JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 10, 221-237 (1994)

Short Paper

Parallel Thinning by a Recurrent Neural Network*

REI-YAO WU† AND WEN-HSIANG TSAI‡
†*Department of Computer Science and Information Engineering*
‡*Department of Computer and Information Science*
National Chiao Tung University
Hsinchu, Taiwan 300, Republic of China

A 3-layer recurrent neural network is proposed for parallel thinning. This network iteratively removes the contour points of an object shape by template matching. After the neural network converges, a perfectly 8-connected skeleton is derived. The set of templates which the neural network tries to match is specially designed for a one-pass parallel thinning algorithm. The proposed neural network is shown both theoretically and experimentally to do the same work as the one-pass parallel thinning algorithm.

Keywords: Parallel thinning, Recurrent neural network, Template matching, Excessive erosion.

1. INTRODUCTION

Thinning is a process which eliminates a large volume of redundant pixels from an image and retains the important pixels without unduly disturbing the topology and connectivity characteristics of the original image. This is important for many image analysis and pattern recognition applications since useful features can be captured from the skeletons resulting from thinning. Many algorithms have been proposed for thinning in the literature. Some of them are sequential algorithms, and the others are parallel ones[1-7]. To preserve the connectivity of a line and prevent excessive eroding of a thin line, some parallel algorithms divide

Received May 31, 1993; revised November 23, 1993.
Communicated by Soo-Chang Pei
* This work was supported partially by National Science Council, Republic of China, under grant No. NSC81-0404-E009-017.
† Also with the World College of Journalism and Communications, Taipei, Taiwan, Republic of China.
‡ To whom all correspondence should be sent.

an iteration into multiple passes[1-3]. On the other hand, one-pass algorithms have also been proposed to save processing time[4-6]. In a single-processor system, sequential algorithms are suitable. Parallel algorithms can also be simulated in a single-processor system. To exert the strength of a parallel algorithm, a multi-processor system or special hardware, which consists of a large number of simple processing elements, may be designed to implement the algorithm. For example, Chin et al.[5] used simple logic gates to match thinning, restoring, and trimming templates.

Neural networks have been widely employed to solve decision and classification problems in various application areas. Because the neurons in a neural network function simultaneously, it is also a good idea to use neural networks to solve image processing problems in which different parts of the spatial data can usually be processed in parallel. For example, Cortes and Hertz[8] used directional second derivatives to detect the edges in an image by using a neural network. Works on thinning by neural networks have rarely appeared in the literature. An example is Matsumoto et al.[9], in which images are thinned by cellular neural networks [10,11] with 8 planes. Another example is Graf et al.[12,13], in which a VLSI architecture is designed to multiply a binary input vector with a stored matrix of weights to skeletonize binary images.

By considering the parallel processing capabilities of neural networks, a thinning algorithm which is suitable for implementation by neural networks should at least have the following two characteristics:

1) It should be a parallel thinning algorithm. For the sake of simplicity and processing speed of the neural network, a one-pass parallel thinning algorithm is preferred.
2) To force the neural network to converge, the algorithm should converge correctly (i.e., should stop iterations with a correct thinning result).

A one-pass parallel thinning algorithm called OPPTA [14], proposed by the authors previously and briefly reviewed in Section 2, does possess the above two characteristics. Based on this algorithm, a 3-layer recurrent neural network, which can be used to perform thinning of binary images, is proposed in this paper. This network iteratively removes the edge points of an object shape. In each iteration, the removal of a point is determined by the criterion of whether or not the neighbors of that point match any of a set of templates. Both template matching and point removal tasks are performed by the neural network. It is proved by lemmas and theorems that the network produces exactly the same thinning results as does the algorithm. Some experimental results are also shown to assure this fact.

The remainder of this paper is organized as follows. In Section 2, the one-pass parallel thinning algorithm OPPTA is described. Section 3 includes the description of the structure of the proposed neural network. The connection weight assignment and activation rules for correct thinning of an image are specified in Section 4. The lemmas and theorems proving the correctness of the neural network are also included. Section 5 gives the experimental results. And finally, concluding remarks are given in Section 6.

2. ONE-PASS PARALLEL THINNING ALGORITHM

A good thinning algorithm should have the following characteristics in general:

1) It should preserve the connectivity of an object shape.
2) It should prevent excessive erosion.
3) It should be noise insensitive.
4) It should produce skeletons topologically equivalent to original object shapes.

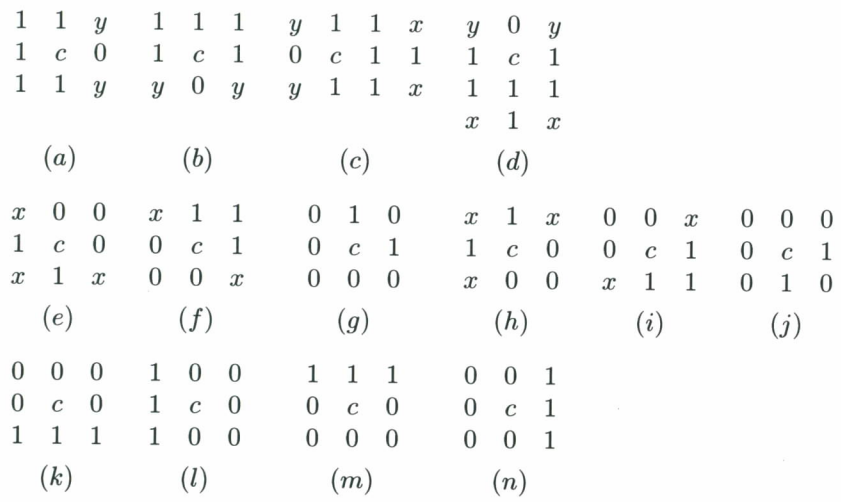


Fig. 1. Thinning templates.

Using the templates shown in Fig. 1, the one-pass parallel thinning algorithm OPPTA described below possesses the characteristics listed above. As shown in Fig. 1, there are twelve 3×3 templates (templates (a), (b), and (e) through (n), one 3×4 template (template (c)), and one 4×3 template(template (d)). The symbols 'c', '0', '1',and'x' used in these templates denote the currently tested pixel, a white pixel, a black pixel, and a don't-care condition, respectively. These symbols follow the conventional notations while the symbol 'y', also appearing in the templates, is a special one. It does not appear singly in a thinning template, and at least one of the pixels represented by the set of symbols 'y' should be a white pixel. By matching the templates shown in Fig. 1 with a given object shape, the thinning algorithm OPPTA iteratively eliminates the edge points of the object shape layer by layer in parallel. The details are as follows.

Algorithm: OPPTA.
Input: a binary image f^0.
Output: the image of the thinning result.
Step 1: $i := 0$.
Step 2: flag:=false.

Step 3: Check each pixel of f^i. If it is a black pixel and its neighbors match any of the templates (a) though (n), then change it to a white pixel and set flag:=true.

Step 4: If flag=false, which means the image is thinned, then go to Step 5 with f^i as the thinning result. Otherwise, $i := i + 1$ and go to Step 2 to perform the next iteration.

Step 5: Output the thinning result.

For convenience, several terms concerning the templates are defined here for further discussion. First, the set T of all the templates shown in Fig. 1 is called the *template set*. Each template consists of several elements. For example, a 3×3 template consists of 9 elements. Any of these elements is denoted by one of the symbols 'c', '0', '1', 'x', and 'y'. Since the corresponding positions of symbols 'c' and 'x' are less important in the proposed neural network, we exclude these two symbols and include the other three (i.e., symbols '0', '1' and 'y') in a *symbol set* S for the convenience for further discussion; i.e., we define $S = \{$'0', '1', 'y'$\}$.

Next, consider an individual template. Symbols '0' and '1' always exist in all of the templates while symbol 'y' exists only in four of the templates (templates (a) through (d)). The set of the symbols which exist in an individual template t is called the *symbol set* of t and is denoted as S_t.

Finally, consider a template as an area in units of pixels with each pixel represented by a symbol. Then, a *sub-template* of a template is defined as a connected sub-area or a combination of several connected sub-areas of the template area. This definition is quite general; however, only a limited set of sub-templates will be discussed in the remainder of this paper. Specifically, only sub-templates with all of their pixels represented by identical symbols in symbol set S are meaningful in the following discussion. For example, there exist three meaningful sub-templates of template (a). They are the sub-area covered by the pixels represented by symbol '1', the sub-area covered by the single pixel represented by symbol '0', and the combination of the sub-areas covered by the pixels represented by symbol 'y'. A sub-template of template t, whose pixels are all represented by a single symbol s, will be denoted as T_{ts} and will be called a *sub-template of* t *corresponding to* s.

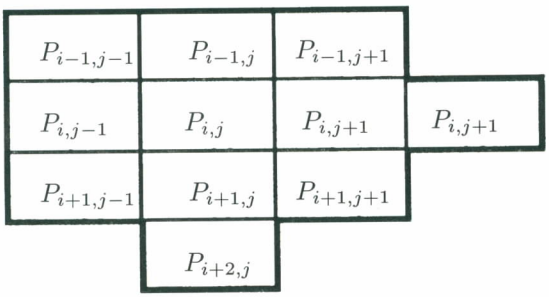


Fig. 2. 10-neighbors of pixel P_{ij}.

If all of the templates in template set T were of size 3×3, the area which should be included to determine whether a black pixel p should be removed or not

in the thinning process would be the 8-neighborhood of p. However, the addition of one 3×4 template and one 4×3 template (i.e., templates (c) and (d) in Fig. 1) increases the number of neighbors which should be considered from 8 to 14. Observing templates (c) and (d), we see that, except for the 8-neighbors in a 3×3 window, each template includes further only one important neighbor (denoted as '1') since the other two neighbors are don't-cares (denoted as 'x'). We can, thus, define a 10-neighborhood of each tested pixel. As shown in Fig. 2, any of the set of pixels appearing in the figure around the tested pixel P_{ij} is defined as a *10-neighbor* of P_{ij}. In addition to the 8-neighbor in a 3×3 window, the east-neighbor of the east-neighbor of $P_{ij}(i.e., P_{i,j+2})$ and the south-neighbor of the south-neighbor of P_{ij} (i.e., $P_{i+2,j}$) are also included. This reduces the number of concerned neighbors from 14 to 10. Note that by 10-neighbors we refer not only to pixels in images, but also to neurons in neuron planes, as we will see later.

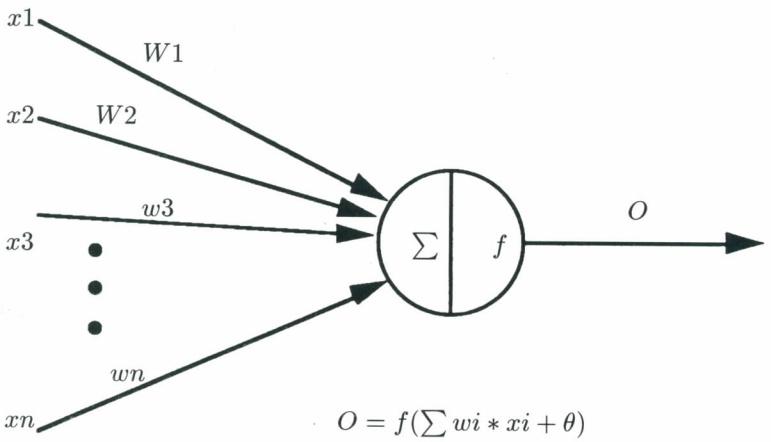


Fig. 3. Processing element of proposed neural network.

3. PROPOSED NEURAL NETWORK

As shown in Fig. 3, all of the neurons in the proposed network are simple processing elements, and each of them receives inputs from others. Through a set of connection strengths or weights, these inputs are summed up. The sum of these weighted inputs is called the net input. The output of a neuron is a nonlinear function of its net input. Each neuron is a binary system with an output value of either 0 or 1 with 0 and 1 representing inactive and active, respectively. That is, the output value, $f(x)$, of a neuron is

$$f(x) = \begin{cases} 1 & \text{if } x > \alpha; \\ 0 & \text{otherwise,} \end{cases}$$

where x is the net input and α is a threshold value.

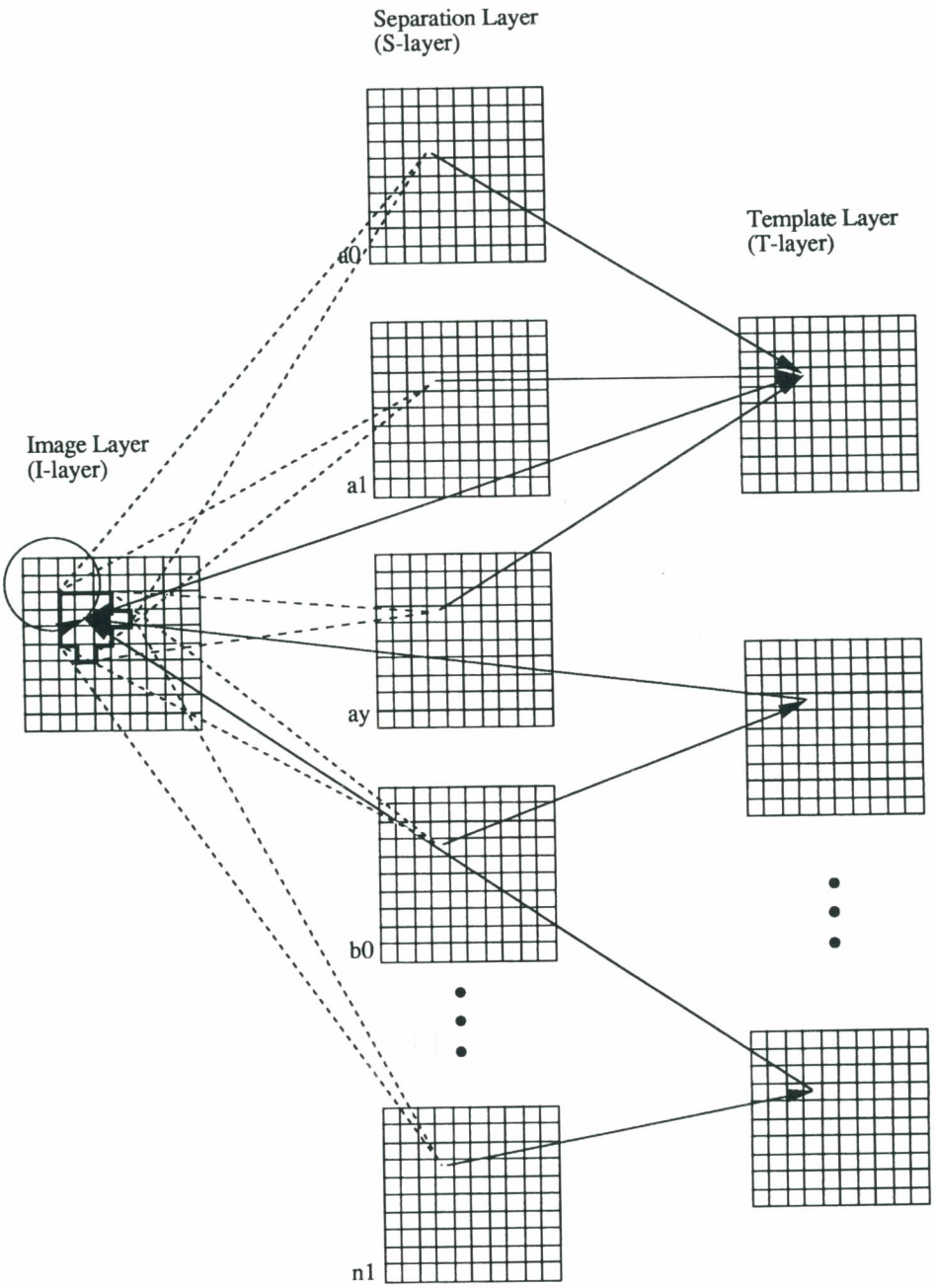


Fig. 4. Architecture of proposed neural network.

As shown in Fig. 4, the proposed neural network has three layers, namely, the image layer, the separation layer, and the template layer.

(1) Image layer (abbreviated as I-layer)

The I-layer is a single neuron plane, which represents the input image. It will also be called the image neuron plane in the following discussion. Each neuron in this plane corresponds one-to-one with a pixel in the input image. The definition of 10-neighborhood is also applicable to the neurons in this plane since the neurons in this plane have the same topological relationship as that of the pixels in the input image. Initially, the input image is fed into this plane by activating the neurons which correspond to the black pixels and leaving the other neurons inactive. After the network becomes stable, the image of the thinning result will appear in this plane with active neurons representing black pixels. When the network is performing the thinning work, this plane represents the temporary thinning result. Following a recurrent network scheme, the neurons in this plane receive inputs from the neurons in the T-layer (described below) and itself.

(2) Separation layer(abbreviated as S-layer)

There are several neuron planes in this layer, and each plane is used to match a sub-template of a template corresponding to a specific symbol in symbol set S. That is, the neurons in this layer will be active if the corresponding sub-template is matched. To collect necessary information, the receptive fields of the neurons in this layer are part of the 10-neighbors of the corresponding neurons in the I-layer, the exact elememts being dependent on the shape of the corresponding sub-template.

(3) Template layer(abbreviated as T-layer)

In this layer, there are as many neuron planes as the templates. Each plane is used to match its corresponding template. The neurons in this layer receive inputs from the S-layer. If two neurons in different neuron planes are in identical positions in their individual neuron planes, we say that they are at the same position. For each neuron X in the plane responsible for matching template t, its receptive field consists of all the neurons which are located at the same position as neuron X, in those planes in the S-layer each of which is expectd to match a sub-template of t corresponding to a symbol s in S_t. The output of a neuron in this layer feeds back to the I-layer to inactivate the corresponding neuron if it is active, which means that the corresponding template is matched.

Fig. 4 shows the network architecture in plane view. Each rectangle in the figure represents a neuron plane. Neurons in a neuron plane are represented by the meshes inside the rectangle. Rectangles of the left part, the middle part, and the right part are the I-layer, the S-layer, and the T-layer, respectively. The area inside the bold polygon in the I-layer is the 10-neighborhood of a neuron in the I-layer. Each pair of dashed lines coming from the I-layer to the S-layer indicates that a neuron P in the S-layer receives inputs from part of the 10-neighbors of the neuron in the I-layer, which is located at the same position as neuron P. The solid lines with arrows link neurons at the same positions in different layers. The circle with an arrow in the image plane represents that the neurons in this plane receive inputs from themselves in addition to those from the T-layer. Fig. 5 shows the network architecture in another view. The small solid rectangles in the figure represent neurons. Each line with an arrow links a neuron to another. In the I-layer, the 10-neighbors of a neuron are shown. On the other hand, only a representative neuron in each neuron plane of the S-layer and the T-layer is shown.

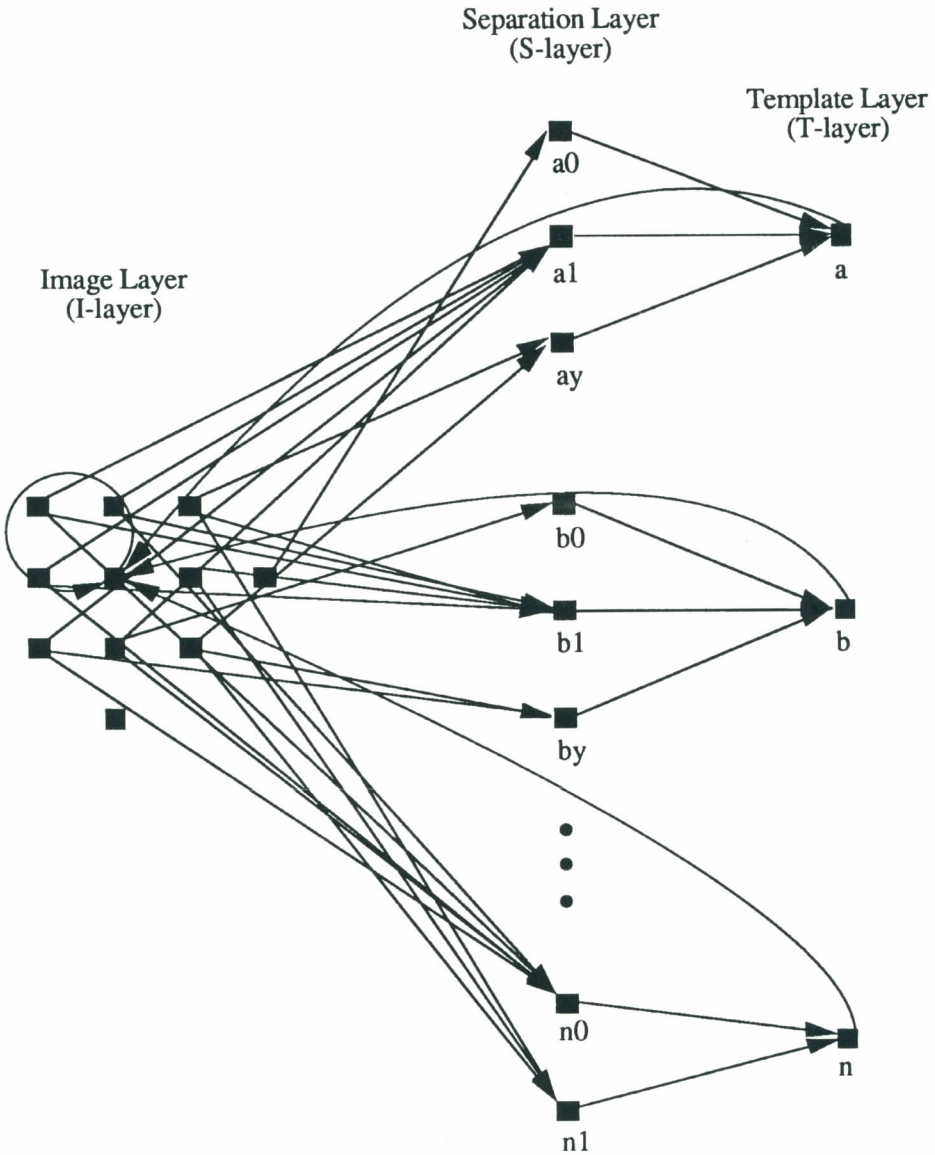


Fig. 5. Receptive fields of neurons in each layer.

To perform the work of thinning an image in dimension $N \times N$, $N \times N$ neurons should be included to construct the neuron plane of the I-layer. Let these neurons be denoted as P_{ij}, $0 \leq i < N$, $0 \leq j < N$. The number of neurons in the S-layer depends on the number of templates and the cardinality of the symbol set for each template. Let the neurons in this layer be denoted as S_{tsij}, where t and s index the corresponding sub-templates (just as in sub-template T_{ts}), and i and j index the corresponding neuron positions. The T-layer has as many neuron planes as

templates. Similarly, the neurons in this layer can be denoted as T_{tij}, where t indexes the corresponding templates, and i and j index the corresponding neuron positions.

A local neighborhood of a pixel P_{ij} in an image is said to *match* a template (or sub-template) if each pixel in the template (or sub-template) and the corresponding pixel in the neighborhood of pixel P_{ij} match each other; that is, if black pixels in the image appear at positions represented by symbol '1' in the template (or sub-template), white pixels appear at positions represented by symbol '0', and at least one white pixel appears at positions represented by symbol 'y'. The match of the neighboring neurons of a neuron P_{ij} in the image neuron plane with a template or sub-template is defined similarly, except that the black pixels and the white pixels in an image are replaced by active neurons and inactive neurons in the image neuron plane, respectively.

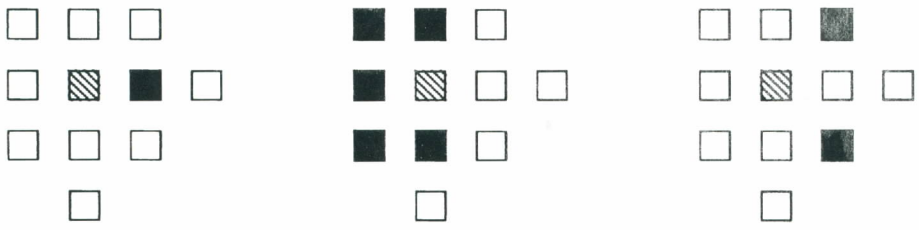
If template t matches neighbors of neuron P_{ij}, it can be seen that each sub-template, T_{ts}, will also match the neighbors of neuron P_{ij} since any sub-template is a sub-area or a combination of sub-areas of template t. On the other hand, if all the sub-template T_{ts} have been checked to match the neighbors of neuron P_{ij}, the only neurons which are still not checked are those corresponding to the don't-care pixels (denoted by 'x') and P_{ij} itself. But it is unnecessary to consider these two kinds of pixels in our template matching process. Therefore, we have the following property.

Property 1: A template t is matched by the neighbors of a neuron P_{ij} in the image neuron plane if and only if the sub-template T_{ts} corresponding to each symbol s in symbol set S_t is matched by the neighbors of neuron P_{ij}.

The neuron planes in the S-layer can be characterized further into three classes. The planes for matching sub-templates corresponding to symbol '1' form one class, called *type 1 planes*. The planes for matching sub-templates corresponding to symbols '0' and 'y' form the other two classes, called *type 0* and *type y planes*, respectively. As shown in Fig. 4, the labels appearing at the lower-left corner of the neuron planes in the S-layer specify the plane types and the corresponding templates of the planes. For example, the label '0' of the top neuron plane in the S-layer indicates that this plane is responsible for matching a sub-template of template (a) and is a type 0 plane.

The receptive field of a neuron in the S-layer is called a *separation field*, abbreviated as *S-field*. The shape of each of these fields is different from that of the others. However, any S-field of a neuron is a sub-field of the 10-neighborhood of the corresponding neuron in the image neuron plane. More specifically, the S-field V_{tsij} of a neuron S_{tsij} in the S-layer is a set of neurons, which is part of the 10-neighborhood of neuron P_{ij} in the I-layer, and each of these neurons is located at a position in the I-layer corresponding to an element of the sub-template of t in symbol s, i.e.,

$$V_{tsij} = \{P \mid P \text{ is one of the 10-neighbors of a neuron } P_{ij} \text{ in the I-layer, and } P \text{ is located in any of the corresponding areas of sub-template } T_{ts}\}.$$



(a). S-field of a neuron in plane '$a0$'

(b). S-field of a neuron in plane '$a1$'

(c). S-field of a neuron in plane 'ay'

Fig. 6. Receptive fields (S-field) of neurons in neuron plane '$a0$', '$a1$', and 'ay'. The black rectangles in each figure represent the neurons covered by an S-field. The rectangles filled with dashed lines represent neuron P_{ij}.

For example, Fig. 6 shows the S-fields of all the neurons which are responsible for matching a sub-template of template (a), where Fig. 6(a), Fig. 6(b) and Fig. 6(c) show the S-fields of neuron planes '$a0$', '$a1$' and 'ay', respectively. Since each S-field is part of the 10-neighborhood of a neuron P_{ij} in the I-layer, the 10-neighbors of neuron P_{ij} in the I-layer and P_{ij} itself are shown for illustration. The black rectangles in each figure respresent the neurons covered by an S-field, and the rectangle filled with dashed lines represents neuron P_{ij}.

4. WEIGHTS AND ACTIVATION FUNCTIONS

No training phase before thinning is required for the proposed neural network. However, to perform thinning correctly, the strength of each connection should be initialized properly. That is, the weights for the inputs into each neuron should be set properly in advance.

Consider a neuron S_{tsij} in the S-layer, which is responsible for matching sub-template T_{ts}. Let all of its inputs have the same weights,

$$\mathrm{WIS}_{tsij} = 1/N_{ts}, \qquad \text{for all } i, j, \tag{1}$$

where N_{ts} is the number of symbols in sub-template T_{ts}.

Since all the neurons in a plane are treated in an identical manner, the neuron indices i and j can be omitted. Formula (1) can then be abbreviated as

$$\mathrm{WIS}_{ts} = 1/N_{ts}. \tag{2}$$

Table 1. WIS_{ts} values.

template	a	b	c	d	e	f	g
0	1	1	1	1	0.333	0.333	0.167
1	0.2	0.2	0.167	0.167	0.5	0.333	0.5
y	0.5	0.5	0.5	0.5			
template	h	i	j	k	l	m	n
0	0.333	0.333	0.167	0.2	0.2	0.2	0.2
1	0.5	0.333	0.5	0.333	0.333	0.333	0.333
y							

From Formula (2), all the connection weights from the I-layer to the S-layer can be obtained. Table 1 shows the weights for the inputs from which the neurons in various separation planes receive.

By summing all the weighted inputs, the net input of a neuron in the S-layer is

$$N_{tsij} = \sum_{P_{ij} \in V_{tsij}} P^0_{ij} \times \mathrm{WIS}_{ts} + \theta, \tag{3}$$

where P^0_{ij} is the output of neuron P_{ij} in the I-layer, V_{tsij} is the S-field of neuron S_{tsij} in the S-layer, and θ is a small positive bias value for the neurons in a type 1 plane or a small negative value for the neurons in a type 0 plane or a type y plane.

The net input of a neuron in a type 1 plane will be 1 if all the outputs of the neurons in the S-field are 1. For a neuron in a type 0 plane, its net input is 0 if the S-field of this neuron matches the corresponding sub-template. Also, we say that a sub-template corresponding to symbol 'y' is matched if at least one of the values of the corresponding neurons is 0. So, for a neuron in a type y plane, the net sum of its two inputs which match the corresponding sub-template will never exceed 0.5 since the weight of either input is 0.5.

As described above, the net inputs of all the neurons in the S-layer are summed up in an identical way. However, the outputs of these neurons are computed in slightly different ways. In order to activate the neurons in the S-layer when their corresponding sub-template is matched, the output of a neuron, S_{tsij}, in the S-layer with net input N_{tsij} is computed by the following function:

$$S^0_{tsij} = F_s(N_{tsij}), \tag{4}$$

where F_s denotes three threshold functions, each for one plane type, as follows:

$$F_0(x) = \begin{cases} 1 & \text{if } x < 0; \\ 0 & \text{otherwise;} \end{cases}$$

$$F_1(x) = \begin{cases} 1 & \text{if } x > 1; \\ 0 & \text{otherwise;} \end{cases}$$

$$F_y(x) = \begin{cases} 1 & \text{if } x < 0.5; \\ 0 & \text{otherwise.} \end{cases}$$

The bias value θ in Formula (3) is included to guarantee that these three threshold functions can obtain correct results. Therefore, its absolute value should

be small enough to prevent side effects. Since the number of neurons in each receptive field doesn't exceed 10, the value of θ will not infect the correct results of the threshold functions if its absolute value is smaller than 0.1. For instance, the absolute value 0.01 is small enough and is proper for each threshold function.

For any neuron S_{tsij} in the S-layer, its receptive field is part of the 10-neighbors of neuron P_{ij} in the I-layer and corresponds to the sub-template T_{ts}. Consider three cases:

Case 1: neuron S_{tsij} is in a type 0 plane. From the definition of function F_0, neuron S_{tsij} will be active if and only if all the neurons in its receptive field are inactive or, equivalently, if and only if the sub-template T_{ts} is matched by the neighbors of neuron P_{ij}.

Case 2: neuron S_{tsij} is in a type 1 plane. If any of the neurons in the receptive field of neuron S_{tsij} are inactive (i.e., if the sub-template T_{ts} is not matched by the neighbors of neuron P_{ij}), the net input of S_{tsij} is less than 1, which means that neuron S_{tsij} will be inactive after function F_1 is applied. On the other hand, if all the neurons in the receptive field are active (i.e., if the sub-template T_{ts} is matched by the neighbors of neuron P_{ij}), the net input will be greater than 1, which will activate neuron S_{tsij}.

Case 3: neuron S_{tsij} is in a type y plane. Note that only two neurons are in the receptive field of neuron S_{tsij}. From function F_y, neuron S_{tsij} will be inactivated if and only if both of these two neurons are active or, equivalently, if and only if sub-template T_{ts} is mismatched. In other words, neuron S_{tsij} will be activated if and only if sub-template T_{ts} is matched.

From the discussions of the above three cases, we have the following property.

Property 2: A neuron S_{tsij} in the S-layer will be active if and only if the sub-template T_{ts} is matched by the neighbors of neuron P_{ij} in the I-layer.

Next, consider a neuron T_{tij}, which is responsible for matching template t in the T-layer. Let the weight of each of its inputs be defined identically as

$$\mathrm{WST}_t = 1/|S_t|, \tag{5}$$

where $|S_t|$ is the number of elements in symbol set S_t, which includes all the symbols in template t.

The net input, i.e., the sum of the weighted inputs of neuron T_{tij} is

$$N_{tij} = \sum_{s \in S_t} S^0_{tsij} \times \mathrm{WST}_t + \theta, \tag{6}$$

where S^0_{tsij} is the neuron S_{tsij} in the S-layer, and θ is a small positive bias value. Furthermore, its output is

$$T^0_{tij} = Ft(N_{tij}), \tag{7}$$

where

$$F_t(x) = \begin{cases} 1 & \text{if } x > 1; \\ 0 & \text{otherwise.} \end{cases}$$

The determination of the value of θ in Formula (6) is similar to that in Formula (3). Value 0.01 is also proper for it in Formula (6).

By observing function F_t, it can be seen that the sufficient and necessary condition to activate neuron T_{tij} is that the net input should be greater than 1. The condition to obtain this net input is that all the neurons, S_{tsij}, in the receptive field of neuron T_{tij} are active for all symbols, s, in symbols set S_t. From Property 2, this means that all sub-templates, T_{ts}, of template t are matched by the neighbors of neurons P_{ij} for all symbols, s, in symbol set S_t. Therefore, a property can be obtained from Property 1.

Property 3: A neuron, T_{tij}, in the T-layer will be active if and only if template t is matched by the neighbors of neuron P_{ij} in the I-layer.

Finally, consider a neuron, P_{ij}, in the I-layer. Let the weights of the inputs received from the T-layer and the weight received from itself be defined as WTI = 1 and WII = 0.5, respectively.

By summing as the weighted inputs the outputs from the neurons which are located at the same position in all the neuron planes of the T-layer as well as the output from P_{ij} itself, the net input is defined as

$$N_{ij} = P_{ij}^0 \times \mathrm{WII} + \sum_{t \in T} T_{tij}^0 \times \mathrm{WTI} + \theta, \tag{8}$$

where T is the template set and the bias value $\theta = -0.5$. Furthermore, its output is defined to be

$$P_{ij}^0 = F_i(N_{ij}), \tag{9}$$

where

$$F_i(x) = \begin{cases} 1 & \text{if } x = 0; \\ 0 & \text{otherwise.} \end{cases}$$

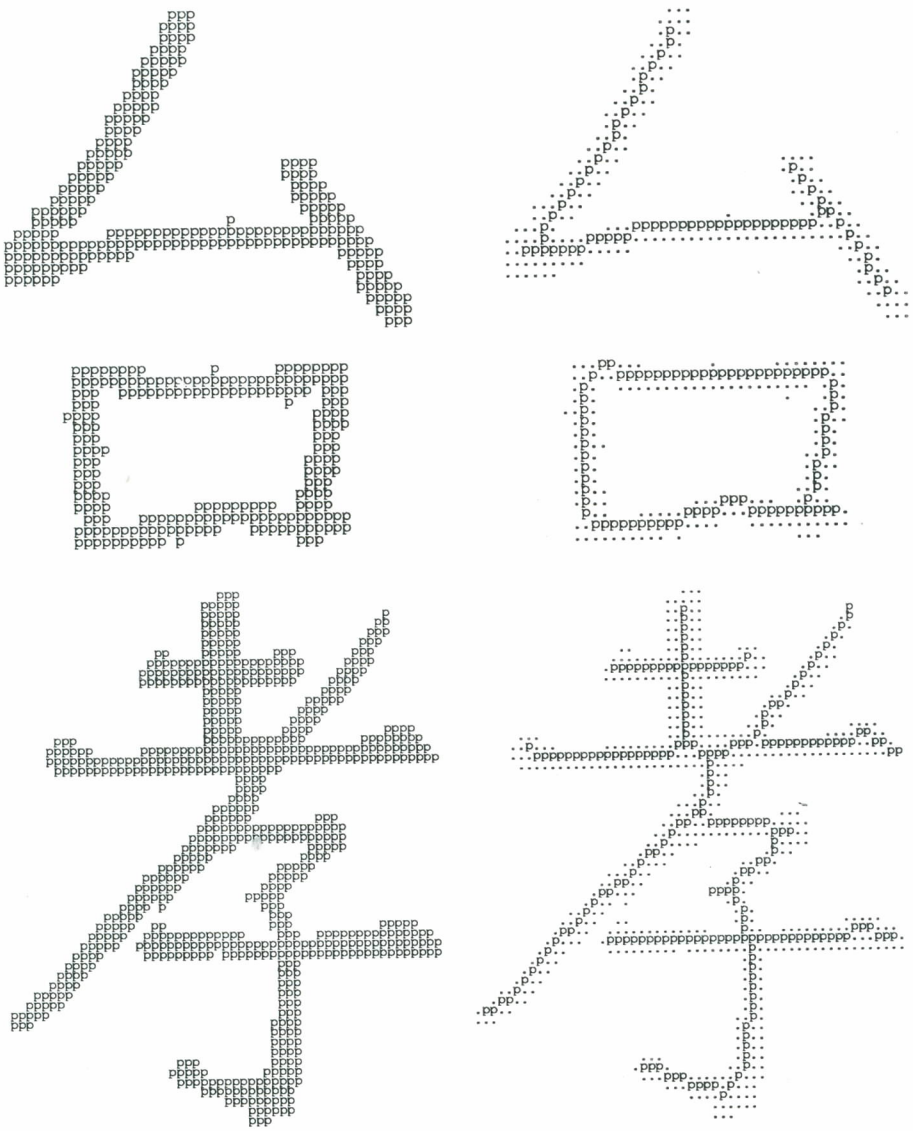
If neuron P_{ij} is active, then the net input N_{ij} must be zero. There is only one case in which such a net input can be obtain, i.e., the case when neuron P_{ij} is active in the previous iteration step, and the net sum of the inputs from the neurons in the T-layer is 0. From Property 3, this zero sum means that all the templates in template set T are not matched. On the other hand, if neuron P_{ij} is active in the previous iteration step, and its neighbors are not matched by any template, t, in template set T, then the value of N_{ij} will be 0, which will activate neuron P_{ij}. Therefore, we have the following property.

Property 4: During the thinning process performed by the proposed neural network, a neuron, P_{ij}, in the I-layer is active if and only if it is active in the previous iteration step, and its neighobors are not matched by any template, t, in template set T.

Theorem 1: When the proposed neural network converges, it yields in the I-layer the same result as does the parallel thinning algorithm OPPTA.

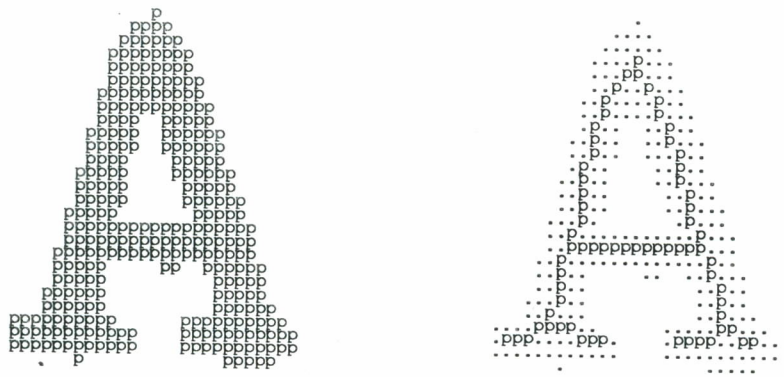
***Proof*:** Let's say that the I-layer correctly represents an image if all the neurons corresponding to the black pixel of the image are active, and all the neurons corresponding to the white pixels are inactive. Initially, an image f^0 is fed into the I-layer with the neurons corresponding to the black pixels being active and the others being inactive. At this time, the I-layer correctly represents image f^0, which

we assume is also the input image for algorithm OPPTA. Assume that the I-layer correctly represents image f^i in algorithm OPPTA at iteration i. Then, at iteration $i+1$, by Property 4, the I-layer will correctly represent image f^{i+1}. By induction, when the neural network converges, i.e., when no more pixel can be removed, the I-layer will correctly represent the thinning result of algorithm OPPTA.



(a) Original image. (b) Thinning results.

Fig. 7. Thinning results of two Chinese characters by using the proposed neural network.



(a) Original image. (b) Thinning results.

Fig. 8. Thinning results of English character 'A' by using the proposed neural network.

5. EXPERIMENTAL RESULTS

Some experiments have been conducted to illustrate the effects of the proposed neural network. Fig. 7 and Fig. 8 show the results in which skeleton points are marked by 'p' and removed points by '·'.

Fig.7 shows the results of thinning two Chinese characters of dimension 64×64. It can be seen that the proposed neural network obtains perfectly 8-connected and noise-insensitive skeletons without excessive erosion, exactly the same as those obtained by algorithm OPPTA. Similar results can be observed in an experiment of thinning an English character '*A*' of dimension 32×32. Fig. 8. shows the results of this experiment.

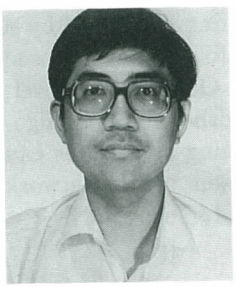
6. CONCLUSIONS

A neural network which can perform thinning of binary images has been proposed. From the properties and theorem, and the good thinning results, it has been proved both theoretically and experimentally that thinning an image by a large number of simple processing elements in parallel is possible. The proposed neural network is based on a fast parallel thinning algorithm, OPPTA. However, the scheme for constructing such a network is quite general for other thinning algorithms; neural networks based on other one-pass parallel thinning algorithms can also be constructed by using the same design principle. A problem of this network is its huge size. Simplification of the network is worthwhile for further study.

REFERENCES

1. Stefanelli, R. and Rosenfeld, A., "Some parallel thinning algorithms for digital

pictures," *J. ACM*, Vol. 18, No. 2, 1971, pp. 255-264.
2. Zhang, T.Y. and Suen, C.Y., "A fast parallel algorithm for thinning digital patterns," *Commun. ACM*, Vol. 27, No. 3, 1984, pp. 236-239.
3. Guo, Z. and Hall, R.W., "Parallel thinning with two-subiteration algorithms," *Commun. ACM*, Vol. 32, No. 3, 1989, pp. 359-373.
4. Holt, C.M., Stewart, A., Clint, M., and Perrott, R. H., "An improved parallel thinning algorithm," *Commun. ACM*, Vol. 30, No. 2, 1987, pp. 156-160.
5. Chin, R.T., Wan, H.K., Stover, D.L., and Iverson, R.D., "A one-pass thinning algorithm and its parallel implementation," *Computer Vision, Graphics, and Image Processing*, Vol. 40, No. 1, 1987, pp. 30-40.
6. Chen, C.S. and Tsai, W.H., "A new fast one-pass thinning algorithm and its parallel hardware implementation," *Pattern Recongnition Letters*, Vol. 11, 1990, pp. 471-477.
7. Yu, S.S. and Tsai, W.H., "A new thinning algorithm for gray-scale images by the relaxation technique," *Pattern Recognition*, Vol. 23, No. 10, 1990, pp. 1067-1076.
8. Cortes, C. and Hertz, J.A., "A network system for image segmentation," *International Joint Conference on Neural Networks*, 1989, pp. 121-125.
9. Matsumoto, T., Chua, L.O., and Yokohama, T., "Image thinning with a cellular neural network," *IEEE Trans. on Circuits and Systems*, Vol. 37, No. 5, 1990, pp. 638-640.
10. Chua, L.O. and Yang, L., "Celluar neural networks: theory," *IEEE Trans. on Circuits and Systems*, Vol. 35, No. 10, 1988, pp. 1257-1272.
11. Chua, L.O. and Yang, L., "Cellar neural networks: application," *IEEE Trans. on Circuits and Systems*, Vol. 35, No. 10, 1988, pp. 1273-1290.
12. Baird, H.S., Graf, H.P., Jackel, L.D., and Hubbard, W.E., "A VLSI architecture for binary image classification," in Simon, J.C. (ed.), *From Pixels to Features*, North-Holland, Amsterdam, 1989, pp. 275-285.
13. Graf, H.P., Jackel, L.D., and Hubbard, W.E., "VLSI implementation of a neural network model," *IEEE Computer*, 1988, pp. 41-48.
14. Wu, R.Y. and Tsai, W.H., "A new one-pass parallel thinning algorithm for binary images," *Pattern Recognition Letters*, Vol. 13, 1992, pp. 715-723.

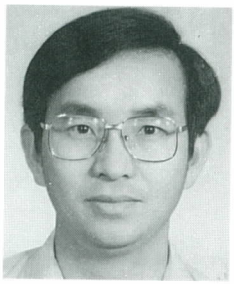


Rei-Yao Wu (吳瑞堯) was born on Oct. 17, 1960 in Changhua, Taiwan, Republic of China. He received the B.S. degree from the Department of Computer Engineering in 1983, the M.S. degree from the Institute of Computer Engineering in 1985, and the Ph. D. degree from the Institute of Computer Science and Information Engineering in 1993, all from National Chiao Tung University, Hsinchu, Taiwan.

In 1985, he joined the Technical Research Division of the Institute for Information Industry, Taipei, Taiwan. He has joined many projects, including the research and development of software engineering environments, study of a man-machine interface, developement of a desktop publishing system, and development of Chinese MS-Windows. He was a project

manager for the research and development of an images and graphics technology project.

He is now an Associate Professor at the World College of Journalism and Communications. His current research interests include image processing, pattern recognition, neural networks, and optical character recognition.



Wen-Hsiang Tsai (蔡文祥) was born in Tainan, Taiwan, Republic of China on May 10, 1951. He received the B.S. degree from National Taiwan University, Taipei, Taiwan, Republic of China in 1973, the M.S. degree from Brown University, Providence, Rhode Island, U.S.A. in 1977, and his Ph.D. degree from Purdue University, West Lafayette, Indiana, U.S.A. in 1979, all in electrical engineering.

Since November 1979, he has been on the faculty of the Institute of Computer Science and Information Engineering at National Chiao Tung University, Hsinchu, Taiwan. From 1984 to 1986, he was an Assistant Director and later an Associate Director of the Microelectronics and Information Science and Technology Research Center at National Chiao Tung University. He joined the Department of Computer and Information Science at National Chiao Tung University in August 1984, acted as the Head of the Department from 1984 to 1988, and is currently a Professor there. He serves as a consultant to several research institutes and industrial companies. His current research interests include computer vision, image processing, pattern recognition, and Chinese information processing.

Dr. Tsai is an Associate Editor of Pattern Recognition, the International Journal of Pattern Recognition and Artificial Intelligence, the Journal of the Chinese Institute of Engineers, and the Journal of Information Science and Engineering, and was an Associate Editor of Computer Quarterly and Proceedings of National Science Council of the Republic of China (Part A). He was elected as an Outstanding Talent of Information Science of the Republic of China in 1986. He was the winner of the 13th Annual Best Paper Award of the Pattern Recognition Society of U.S.A. He obtained the 1987 Outstanding Research Award and the 1988-1989, 1990-1991, 1992-1993 Distinguished Research Awards of the National Science Council of the Republic of China. He also obtained the 1989 Distinguished Teaching Award of the Ministry of Education of the Republic of China. He was the winner of the 1989 and 1992 Acer Long Term Award for Outstanding Ph.D. and Master Thesis Supervision, the 1991 Xerox Foundation Award for Ph.D. Dissertation Study Supervision, and the 31st S.K. Chuang Foundation Award for Science Reserach. He was also the winner of 1990 Outstanding Paper Award of the Computer Society of the Republic of China. Dr. Tsai has published more than seventy papers in well-known international journals.

Dr. Tsai is a senior member of the IEEE, and a member of the Chinese Image Processing and Pattern Recognition Society, the Computing Linguistics Society of the Republic of China, and the Medical Engineering Society of the Republic of China.