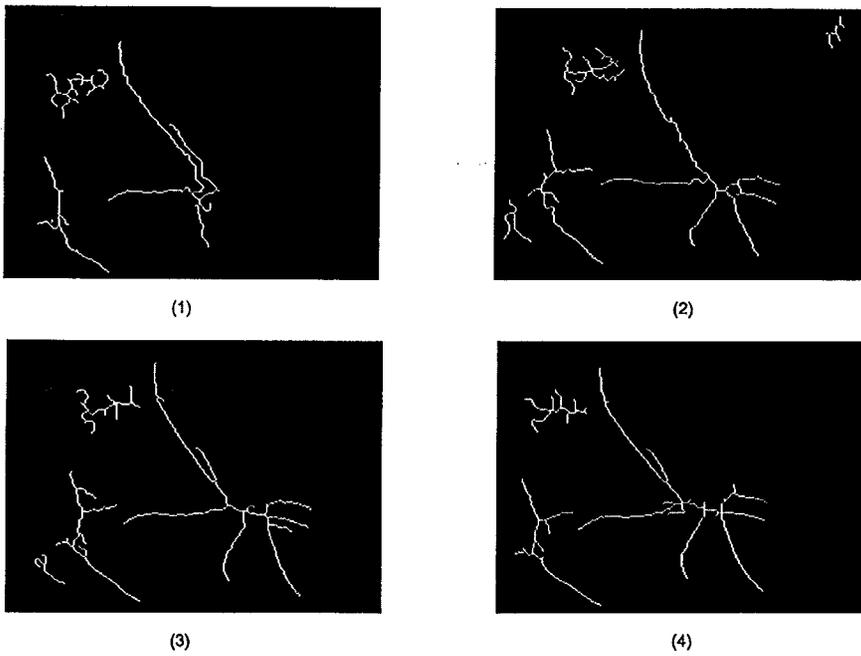


Fig. 15. The extracted skeleton images with filling holes (Fig. 14).

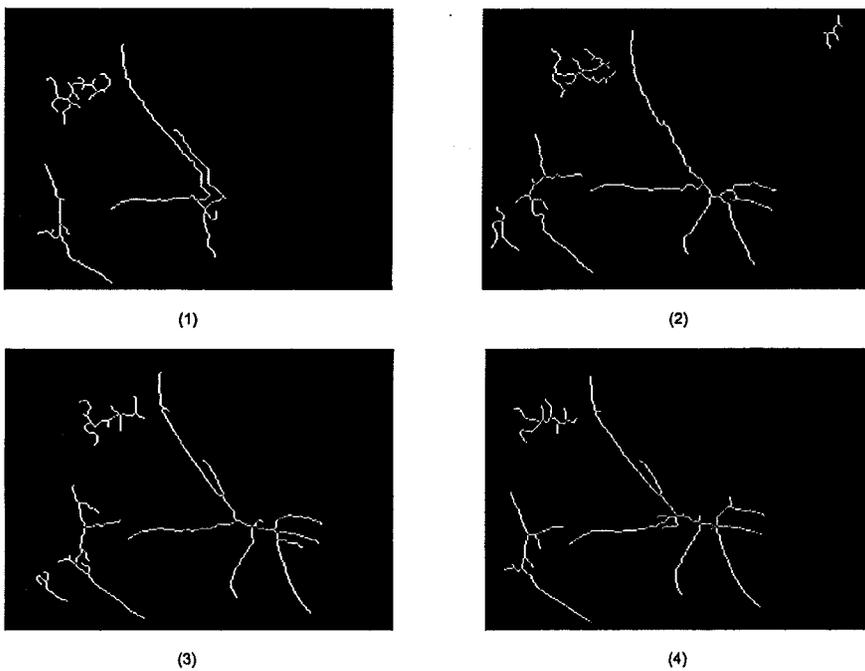


Fig. 16. The smoothing skeleton images of extracted skeleton images (Fig. 15).

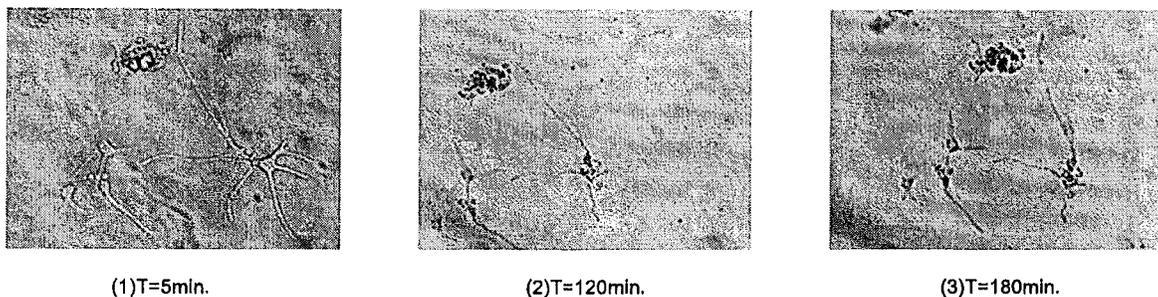


Fig. 17. Three original neural frame images.

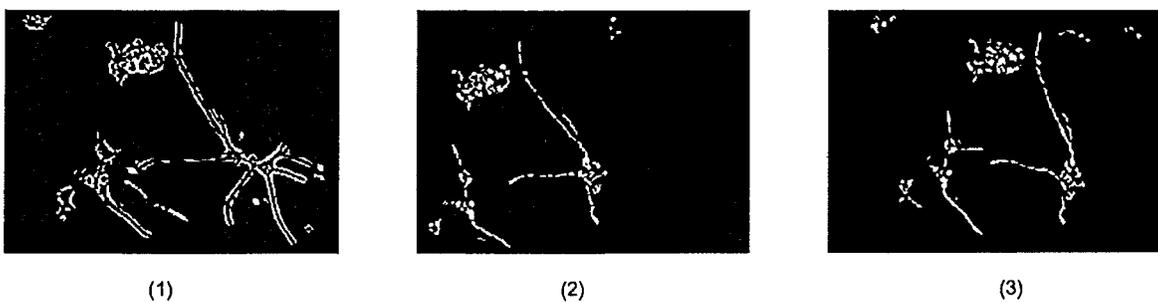


Fig. 18. The processed results of three frame images (see Fig. 17) using the proposed binary method and a filtering window with two objects.

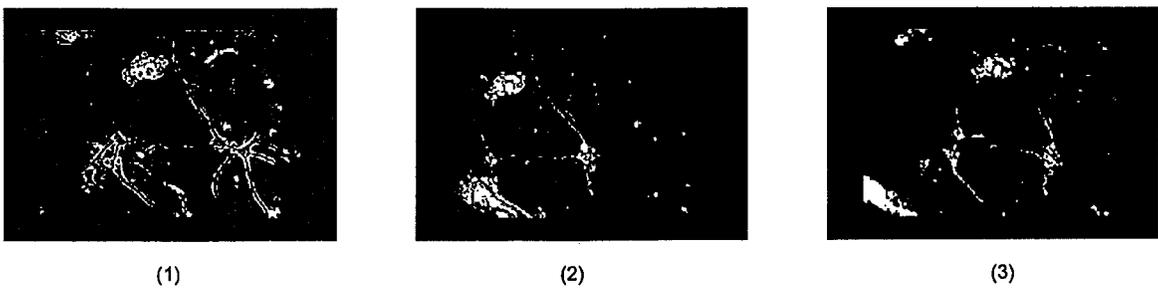


Fig. 19. Segmentation of images from Fig. 17 using Otsu's method.

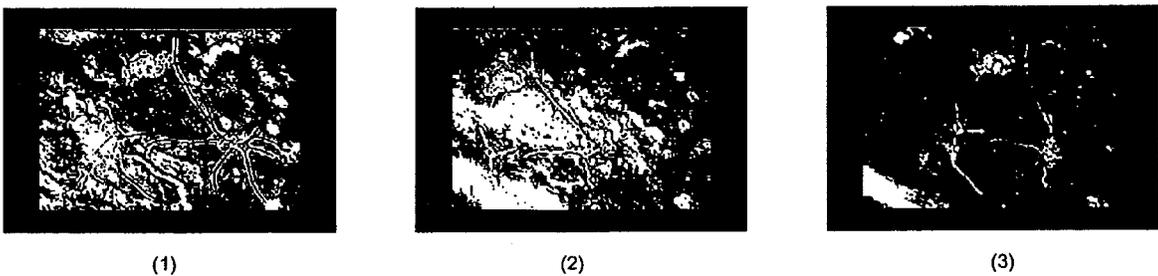


Fig. 20. Segmentation of images from Fig. 17 using the FCM method.

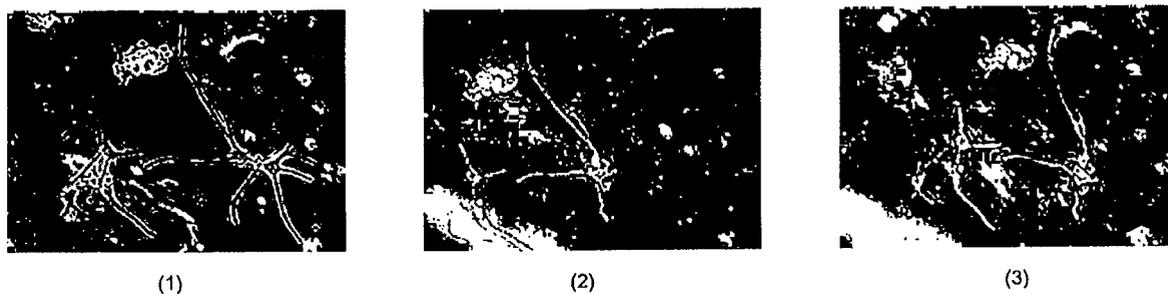


Fig. 21. Segmentation of images from Fig. 17 using unimodal thresholding.

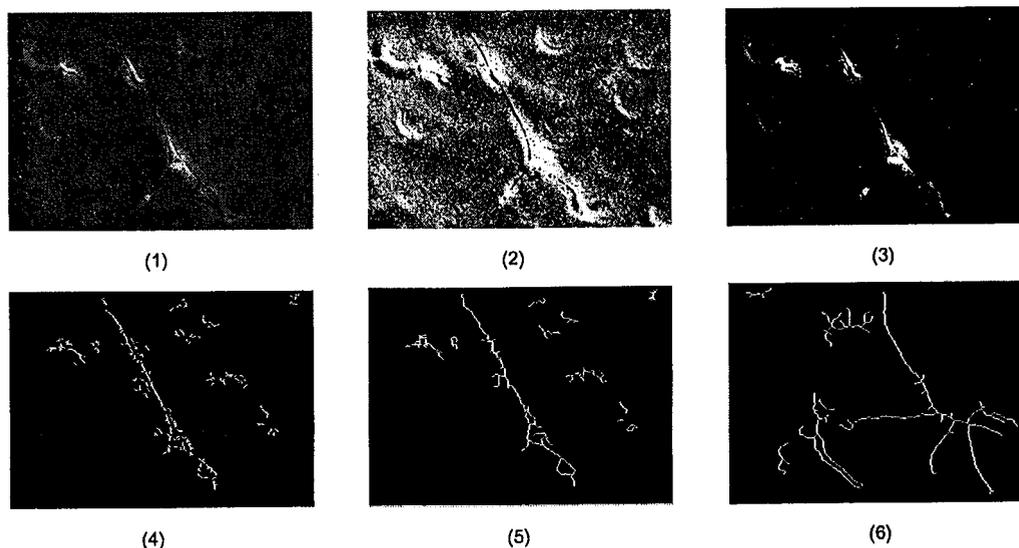


Fig. 22. The failed examples of neuron reconstruction and analysis. (1) An original image in a neuron database. (2) Segmentation result of preprocessed neuron image (Fig. 22(1)) using Otsu's method. (3) Segmentation result of preprocessed neuron image (Fig. 22(1)) using unimodal thresholding method. (4) Segmentation result of preprocessed neuron image (Fig. 22(1)) using the developed method. (5) Reconstruction result of preprocessed neuron image (Fig. 22(4)) using the proposed reconstruction method. (6) The reconstruction result of the segmented image (Fig. 18(1)) using the proposed reconstruction method.

## Acknowledgement

This work is supported by the Australia Research Council ARC-DP grant (DP0665598) to Tuan Pham.

## References

- [1] P.K. Allani, T. Sum, S.G. Bhansali, S.K. Mukherjee and M. Sonce, A comparative study of the effect of oxidative stress on the cytoskeleton in human cortical neurons, *Toxicol Appl Pharmacol* 196 (2004), 29–36.
- [2] J. Andersen, Oxidative stress in neurodegeneration: cause or consequence, *Nat Med* 10 (2004), 18–25.
- [3] B. Ischiropoulos, H. Ischiropoulos and J.S. Beckman, Oxidative stress and nitration in neurodegeneration: Cause, effect, or association, *J Clin Invest* 111 (2003), 163–169.
- [4] M. Smit, J. Leng and J.R. Klemke, Assay for neurite outgrowth quantification, *Biotechniques* 35(2) (2003), 254–256.
- [5] G. Xiong, X. Zhou, A. Degterev, L. Ji and S.T.C. Wong, Automated neurite labeling and analysis in fluorescence microscopy images, *Cytometry A* 69A(6) (2006), 494–505.
- [6] E. Meijering, M. Jacob, J.-C.F. Sarria, P. Steiner, H. Hirling, and M. Unser, Neurite Tracing in Fluorescence Microscopy Images using Ridge Filtering and Graph Searching: Principles and Validation, *Proceedings of the 2004 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, R. Leahy and C. Roux, eds, IEEE, Piscataway, NJ, 2004, pp. 1219–1222.
- [7] N. Ostu, A thresholding selection method from any level histogram, *IEEE Trans Systems Man Cybernet SMC-8* (1978), 62–66.
- [8] S.U. Lee, S.Y. Chung and R.H. Park, A comparative performance study of several global thresholding techniques for segmentation, *CVGIP* 52 (1990), 171–190.

- [9] Z. Chi, H. Yan and T. Pham, Fuzzy Algorithm: With Application to Image Processing and Pattern Recognition, *World Scientific Publishing Co.*, Singapore, 1996.
- [10] N. Papamarkos and B. Gatos, A new approach for multilevel threshold selection, *CVGIP: Graphical Models Image Process* 56 (1994), 357–370.
- [11] P.L. Rosin, Unimodal thresholding, *Pattern Recognition* 34 (2001), 2083–2096.
- [12] D. Yu, T.D. Pham, H. Yan, B. Zhang and D. Crane, Segmentation of cultured neurons using logical analysis of grey and distance difference, *Journal of Neuroscience Methods* 166 (2007), 125–137.
- [13] M. Kamel and A. Zhao, Extraction of binary character/ graphics images from grayscale document images, *CVGIP: Graphical Models Image Process* 55 (1993), 203–217.
- [14] Y. Yang and H. Yan, An adaptive logical method for binarization of degraded document images, *Pattern Recognition* 33(5) (2000), 787–807.
- [15] O.D. Trier and T. Taxt, Improvement of intergrated function algorithm' for binarization of document images, *Pattern Recognition Lett* 16 (1995), 277–283.
- [16] T.Y. Zhang and C.Y. Suen, A fast parallel algorithm for thinning digital patterns, *Communications of ACM* 27 (1984), 236–239.
- [17] D. Yu and H. Yan, An efficient algorithm for smoothing, linearization and detection of structure feature points of binary image contours, *Patt Recog* 30 (1997), 57–69.
- [18] P. Soille, *Morphological Image Analysis: Principles and Applications*, Springer-Verlag, 1999.