Fast one pass knowledge-based system for thinning

Maher Ahmed
Rabab Kreidieh Ward
University of British Columbia
Electrical Engineering Department
Vancouver, BC, V6T 1Z4, Canada
E-mail mahera@ee.ubc.ca
E-mail: rababw@cicsr.ubc.ca

Abstract. A knowledge based system that uses one pass only for thinning is described. The study of existing thinning algorithms reveals that appropriate thinning techniques are application dependent. Thinning algorithms suitable for applications such as Chinese characters unfortunately leave extraneous pixels (branches) in other applications such as Arabic characters. Alternatively, existing algorithms that delete extraneous pixels in certain applications may delete important pixels in other applications. We present a general system where the user can select the degree of freedom of having branches in the output pattern according to his or her application. While the user selects one of three possible options, the system default is suitable for most usual applications such as English characters. Our system is a knowledge-based system whose rules guarantee the connectivity and preserve the symbol shape to a great extent. The implemented system can be used as a general thinning method for various applications. By having some knowledge about the input symbols, the user determines the best thinned output pattern or the degree of branches allowed. The implemented system helps to test the effect of each rule separately. Prolog built-in features for backtracking, strong logic handling and rule manipulation have proved to be very helpful in implementing this system.

I Introduction

Thinning is the process of reducing the thickness of each line of patterns to just a single pixel. This is usually used as the first step in applications such as optical character recognition (OCR). A thinning algorithm is considered to be useful if it preserves the shape of the original image, preserves the connectivity, and does not leave extraneous pixels (branches). A comprehensive survey of thinning algorithms is described by Lam et al. Templates of 3X3 windows are usually used to perform the thinning and trimming of the extraneous pixels. However, using these windows simultaneously without extra conditions may cause the loss of connectivity. Kim et al. introduced new 3X3, 3X4 and 4X3 windows with extra conditions to perform parallel thinning which preserves connectivity. In some thinning algorithms, the 45-deg diagonal lines that are 2 pixels wide only become 1 or 2 pixels. A solution for this problem is suggested by Raju and Xu. Thinning involves deleting pixels, but at the same time preserving the shape and connectivity. To preserve connectivity and shape, many extraneous as well as important branches are also preserved. In some applications such as Chinese characters it is important to preserve these branches, while for applications such as hand-written Arabic
character recognition these branches should be minimized. Parallel thinning algorithms for Arabic OCR are presented by Tellache et al. 4 To preserve the shape and delete the branches (extraneous pixels), the algorithm imposes certain conditions to be satisfied by the pixels that will be deleted. The first and second conditions preserve the end points and the connectivity. The third and fourth conditions ensure that the end point must be one of the following points: an east border, west border, south border, north border, northwest corner, northeast corner, southeast corner or a southwest corner. An artificial neural network based on adaptive resonance theory (ART) has been used successfully for thinning Arabic characters by Altuwaijri and Bayoumi. 5 This thinning algorithm clusters the data image, then the skeleton is generated by plotting the cluster centers. Finally, adjacent clusters are connected by straight lines.

Researchers have presented various techniques for thinning. These methods differ by using different templates or conditions. None of these techniques can be used as a general thinning method, and each method is suitable only for certain applications. They address the thinning problem by matching each window in the pattern (usually a 3X3 window) with a predetermined set of windows. This predetermined set of windows is designed so that the central pixel of the pattern window will be “deleted”. In our algorithm, we first describe (through certain rules) the situations that maintain the connectivity of the ink pixels. If any of these rules is satisfied then the ink pixel will be “maintained”. However, according to these rules all branches whether extraneous or not are also maintained. Other rules describe the conditions under which the extraneous branches will be deleted. In addition, important branches with their end points will be preserved.

Our goal is to implement a thinning system that can be used generally. Our system has four processing directions. For each processing direction, there are three thinned options for the output pattern. The processing direction and the appropriate pattern are to be selected by the user according to his or her application.

2. Knowledge-Based Systems

A knowledge based system is a computer program that performs a specific task normally done by a human expert when the human expertise is in high demand. Knowledge based systems have the ability to manipulate problem statements and integrate relevant pieces of knowledge from the knowledge base using heuristic techniques. These systems are problem-solving computer programs that use their knowledge in flexible ways, and can reason with inexact knowledge which may be declarative or procedural. The difference between knowledge-based systems and ordinary programs is that the knowledge based systems do not encode the knowledge but they represent the knowledge in a higher-level form. These forms include rules and facts. They are similar to the way people think about problems. They have an inference engine for reasoning with rules and facts.
We designed a knowledge based system for thinning that has only 12 rules. The system is able to draw conclusions about how the thinned output pattern was reached, and which rules were applied. The system can learn new rules if desired. Knowledge is distributed over the individual rules, which are independent of each other. An important goal is to establish an inference mechanism among these rules. We have implemented this knowledge based system by using the Prolog language, as well as the object-oriented language C++. The program runs on an IBM-PC. Prolog has the advantage of being a descriptive language; it solves the problem by describing it rather than showing how to solve it. The Prolog language is the most appropriate language for implementing knowledge based systems. C++ language has the advantages of being fast and capable of handling large documents.

3. Knowledge in a Symbol

Most of the knowledge in any digitized pattern is contained in a small fraction of pixels. These pixels are mainly the central lines of the patterns. However, these central lines are computationally expensive to determine. As an alternative, we suggest to represent the symbol segments by their edges. However, there may be more than one edge in a segment (e.g., for a thick straight line there will be right, left, top, and bottom edges). Thus, we shall represent the segment by one (or more than one) of these edges.

Processing each line of the pattern (pixel by pixel) from left to right, and from top to bottom will have the tendency to preserve the right and bottom edges as shown in Fig. 1. Alternatively, processing the pattern from right to left, and top to bottom will preserve the left and bottom edges as shown in Fig. 2. Similarly, processing the pattern from left to right, and bottom to top will preserve the right and upper edges as shown in Fig. 3. Processing the pattern from right to left, going bottom to top will preserve the left and upper edge as shown in Fig. 4.

Our objective is to thin the pattern but preserve, as closely as possible, the symbol's shape. Left or right edges can be used to represent many symbols accurately (such as English characters). However, when symbols have many branches (such as Chinese characters), it is more useful to process the pattern not only from just one direction (e.g., left or right) but from many directions.
Fig. 1 Symbols processed from left to right going from top to bottom:
(a) the original symbols, (b) a large number of branches, (c) a medium number of branches, and (d) a small number of branches.

Fig. 2 Symbols processed from right to left going from top to bottom:
(a) the original symbols, (b) a large number of branches, (c) a medium number of branches, and (d) a small number of branches.

Fig. 3 Symbols processed from left to right going from bottom to top:
(a) the original symbols, (b) a large number of branches, (c) a medium number of branches, and (d) a small number of branches.
Fig. 4 Symbols processed from right to left going from bottom to top:

(a) the original symbols, (b) a large number of branches, (c) a medium number of branches, and (d) a small number of branches.

Fig. 5 Symbol M and the Arabic character letter “seen” shown in Figs 1(a), 1(b) and 1(c) magnified.

The information in any symbol lies in the changes in its shape. A straight line with one or more branches has more knowledge than a simple straight line. In some applications, it is desirable to consider a straight line with three branches as a straight line. However, if this pattern represents an Arabic letter (Seen) when written at the beginning or the middle of a word then it will appear as or (i.e., the branches may or may not be there). This letter “seen” when written at the end of the word or in isolated form is shown in the third symbol in column five in Fig. 1 and magnified in Fig. 5. For this letter, any of the twelve options that we have proposed may be used except for option of Fig. 2(d). On the other hand, for the letter “D” the option of Fig. 2(d) along with option of Fig. 3(d) form the best option for its thinning. Another example is the letter M shown in Fig. 1 and magnified in Fig. 5. If it is to be considered as the letter M then, the pattern generated in Fig. 1(c) will be better than the one shown in Fig. 1(b) which has many extraneous branches. However, if the symbol does not represent the letter M then, the step edges in the left edge of the left diagonal may represent some knowledge. In this case the option of Fig. 1(b) should be selected.

Processing symbols from different directions will suit certain symbols. Fig. 6 shows the output thinned symbol by going through the outer symbol limits of the symbol, starting from the upper left corner to the upper right corner, going down to the bottom right corner, across to the bottom left corner, proceeding
upwards to the upper left corner and reducing the limits of each cycle until reaching the center. Another method as shown in Fig. 7, is to go one cycle clock wise, and another counter clockwise starting at two different corners (as for example the upper left corner and the bottom right corner). Processing symbols from different directions will maintain the geometric shape for symbols, but may leave many extraneous branches.

![Fig. 6](image1)

**Fig. 6** Symbols thinned by going clockwise direction starting at the upper left corner and having a large number of branches.

![Fig. 7](image2)

**Fig. 7** Symbols thinned by going clockwise direction starting at the upper left corner and followed by a counter clockwise direction starting at the bottom right corner alternately and having a large number of branches.

### 4. Thinning System

Thinning reduces the binary image into one in which most dark pixels have one, two or three dark neighbors. Our thinning system has the following characteristics:

1. It preserves and ensures the continuity of the symbol.
2. It preserves the shape of the symbol by preserving some edges.
3. It preserves the end points. An end point is defined here as the start or end of edges, and also as a point where two straight edges meet.

Existing thinning algorithms are designed to be suitable for just one application. They also produce just one output pattern. Our knowledge-based system has the unique feature of producing different outputs to suit different applications. The appropriate output will be selected by the user according to his or her application. For example, if the user knows that the branches in the input pattern convey information such as Chinese characters, he or she should select the method resulting in Figs. 1(b), 2(b), 3(b) and
4(b) where the algorithm magnifies the changes in the patterns. On the other hand, if the patterns in the application contain a moderate number of branches, he or she should select the method resulting in Figs 1(c), 2(c), 3(c) and 4(c). If the application has symbols that have few branches, the user should select the method resulting in Figs. 1(d), 2(d), 3(d), and 4(d). Finally, if the geometric shapes of symbols are to be retained, then the user should select the method resulting in Fig. 7.

Figs 1 through 4 represent the results of processing the patterns from different directions. Each Figure represents the results of the three proposed methods.

5. Rules Explanations

The thinning rules are listed in Fig. 8. There are 12 rules in our knowledge-based system. The rules are applied to the pattern with a specific order (rule 1, 2, 3 and so on). Rule 1 prevents the system from creating unnecessary holes. The rules that preserve the connectivity are rules 4, and 7 through 11. The rules that preserve the shape and the end points are rules 2, 3, 5 and 6. Finally, rule 12 deletes the pixel if none of the previous rules are applied.

Rule 4 describes the pattern

\[
\begin{array}{ccc}
\times & \times & \times \\
0 & 1 & 0 \\
\times & \times & \times \\
\end{array}
\]

where we can not delete the middle pixel in order to maintain the connectivity. For this pattern, only if we have

\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 1 & 0 \\
\times & \times & \times \\
\end{array}
\]

or

\[
\begin{array}{ccc}
\times & \times & \times \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{array}
\]

we could delete the middle pixel without losing the connectivity. The last two conditions (patterns) must be checked before applying rule 4. These last two patterns are described by rules 2 and 3. Similarly, rule 7 describes the pattern
where we should not delete the middle pixel to maintain the connectivity. For this pattern, only if we have these conditions

```
 0 0 x
0 1 x
0 0 x
```

or

```
 x 0 0
x 1 0
x 0 0
```

we could delete the middle pixel without losing the connectivity. We have to check these last two conditions (patterns) before applying rule 7. These last two patterns are described by rules 5 and 6. Consequently, rules 2, 3, 5 and 6 (shown in Fig. 8) are responsible for the number of branches (extraneous pixels) in the output pattern. Each of these rules has two shaded squares. If we replace one of these shaded squares by x (don’t care), we will get thinned symbols that have a lower number of branches (strokes). If we replace both shaded squares by x (don’t care), we will get thinned symbols that have an even lower number of branches (strokes). Hence, the system produces three outputs. Rule 8 describes the pattern

```
1 0 x
0 1 x
x x x
```

where to maintain the connectivity, we can not delete the middle pixel. Rules 9, 10, and 11 are similar to rule 8. Finally, rule 12 deletes the middle pixel if non of the previous 11 rules were applied.

Some symbols before thinning are shown in Fig. 1(a). The symbols after thinning and without changing rules 2, 3, 5 and 6 (i.e. having ones in the shaded squares) are shown in Figs. 1(b), 2(b), 3(b) and 4(b). The symbols after thinning and changing rules (3, 4, 6 and 7) by having one x (don’t care) in either one of the two shaded squares are shown in Figs. 1(c), 2(c), 3(c) and 4(b). The symbols after thinning and changing rules (3, 4, 6 and 7) by having two x (don’t care) in the shaded squares are shown in Figs. 1(d), 2(d), 3(d) and 4(d).
We conclude that some rules are good for certain symbols, while others are not. The processing direction will affect the results to some extent. One of the advantages of our knowledge-based system is that we can determine which output is correct, and what rules are suitable for each application. The default scanning method processes the pattern from left to right, going from top to bottom. The default thinning rules (2, 3, 5 and 6) are the ones that have one don’t care in the shaded squares.

6 Conclusion

We have described a knowledge based system for a general thinning. We have shown that symbol understanding is part of producing good thinning. Since thinning is application dependent, our system produces three thinning methods and four processing directions. The user selects the best method and processing direction according to his/her application. Our system has a default method that produces, in general, good outputs. However, for excellent results the user has to select the most suitable option for the application at hand.

This system is fast and needs only one pass. It has 12 independent thinning rules that preserve the connectivity and the shape of the symbol.

A challenging task remains to develop a system that can automatically recognize the application. Consequently, the degree of maintaining the branches and the processing direction will be also recognized. This may be accomplished by the system’s ability to understand and infer from the previous processed symbols.
References


Maher Ahmed is doing his PhD at University of British Columbia, Vancouver. He received MSc degree in systems and control from Queen's University, Kingston, Ontario in 1994. He was working at Ontario Hydro during 1990 to 1991. He received MSc degree in computer science from Cairo University in 1988. He was working in the National Research Center, at Cairo, Egypt from 1987 to 1990. His research interests include pattern recognition, artificial neural networks and expert systems.

Rabab Kreidieh Ward was born in Beirut, Lebanon. She received the BEng. degree from the University of Cairo, Egypt, and her Masters and PhD. degrees from the University of California, Berkeley. She is the Director of the Centre for Integrated Computer Systems Research and Professor in the Electrical Engineering Dept. at the University of British Columbia. Her research interests are mainly in the areas of signal processing and image processing. She has made contributions in the areas of signal detection, image encoding, compression, recognition restoration and enhancement, and their applications to infant cry signals, cable TV, HDTV, medical images, and astronomical images. She holds five patents related to cable television picture monitoring, measurement and noise reduction. Applications of her work have been transferred to U.S. and Canadian industries. She is the inventor of the non-intrusive measurement method for cable TV video impairments, currently being incorporated into video instruments by a leading U.S. instruments and computer manufacturing company. She is co-inventor of a non-interfering video system for measuring size and biomass of fish in cages and tanks currently being tested for manufacturing by a Canadian company.