



ELSEVIER

Pattern Recognition Letters 18 (1997) 37–48

Pattern Recognition
Letters

A Hopfield neural network for adaptive image segmentation: An active surface paradigm ¹

Dinggang Shen ^{*}, Horace H.S. Ip

Image Computing Group, Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong

Received 10 April 1996; revised 12 November 1996

Abstract

This paper presents an adaptive thresholding technique for separating objects from noisy and non-uniformly illuminated images. The construction of the threshold surface is formulated as an active surface optimization problem, which is then solved by a Hopfield neural network. We proposed four constraints which ensure the active threshold surface to conform with the underlying image topography. Compared with Yanowitz and Bruckstein's method, this method produces superior segmentations particularly when the edge segments are sparsely distributed in the image and under non-uniform illuminations. Using three types of artificial and real images, we show that this method converges faster and produces better segmentations compared with previous interpolation-based adaptive thresholding techniques. © 1997 Elsevier science B.V.

Keywords: Image segmentation; Document segmentation; Adaptive thresholding; Active threshold surface; Hopfield neural network; Interpolation; Optimization problem

1. Introduction

Image segmentation is an important component of computer vision systems, whose goal is to partition a given image into meaningful regions and label each region by a region type. Also, it is a prerequisite for high-level image understanding and interpretation. Thresholding is one of the most popular techniques (Sahoo et al., 1988, Wu and Zhu, 1993), studied since 1962. Threshold selection is usually based on the information contained in the gray-level histogram of an image. However, it is difficult to select the thresholding level when the objects in an image appear lighter (darker) than the background in some regions but darker (lighter) in other regions possibly due to uneven illuminations. To overcome this problem, histogram transformation methods, which transform the gray-level histogram of the image into one with deeper valleys and sharper peaks has been applied to determine the threshold. Several variations of histogram transformation methods were unified by

^{*} Corresponding author. E-mail: cshen@cityu.edu.hk, cship@cityu.edu.hk.

¹ Electronic Annexes available. See <http://www.elsevier.nl/locate/patrec>.

using a gray-level versus edge-value scatter plot (Panda and Rosenfeld, 1978). It is clear that edge information can provide useful clues for selecting a good threshold.

However, a single fixed threshold, no matter how well chosen, cannot be used for the entire image in cases of uneven backgrounds and poor illumination. A method which tried with some success to solve this problem was developed by Chow and Kaneko (1972) and was further studied by Nakagawa and Rosenfeld (1979). The method uses local information on gray-level distributions to create an adaptive threshold surface. This adaptive thresholding method fails to give proper segmentation when the threshold surface is not in coherence with the image content. Attempting to overcome some of its disadvantages, Yanowitz and Bruckstein (1989) presented a method for constructing a threshold surface, based on the gradient magnitude image computed using edge operators, for example, the Canny edge detector (Jain and Dubuisson, 1992). The gray-level values on the edge points of the objects are chosen as local thresholds. Then the threshold surface is interpolated from these edge points over the entire image. There are several ways of interpolating a surface given a set of fixed points. Yanowitz and Bruckstein suggested that the Laplacian of the threshold surface should be zero and tried to obtain such a threshold surface using SORM (Successive Over-Relaxation Method). They started with a surface having a gray-level 0 everywhere except at those edge points that are fixed to their initial gray-level values during the iterative process. Since the set of edge points scattered in a manner which is unknown in advance and the neighborhood size used in SORM is chosen to be 3×3 , there exist many local minimal solutions. The process requires lengthy computations.

In this paper, we formulate the construction of the threshold surface as an active surface optimization problem, subjected to certain constraints, which can be solved by a Hopfield neural network. We make use of four constraint functions for the purpose. One constraint function, E_1 , guarantees that the Laplacian of the threshold surface be zero, however, we expand the neighborhood size to 5×5 and derive the kernel values for it. Another constraint function, E_2 , requires that the threshold surface should be smooth. More importantly, we propose here two other constraint functions, E_3 and E_4 , which ensure that the resulting threshold surface conforms to the topography of the image intensities for the removal of ghost objects, and that the construction process is speeded up. It is these latter constraint functions which give the resulting threshold surface distinguishing characteristics of an active surface and hence its superior performance in image segmentation compared with previous interpolation-based approaches.

The initial values at the neurons corresponding to edge points are set to their original gray-level values and will be kept fixed in the motion of the Hopfield neural network, while the values at the other neurons are determined by Niblack's (1986) method, based on the local mean and local standard deviation of the image. Once the initial state of the neurons has been set, the Hopfield neural network begins to work continuously until the energy function of the network stops decreasing. The image is segmented pixel by pixel based on the threshold surface just constructed. This approach is more efficient in terms of threshold surface construction time and is more likely to converge to the global minimum than SORM. One of the reasons why this approach is much faster is that the adaptive threshold surface is not initialized to 0 but to a reasonably good estimation through the use of Niblack's method.

In the literature, there exist many locally adaptive threshold methods. Eleven most promising locally adaptive threshold methods were evaluated by Trier and Jain (1995). With a post-processing step, Niblack's method gave the best performance. However, without post-processing step, Yanowitz and Bruckstein's method gave the best performance. Niblack's method was one of two fastest methods, while Yanowitz and Bruckstein's method was the slowest. Parker's method, faster than Yanowitz and Bruckstein's method but slower than Niblack's method, is one of these 11 locally adaptive threshold methods reviewed. In our experiments, four segmentation techniques, namely our method, Yanowitz and Bruckstein's method (SORM), Niblack's method and Parker's method, were applied to three types of artificial and real images with non-uniform illumination and noise. The results of these four methods were compared and our method gives the best performance.

The remainder of this paper is organized as follows. In Section 2, the general idea for adaptive image segmentation will be introduced, and Yanowitz and Bruckstein's method will be briefly described. Four error

(constraint) functions are derived in Section 3. At the same time, the connection weights of the Hopfield network are obtained. In Section 4, four segmentation methods, including our method, are compared by applying them to three types of artificial and real images. This paper concludes in Section 5.

2. Adaptive image segmentation

Let $M \times M$ be the size of the digitized image and $H = \{0, 1, \dots, 255\}$ be a set of gray-level values. G_{ij} and V_{ij} respectively express the gray-level value and the threshold on the pixel with coordinate (i, j) . Once the threshold surface has been constructed, the process of image segmentation actually becomes a mapping from G_{ij} to G'_{ij} , where $G'_{ij} \in H'$ and $H' = \{0, 1\}$. That is,

$$G'_{ij} = \begin{cases} 1, & G_{ij} \geq V_{ij}, \\ 0, & G_{ij} < V_{ij}. \end{cases}$$

Since pixels belonging to the objects or the background always have low gradient magnitude while pixels belonging to the boundaries always have high gradient magnitude, it is reasonable to regard the pixels with high gradient magnitude as edge pixels. The original gray-level values at these edge pixels are good candidates for local thresholds, and can be interpolated over the entire image to obtain the threshold surface.

The gradient magnitude image can be obtained by computing the Laplacian gradient of the image or using edge detectors, such as Sobel's edge operator and Canny's edge detector. Specifically, Canny's edge detector was the best choice for segmenting X-ray and C-scan images (Jain and Dubuisson, 1992). Those pixels with high gradient magnitude are determined by applying a single threshold to the gradient magnitude image and the resulting thresholded image is then thinned to yield a set of edge pixels.

There exist many ways to interpolate a surface, given a set of fixed points. One way is to set the discrete Laplacian of the threshold surface to zero. That is,

$$\text{LAP}(i, j) = \sum_{k=-1}^1 \sum_{l=-1}^1 L(k+1, l+1)V_{i+k, j+l} = 0, \quad (1)$$

where the kernel elements are from

$$L = \begin{bmatrix} c & b & c \\ b & a & b \\ c & b & c \end{bmatrix}.$$

By setting $a = -4$, $b = 1$ and $c = 0$, Yanowitz and Bruckstein used the following iteration equation, called SORM, for the interpolation of the threshold surface:

$$V_{ij}(t+1) = V_{ij}(t) + \frac{\omega \text{LAP}(i, j)}{4}, \quad \text{where } 1 < \omega < 2.$$

They suggested that the initial surface is set to gray-level 0 everywhere except at the edge points which are set to their original gray-values. However, since we do not know in advance how the edge points are scattered, and the edge segments may scatter sparsely over the entire image, it is difficult for this interpolation scheme to guide the threshold surface to converge to the desired position. Since the above updating rule is based on a neighbourhood size of 3×3 , this formulation gives rise to many local minimums. Fig. 6(b) shows a scan line of the locally stable threshold surface, interpolated by SORM, for Fig. 2(d).

In the next section, we will introduce a Hopfield neural network which not merely interpolates but constructs a threshold surface which conforms to the topography of the underlying image intensities. This approach provides a more robust and optimal solution for the threshold surface.

3. Hopfield neural network for active surface construction

3.1. Mathematical foundations of Hopfield network

Hopfield neural networks have been used successfully in solving optimization problems such as the traveling salesman problem (Hopfield and Tank, 1985). In recent years, Hopfield neural networks (Hung, 1992; Tenorio, 1987; Li et al., 1992) were applied to image segmentation and image matching. In this paper, we formulate the construction of the threshold surface as an optimization problem, subjected to four constraints, that can be solved by a Hopfield neural network.

The Hopfield neural network model is characterized by a network of neuron-like units with symmetric connections between units $T_{ij} = T_{ji}$, continuous values, a sigmoidal input–output transfer function, and an appropriate global energy function.

The motional equation of the neuron i is described as

$$C_i \frac{dU_i}{dt} = -\frac{U_i}{R_i} + \sum_{j=1}^N T_{ij} V_j + I_i, \quad V_i = g(U_i),$$

where $g(x)$ is a sigmoidal monotonic transfer function. In our application, we define $g(x)$ as $g(x) = x$ if $0 \leq x \leq 1$, otherwise $g(x) = 1$. U_i is the total input of the i th neuron and V_i is the output of the i th neuron. T_{ij} is the conductance between the output of the neuron j and the input of neuron i . C_i is the input capacitance of the i th neuron. I_i represents the external input current. N is the number of input synapses.

A suitable energy function for the Hopfield neural network is defined as follows:

$$E = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N T_{ij} V_i V_j - \sum_{i=1}^N V_i I_i + \sum_{i=1}^N \int_0^{V_i} g^{-1}(V) dV. \quad (2)$$

$dE/dt \leq 0$ when $C_i > 0$ and $T_{ij} = T_{ji}$. $dE/dt = 0$ if and only if $dV_i/dt = 0$ ($i = 1, 2, \dots, N$). It shows that the Hopfield network with symmetric connections ($T_{ij} = T_{ji}$) always converges to one of the stable minima of the energy function, since the evolution of the system is in the general direction of the negative gradient of the energy function. If the network energy function can be made equivalent to a certain objective (penalty) function that needs to be minimized, then the search for an energy minimum performed by the Hopfield network corresponds to the search for a solution of an optimization problem. Thus, if we devise penalty functions for the construction of the threshold surface, the Hopfield network can then be used to generate the threshold surface.

3.2. Error functions based on Laplacian and smoothness constraints

According to the requisite described in Eq. (1), the first error function E_1 can be defined as follows:

$$E_1 = \frac{1}{2} \sum_{(i,j)} [\text{LAP}(i, j)]^2 = \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^1 \sum_{l=-1}^1 L(k+1, l+1) V_{i+k, j+l} \right]^2.$$

Since the above error function involves a square term, we show in the following that it can in fact be rewritten as follows:

$$E_1 = \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-2}^2 \sum_{l=-2}^2 \text{Mask}(k+2, l+2) V_{i+k, j+l} V_{i, j} \right],$$

where Mask is a 5×5 symmetric kernel, as matrix L is a 3×3 symmetric matrix. Thus, Mask can be expressed in the following form:

$$\text{Mask} = \begin{bmatrix} W_5 & W_4 & W_2 & W_4 & W_5 \\ W_4 & W_3 & W_1 & W_3 & W_4 \\ W_2 & W_1 & W_0 & W_1 & W_2 \\ W_4 & W_3 & W_1 & W_3 & W_4 \\ W_5 & W_4 & W_2 & W_4 & W_5 \end{bmatrix}.$$

Our immediate goal is to determine the values for six kernel elements W_0, W_1, W_2, W_3, W_4 and W_5 . Since there exist six terms $W_0 \cdot V_{i,j}V_{i,j}, W_1 \cdot V_{i,j+1}V_{i,j}, W_2 \cdot V_{i,j+2}V_{i,j}, W_3 \cdot V_{i-1,j+1}V_{i,j}, W_4 \cdot V_{i-1,j+2}V_{i,j}$ and $W_5 \cdot V_{i-2,j+2}V_{i,j}$ in E_1 , those six kernel elements can thus be obtained through respectively analyzing the six coefficients for $V_{i,j}V_{i,j}, V_{i,j+1}V_{i,j}, V_{i,j+2}V_{i,j}, V_{i-1,j+1}V_{i,j}, V_{i-1,j+2}V_{i,j}$ and $V_{i-2,j+2}V_{i,j}$.

Without loss of generality, let us write

$$[\text{LAP}(i, j)]^2 = \begin{bmatrix} cV_{i-1,j-1} & +bV_{i-1,j} & +cV_{i-1,j+1} \\ +bV_{i,j-1} & +aV_{i,j} & +bV_{i,j+1} \\ cV_{i+1,j-1} & +bV_{i+1,j} & +cV_{i+1,j+1} \end{bmatrix}^2.$$

$[\text{LAP}(i, j)]^2$ is the local Laplacian error on pixel (i, j) . Pixel (i, j) is shown as a white circle in Fig. 1(a). Based on the expression of $[\text{LAP}(i, j)]^2$, we know there exists a term $a^2V_{i,j}V_{i,j}$ in $[\text{LAP}(i, j)]^2$. That is, the local Laplacian error $[\text{LAP}(i, j)]^2$ on pixel (i, j) contributes $a^2V_{i,j}V_{i,j}$. Similarly, four local Laplacian errors ($[\text{LAP}(i, j-1)]^2, [\text{LAP}(i, j+1)]^2, [\text{LAP}(i-1, j)]^2$ and $[\text{LAP}(i+1, j)]^2$) on the four pixels $((i, j-1), (i, j+1), (i-1, j)$ and $(i+1, j))$, whose positions are shown in the black circles in Fig. 1(a), contribute $b^2V_{i,j}V_{i,j}$ each, giving a total of $4b^2V_{i,j}V_{i,j}$. In addition, the other four local Laplacian errors ($[\text{LAP}(i-1, j-1)]^2, [\text{LAP}(i-1, j+1)]^2, [\text{LAP}(i+1, j-1)]^2$ and $[\text{LAP}(i+1, j+1)]^2$) on the four black pixels $((i-1, j-1), (i-1, j+1), (i+1, j-1), (i+1, j+1))$

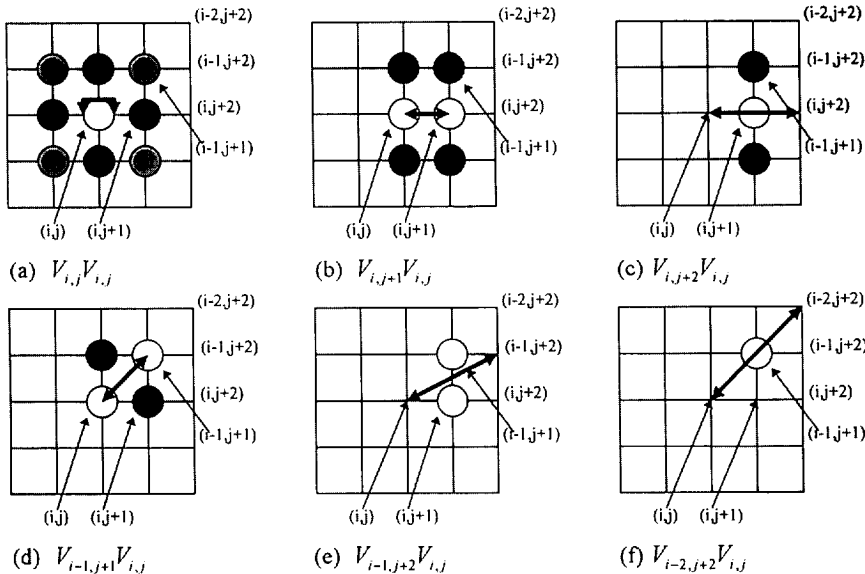


Fig. 1. Relationship between $V_{i,j}$ and its 5×5 neighboring pixels $V_{i+k,j+l}, -2 \leq k, l \leq 2$.

1), $(i-1, j+1)$, $(i+1, j-1)$ and $(i+1, j+1)$ contribute $c^2 V_{i,j} V_{i,j}$ each, giving a total of $4c^2 V_{i,j} V_{i,j}$. Consequently, the coefficient W_0 for $V_{i,j} V_{i,j}$ is equal to $(a^2 + 4b^2 + 4c^2)$.

From Fig. 1(b), we can see that both $V_{i,j}$ and $V_{i,j+1}$ are included in the 3×3 -connected neighborhoods of the six circled pixels. In $[\text{LAP}(i, j)]^2$, there exists a term $2abV_{i,j+1}V_{i,j}$. Considering the symmetric property of E_1 , we only take $abV_{i,j+1}V_{i,j}$ as the contribution of $[\text{LAP}(i, j)]^2$. Thus, two local Laplacian errors ($[\text{LAP}(i, j)]^2$ and $[\text{LAP}(i, j+1)]^2$) on the two white pixels offer a total of $2abV_{i,j+1}V_{i,j}$, and four local Laplacian errors ($[\text{LAP}(i-1, j)]^2$, $[\text{LAP}(i-1, j+1)]^2$, $[\text{LAP}(i+1, j)]^2$ and $[\text{LAP}(i+1, j+1)]^2$) on the four black pixels offer a total of $4bcV_{i,j+1}V_{i,j}$. The result is $W_1 = 2ab + 4bc$. According to Fig. 1(c), we see that both $V_{i,j}$ and $V_{i,j+2}$ are contained in the 3×3 -connected neighborhoods of the three circled pixels. One local Laplacian error $[\text{LAP}(i, j+1)]^2$ on the white pixel $(i, j+1)$ offers $b^2V_{i,j+2}V_{i,j}$, and the two local Laplacian errors ($[\text{LAP}(i-1, j+1)]^2$ and $[\text{LAP}(i+1, j+1)]^2$) on the two black pixels together offer $2c^2V_{i,j+2}V_{i,j}$. The result is $W_2 = 2c^2 + b^2$. Fig. 1(d) denotes that both $V_{i,j}$ and $V_{i-1,j+1}$ are contained in the 3×3 -connected neighborhoods of the four circled pixels. Thus the two local Laplacian errors ($[\text{LAP}(i, j)]^2$ and $[\text{LAP}(i-1, j+1)]^2$) on the two white pixels together contribute $2acV_{i-1,j+1}V_{i,j}$, and the other two local Laplacian errors ($[\text{LAP}(i-1, j)]^2$ and $[\text{LAP}(i, j+1)]^2$) on the two black pixels offer $2b^2V_{i-1,j+1}V_{i,j}$. The result is $W_3 = 2ac + 2b^2$. Fig. 1(e) also indicates that both $V_{i,j}$ and $V_{i-1,j+2}$ are contained in the 3×3 -connected neighborhoods of the two white pixels. Since the two Laplacian errors ($[\text{LAP}(i-1, j+1)]^2$ and $[\text{LAP}(i, j+1)]^2$) on these two white pixels together contribute $2bcV_{i-1,j+2}V_{i,j}$, we get $W_4 = 2bc$. Finally, in Fig. 1(f), both $V_{i,j}$ and $V_{i-2,j+2}$ are contained in the 3×3 -connected neighborhood of only one white pixel. $[\text{LAP}(i-1, j+1)]^2$ on this white pixel offers $c^2V_{i-2,j+2}V_{i,j}$, which leads to $W_5 = c^2$.

Based on the analysis described above, we get $W_0 = a^2 + 4b^2 + 4c^2$, $W_1 = 2ab + 4bc$, $W_2 = 2c^2 + b^2$, $W_3 = 2ac + 2b^2$, $W_4 = 2bc$, $W_5 = c^2$. Setting $a = -4$, $b = 1$, $c = 0$ (Eq. (1)), we have

$$\text{Mask} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & -8 & 2 & 0 \\ 1 & -8 & 20 & -8 & 1 \\ 0 & 2 & -8 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

To make the threshold surface smooth, the second error function E_2 is defined as follows:

$$E_2 = \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^1 \sum_{l=-1}^1 (V_{i+k,j+l} - V_{i,j})^2 \right] = \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^1 \sum_{l=-1}^1 S(k+1, l+1) V_{i+k,j+l} V_{i,j} \right],$$

where

$$S = \begin{bmatrix} -2 & -2 & -2 \\ -2 & 16 & -2 \\ -2 & -2 & -2 \end{bmatrix}.$$

The first and the second error functions can be combined to give

$$E_{12} = E_1 + \alpha E_2 = \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-2}^2 \sum_{l=-2}^2 M_{-} S(k+2, l+2) V_{i+k,j+l} V_{i,j} \right],$$

where

$$M_{-} S = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 2-2\alpha & -8-2\alpha & 2-2\alpha & 0 \\ 1 & -8-2\alpha & 20+16\alpha & -8-2\alpha & 1 \\ 0 & 2-2\alpha & -8-2\alpha & 2-2\alpha & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix},$$

and α is a coefficient.

3.3. Error functions based on similarity between the topography of the threshold surface and the image intensities: an active threshold surface

Since we do not know in advance how the edge points are scattered and the edge segments may scatter sparsely over the entire image, we exploit information on the image intensities to guide the threshold surface to converge to the desired position. To achieve this, we suggest here that the threshold surface should be locally parallel to the image intensities, i.e., the local topography of the threshold surface and that of the underlying image intensities should be similar, and that the difference between the local threshold and the gray-level value on the corresponding pixels should be small. Based on these ideas, we derive the constraints E_3 and E_4 as follows. Owing to the constraint E_3 , our approach eliminates the post-processing step needed in Yanowitz and Bruckstein's method for the removal of ghost objects. Owing to the constraint E_4 which obtains the guiding information from the image intensities, our approach is more efficient in terms of speed of convergence.

The third error function E_3 expresses the requirement about local parallelism. In the following expression, $f_{i,j}$ is the normalized form of gray-level value $G_{i,j}$. The range of $f_{i,j}$ is from 0 to 1.

$$\begin{aligned}
 E_3 &= \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} \left((V_{i+k,j+l} - V_{i,j}) - (f_{i+k,j+l} - f_{i,j}) \right)^2 \right] \\
 &= \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} \left((V_{i+k,j+l} - V_{i,j})^2 - 2(V_{i+k,j+l} - V_{i,j})(f_{i+k,j+l} - f_{i,j}) + (f_{i+k,j+l} - f_{i,j})^2 \right) \right] \\
 &= \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (V_{i+k,j+l} - V_{i,j})^2 \right] \\
 &\quad + \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (-2)(V_{i+k,j+l} - V_{i,j})(f_{i+k,j+l} - f_{i,j}) \right] \\
 &\quad + \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (f_{i+k,j+l} - f_{i,j})^2 \right].
 \end{aligned}$$

The first term of E_3 is equal to E_2 , thus it can be simplified as done for E_2 . Since the last term of E_3 is fixed during the construction process of the threshold surface, it is a constant. The second term can be rewritten in the following form:

$$\begin{aligned}
 \text{Term}_2 &= \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (-2)(V_{i+k,j+l} - V_{i,j})(f_{i+k,j+l} - f_{i,j}) \right] \\
 &= \sum_{(i,j)} \left[V_{i,j} \sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (f_{i+k,j+l} - f_{i,j}) \right] + \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} V_{i+k,j+l} (f_{i,j} - f_{i+k,j+l}) \right] \\
 &= 2 \sum_{(i,j)} \left[V_{i,j} \sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (f_{i+k,j+l} - f_{i,j}) \right] \\
 &= \sum_{(i,j)} \left[V_{i,j} \sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (-S(k+1, l+1)) f_{i+k,j+l} \right],
 \end{aligned}$$

where S is the same matrix as the one defined in E_2 . The third error function can thus be rewritten as follows:

$$E_3 = \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} S(k+1, l+1) V_{i+k, j+l} V_{i,j} \right] \\ + \sum_{(i,j)} \left[V_{i,j} \sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (-S(k+1, l+1)) f_{i+k, j+l} \right] + \text{const}_1.$$

The fourth error function E_4 is based on the difference between the local threshold value and the gray-level value on corresponding pixels.

$$E_4 = \frac{1}{2} \sum_{(i,j)} (V_{ij} - f_{ij})^2 = \frac{1}{2} \sum_{(i,j)} V_{ij} V_{ij} + \sum_{(i,j)} V_{ij} (-f_{ij}) + \frac{1}{2} \sum_{(i,j)} f_{ij} f_{ij} \\ = \frac{1}{2} \sum_{(i,j)} V_{ij} V_{ij} + \sum_{(i,j)} V_{ij} (-f_{ij}) + \text{const}_2.$$

Combining the third and the fourth error functions, we get

$$E_{34} = \beta E_3 + \gamma E_4 = \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} P(k+1, l+1) V_{i+k, j+l} V_{i,j} \right] \\ + \sum_{(i,j)} \left[V_{i,j} \sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (-P(k+1, l+1)) f_{i+k, j+l} \right] + \text{const},$$

where

$$P = \begin{bmatrix} -2\beta & -2\beta & -2\beta \\ -2\beta & 16\beta + \gamma & -2\beta \\ -2\beta & -2\beta & -2\beta \end{bmatrix}, \quad \text{const} = \beta \times \text{const}_1 + \gamma \times \text{const}_2.$$

3.4. Extracting the connection weights for the Hopfield network

The total error function for the construction of the threshold surface is therefore

$$E = E_{12} + E_{34} \\ = \frac{1}{2} \sum_{(i,j)} \left[\sum_{k=-2}^{k=2} \sum_{l=-2}^{l=2} \text{Msp}(k+2, l+2) V_{i+k, j+l} V_{i,j} \right] \\ + \sum_{(i,j)} \left[V_{i,j} \sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} (-P(k+1, l+1)) f_{i+k, j+l} \right] + \text{const}, \quad (3)$$

where

$$\text{Msp} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 2 - 2(\alpha + \beta) & -8 - 2(\alpha + \beta) & 2 - 2(\alpha + \beta) & 0 \\ 1 & -8 - 2(\alpha + \beta) & 20 + 16(\alpha + \beta) + \gamma & -8 - 2(\alpha + \beta) & 1 \\ 0 & 2 - 2(\alpha + \beta) & -8 - 2(\alpha + \beta) & 2 - 2(\alpha + \beta) & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

Comparing the corresponding coefficients between the energy function (2) and the error function (3) results in the following weights for the Hopfield network:

$$T_{ij,(i+k)(j+l)} = -M\text{sp}(k+2, l+2), \quad -2 \leq k, l \leq 2,$$

$$I_{ij} = \sum_{k=-1}^{k=1} \sum_{l=-1}^{l=1} P(k+1, l+1) f_{i+k, j+l}.$$

The resulting network is a special case of the general Hopfield network and the motion equation of the neuron ij can be obtained as follows:

$$C_{ij} \frac{dU_{ij}}{dt} = -\frac{U_{ij}}{R_{ij}} - \sum_{k=-2}^2 \sum_{l=-2}^2 M\text{sp}(k+2, l+2) V_{i+k, j+l} + \sum_{k=-1}^1 \sum_{l=-1}^1 P(k+1, l+1) f_{i+k, j+l},$$

$$V_{ij} = g(U_{ij}).$$

When the continuous Hopfield network is simulated by a computer, the following iteration equation is operated on every neuron except on the neurons representing the edge points of the image:

$$U_{ij}(t+1) = U_{ij}(t) + \eta \frac{dU_{ij}(t)}{dt},$$

where $\eta = 1/20$, i.e. a normalization factor for Mask.

4. The active threshold surface algorithm and experimental results

Our Hopfield Net Active Threshold Surface algorithm can be stated as follows.

- (1) Derive the gradient magnitude image from the smoothed original image, based on the Canny edge detector.
- (2) Determine a single threshold for segmenting the gradient magnitude image, obtain the edge points, and then get a thinned image of edge points. This single threshold is calculated by the mean and standard deviation of the gradient magnitude image. That is $\text{Th} = m + 1.9s$, where Th is the threshold value, m and s are the mean and standard deviation values.
- (3) Apply the Hopfield neural network described in Section 3 to construct the active threshold surface. The initial threshold surface is calculated by Niblack's method.
- (4) Obtain the binary image based on the threshold surface.

In our experiments, four segmentation techniques, i.e. our method, Yanowitz and Bruckstein's method (SORM), Niblack's method and Parker's method, are applied to three types of artificial and real images with non-uniform illumination and noise. These image are all of size 256×256 pixels. Fig. 2(a) is a real tool image,

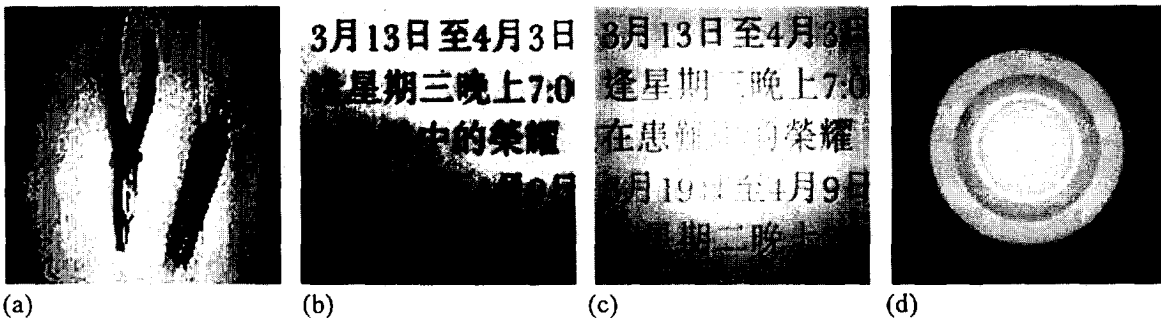


Fig. 2. The test images to be segmented. (a) Tool image. (b) Chinese characters. (c) Spotted characters. (d) Artificial image.

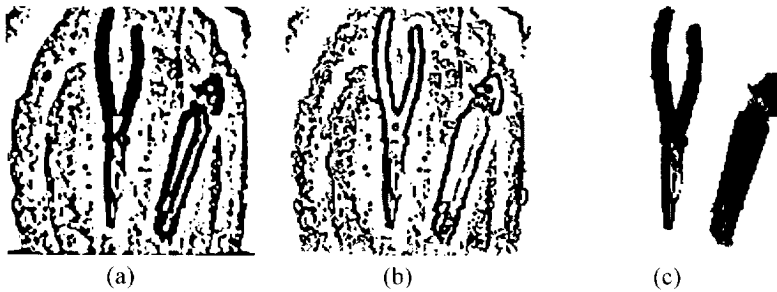


Fig. 3. Segmentation results of Fig. 2(a). (a) Segmentation by Niblack's method. (b) Segmentation by Parker's method. (c) Segmentation by our method.

which was captured by a CCD camera under conditions of non-uniform illumination. Fig. 2(b) is also a real image containing Chinese characters, which was obtained using a scanner, while Fig. 2(c) is formed by adding a Gaussian spot to the image of Fig. 2(b). Fig. 2(d) is an artificial image, which is used to test which method converges to the global minimum.

The segmentation results for Figs. 2(a), 2(b) and 2(c) by Parker's method, Niblack's method and our method are presented in Figs. 3, 4 and 5, respectively. It is easy to see that the images segmented by our method contain only the valid objects and none of the background noise. For the images containing Chinese characters, the characters are clearly segmented despite the highly unevenly illuminated background. The segmented results due to Parker's and Niblack's methods, Figs. 4(a) and 4(b), however, contain both false-positive and false-negative pixels. Since the edge points are evenly and densely distributed in the images, Yanowitz and Bruckstein's

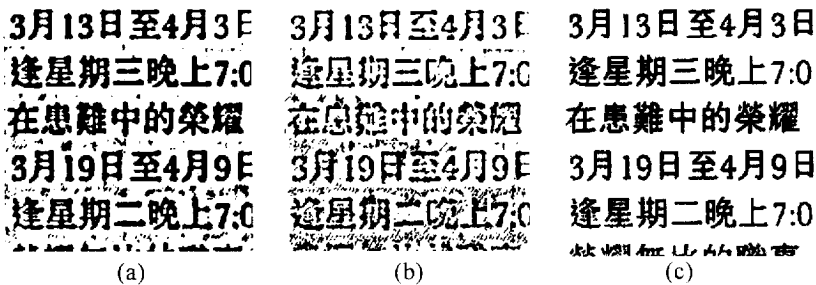


Fig. 4. Segmentation results of Fig. 2(b). (a) Segmentation by Niblack's method. (b) Segmentation by Parker's method. (c) Segmentation by our method.

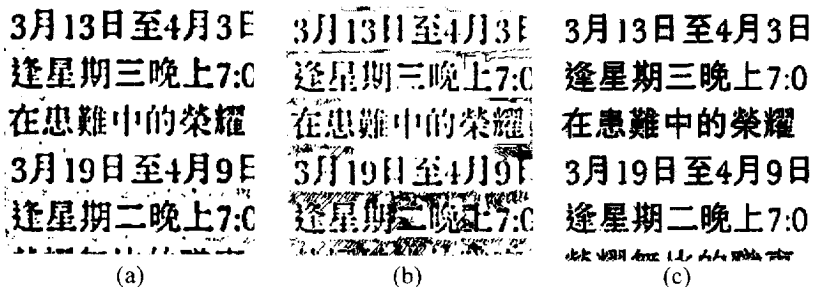


Fig. 5. Segmentation results of Fig. 2(c). (a) Segmentation by Niblack's method. (b) Segmentation by Parker's method. (c) Segmentation by our method.

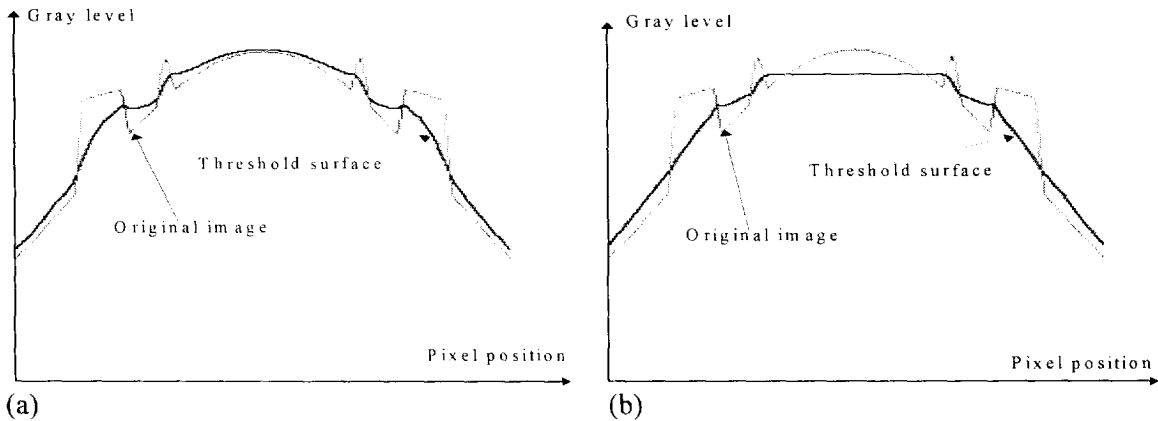


Fig. 6. Cross-section through picture in Fig. 2(d) and the interpolated threshold surface. (a) Interpolated by our method. (b) Interpolated by SORM.

method produced similar segmentation results as our method. When applied to Figs. 2(b) and 2(c), our method took about 93 seconds to segment each image on a Pentium-166 personal computer, while Yanowitz and Bruckstein's method took about 182 seconds. The segmentation speed of our method is about 2 times faster than that of Yanowitz and Bruckstein's method. This outcome is partly attributed to the design of the constraints E_3 and E_4 , and the method of obtaining a reasonably good estimation for the initial threshold surface by using Niblack's method.

Fig. 2(d), however, clearly demonstrates the difference between the SORM method by Yanowitz and Bruckstein and the one reported here since the edge segments in the image are sparsely distributed. Fig. 6 shows a cross-section through the threshold surfaces constructed by the two methods. Figs. 6(a) and 6(b) correspond to the result of our method and that of Yanowitz and Bruckstein's method, respectively, and the shape of the two surfaces are clearly different. The threshold surface due to Yanowitz and Bruckstein's method does not take account of the underlying profile of the image intensities and consequently it generates two rings and a central circle as the result of the segmentation (Fig. 7(b)) which is visually incorrect – the central circle is in fact an artifact of the gradual increase in intensities towards the centre of the image. This result corresponds to a local minimum of the SORM iterative function. The shape of our active threshold surface approximates that of the underlying image intensities, particularly towards the centre of the image, and produces an adaptive threshold surface which gives a visually correct result containing only the two brighter rings (Fig. 7(a)). This threshold

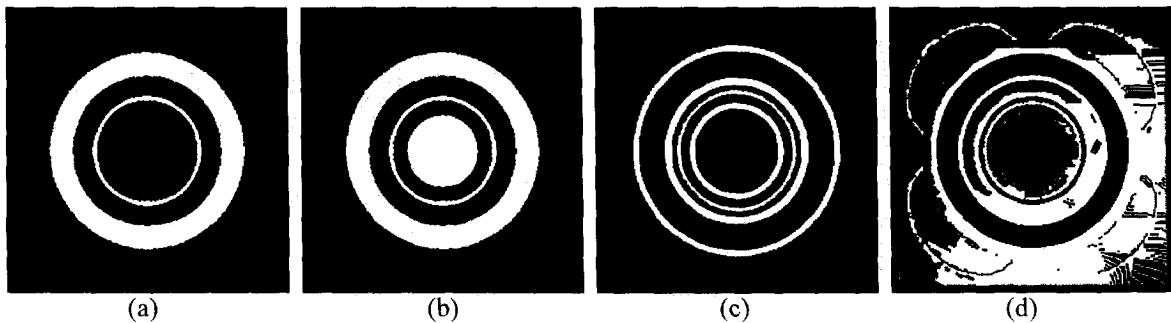


Fig. 7. Segmentation results of Fig. 2(d). (a) Segmentation by our method. (b) Segmentation by SORM. (c) Segmentation by Niblack's method. (d) Segmentation by Parker's method.

surface corresponds to a global minimum of the error function, i.e., Eq. (3). Since the edge segments are sparsely distributed in this image (Fig. 2(d)), Yanowitz and Bruckstein's method took more than 37 minutes to converge to a local minimum, while our method took less than 6 minutes. The running speed is about 6 times faster. This example clearly demonstrates the significant contribution of the additional constraints E_3 and E_4 , compared with Yanowitz and Bruckstein's method. Fig. 7 compares the segmented results of the four methods. It can be seen that our method correctly segments the artificial image, while segmentation results from the other methods are not satisfactory and contain additional artifacts.

5. Conclusion

We have formulated the construction of the threshold surface as an active surface optimization problem, and solved this optimization problem using a Hopfield neural network. In our formulation, we derived penalty functions based on four different constraints which ensure that the resulting threshold surface conforms to the topography of the underlying image intensities. In addition, we derived the 5×5 kernel elements which enforce the discrete Laplacian of the threshold surface to be zero. Since our method also uses edge points as the control points for the construction of the threshold surface, for images with densely scattered edge points, the speed of our method is about 2 times faster than Yanowitz and Bruckstein's method (SORM) and the results are similar for the two methods. But for images with sparsely scattered edge points or segments, our method produces better segmentation results and the running speed is about 6 times faster. This result is a direct consequence of the additional constraints E_3 and E_4 proposed here, and the method of obtaining a reasonably good estimation for the initial threshold surface using Niblack's method.

References

- Chow, C.K. and T. Kaneko (1972). Automatic boundary detection of the left-ventricle from cineangiograms. *Comput. Biomed. Res.* 5, 388–410.
- Hopfield, J.J. and D.W. Tank (1985). Neural computation of detection in optimization problems. *Biol. Cybernet.* 52, 141–152.
- Huang Chung-Lin (1992). Parallel image segmentation using modified Hopfield model. *Pattern Recognition Letters* 13 (5), 345–353.
- Jain, A.K. and M.P. Dubuisson (1992). Segmentation of X-ray and C-scan images of fiber reinforced composite materials. *Pattern Recognition* 25 (3), 257–270.
- Li Qian, Chen Di and Wu Jiankang (1992). Image matching using neural network. *Chinese J. Pattern Recognition Artificial Intelligence* 5 (3), 221–227.
- Nakagawa, Y. and A. Rosenfeld (1979). Some experiments on variable thresholding. *Pattern Recognition* 11 (3), 191–204.
- Niblack, W. (1986). *An Introduction to Digital Image Processing*. Prentice-Hall, Englewood Cliffs, NJ, 115–116.
- Panda, D.P. and A. Rosenfeld (1978). Image segmentation by pixel classification in (gray level, edge value) space. *IEEE Trans. Comput.* 27, 875–879.
- Sahoo, P.K., S. Soltani and A.K.C. Wong (1988). A survey of thresholding techniques. *Comput. Vision Graphics Image Process.* 41, 233–260.
- Tenorio, M.P. (1987). Real time noisy image segmentation using an artificial neural network model. *IEEE First Internat. Conf. on Neural Networks*, San Diego, CA, 21–24 June, Vol. 4, 357–363.
- Trier, Ø.D. and A.K. Jain (1995). Goal-directed evaluation of binarization methods. *IEEE Trans. Pattern Anal. Machine Intell.* 17 (12), 1191–1201.
- Wu Yiquan and Zhaoda Zhu (1993). Survey of thresholding selection in image process from 1962 to 1992. *J. Data Acquisition Processing* 8 (3).
- Yanowitz, S.D. and A.M. Bruckstein (1989). A new method for image segmentation. *Comput. Vision Graphics Image Process.* 46, 82–95.