Image Processing: Finding Structures In Images

by Max Viktor Spreng, SN: 17130609, 23/03/18

1.0 Abstract

I explore the fundamentals of skeletonization in image processing by taking a look at basic thinning algorithms such as the Stentiford Thinning algorithm, the Zhang-Suen algorithm and a simple Force-Based Thinning algorithm, the charged particle method (CPM-algorithm).
I explain the background and real world application of these algorithms.
Supporting the results and the overall experimental progress I assemble information about the basic theory, terminology and essential methods behind each algorithm giving an in-depth introduction into the field of structural image processing.
I take a look at the strength and weaknesses and especially flaws of each algorithm. I try to enhance performance by using pre- and post-processing algorithms which are thought to eliminate certain unwanted artefacts like spurious projections and necking that result from the different thinning methods and reflect on the effectiveness of those.
As this is thought to be an introduction into the field of thinning algorithms I provide ideas on how these very basic algorithms could be improved in further application or research.

2.0 Background

Digital and analogue image processing are the two major areas of image processing whereas this paper deals solely with the topic of digital image processing. The standard process of a digital image undergoing image processing is done by computers and can be divided into three main sections: The pre-processing, the enhancement and the post-processing. Pre-processing is a process at a low level of abstraction and meant to prepare the image for the actual enhancement process by reducing unwanted effects. The enhancement process is the main part of image processing and its functionality can vary broadly depending on its application. Post-processing basically has the same intention as pre-processing. But other than pre-processing in some times it even noticeably changes the appearance of an image.

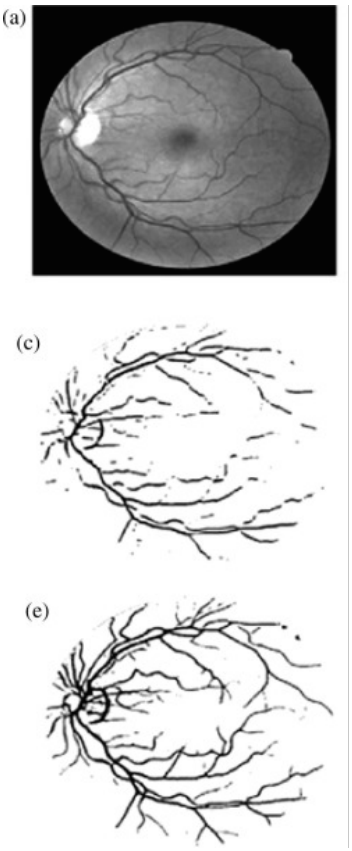


Figure 1.0 Retina Skeletonized

In this case I investigate thinning algorithms which are a certain type of an image enhancement process. More specifically one regards these as a pre-processing method that focuses on the skeletonization of an image. Ideally, that means reducing the actual shape of the object displayed in an image to its topological skeleton which is a thin representation of the object. This is done by distributing importance to different geometrical and topological properties of the object and then conserving those in the process of thinning. Connectivity, direction, length and width are four examples for the criteria.

2.1 Application

Before I start introducing the fundamental theory of thinning algorithms I want to describe the vast amount of possible applications for the skeletonization of images. Nowadays, improved strong and efficient thinning algorithms are used in famous mobile applications for face recognition. They are used for security and medical purposes for example fingerprint recognition (see *Figure 1.0*) and breaking down images of the human retina (see *Figure 2.0*).

Figure 2.0 – Fingerprint Thinning Algorithm – a) original image b) binary image c) binary image after thinning process

2.2 Basics and Terminology

To explain the theory behind thinning algorithms, or scientifically called medial-axis transforms or iterative morphological methods, I should first introduce basic terminology. All those words are based on the foregoing fact that for simplicity we now look at an image as an array of ones and zeros, each number representing a pixel of the image. A zero indicates a black pixel (object) and a one indicates a white pixel (background) as illustrated in *Figure 3.0*.

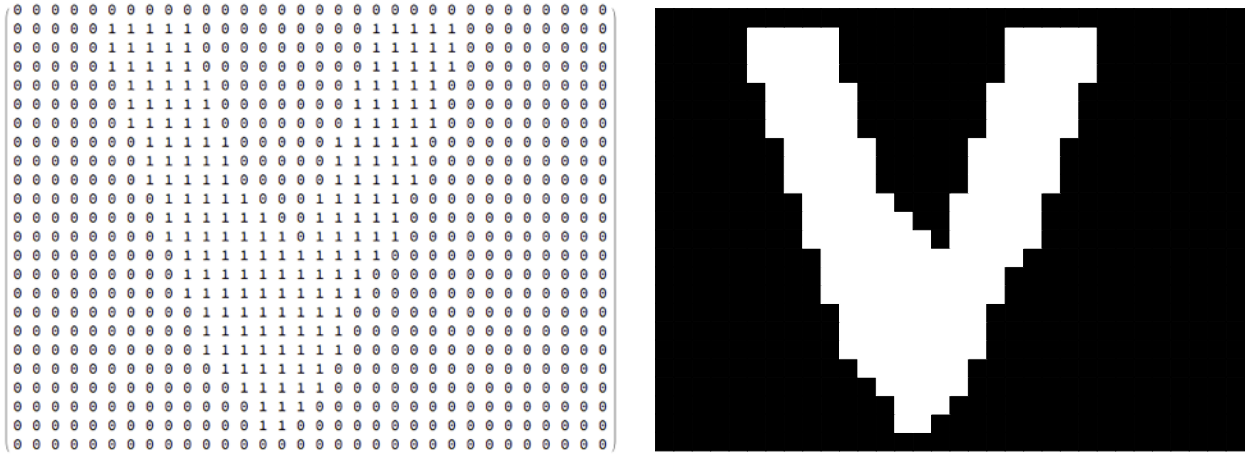


Figure 3.0 Binary Image

8-neighbours: The 8-neighbours of a central pixel P are defined as the surrounding pixels P_1 to P_8 as illustrated in *Figure 4.0*.

6-neighbours: The 6-neighbours of a central pixel P are defined as the surrounding pixels P_1, P_3, P_4, P_5, P_7 and P_8. (see Image 4.0)

P_8	P_1	P_2
P_7	P	P_3
P_6	P_5	P_4

Figure 4.0 8-neighbours

4-neighbours: The 4-neighbours of a central pixel P are defined as the pixels P_1, P_3, P_5 and P_7 and are also called direct neighbours or D-neighbours. (see *Figure 4.0*)

End-Point: A black or zero-pixel with no more than one black 8-neighbour pixel.

Edge-Point: A black or zero-pixel with at least one of its 4-neighbours being a white pixel.

Connectivity-Number (CN): The connectivity-number is a way of quantising the level of connectivity of a single black pixel. It is calculated by the following formula:

$$C_n = \sum_{k=1,3,5,7} \left(p_k - p_k p_{k+1} p_{k+2}\right), \tag{1}$$

Where p_k denotes the value of the pixel (either 1 or 0) and the index k specifies the direct neighbours (8-neighbours) of the central pixel (see *Figure 4.0*). Depending on its 8-neighbours a black pixel can thus be categorised by a connectivity number between zero and four, where four is the strongest connectivity and 0 the weakest (see *Figure 5.0*).

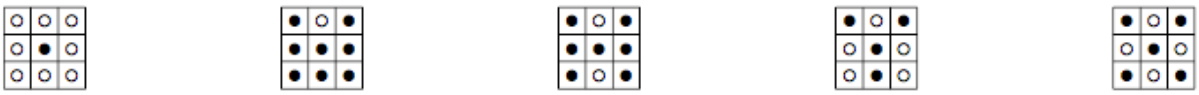


Figure 5.0 Connectivity Number

Possible constellations of the connectivity numbers from 0 to 4 are chronically shown in *Figure 5.0*. The central black pixel in the first figure with (CN = 0) would be called an isolated-point. The central pixel in the second (CN = 1) would be called edge-, in the third (CN = 2) connecting-, in the forth (CN = 3) branching- and in the fifth (CN = 4) crossing-point.

2.3 Basic Functionality of Iterative Morphological Methods

At first focusing on the main process of basic binary thinning algorithms leaving pre- and post-processing methods for later the basic functionality can be described as follows:

In the beginning the image that is about to be processed needs to be converted into a binary image so its image data can be expressed as an array of zeros and ones (see *Figure 3.0*). The general procedure now is as follows: The algorithm goes through all inner points of the array, which is representing the image, from top left to bottom right and for every ”one” in that array (representing a black pixel) it checks whether algorithm specific conditions for the pixel survival are met and accordingly marks the black pixel for deletion or not. After the algorithm went through all inner points, all marked pixels get deleted, which in that case means setting the value of the entry in the array representing the pixel from ”zero“ to ”one” (from black to white). If at least one pixel got marked and deleted in the last iteration, the algorithm starts off again with the updated array scanning and checking the pixels from top left to bottom right. It iterates until in one whole marking process every single pixel met the criteria for survival and thus the array of the binary image stays the same. At this point the algorithm finishes and returns the updated image.

Pre- and post-processing methods can be prepended or appended to the enhancing process to improve the performance of the algorithms.

3.0 Implementing the Thinning Algorithms

I implement and explore three different thinning algorithms: The Stentiford Thinning Algorithm, the Zhang-Suen algorithm and the force-based CPM-algorithm. In the following I summarise the work I do on the different algorithms focusing on explaining their conditions for pixel survival and special features (see *2.3 Basic Functionality of Iterative Morphological Methods*).

3.1.0 The Stentiford Thinning Algorithm

The Stentiford thinning algorithm is one of the most basic template based thinning algorithms introduced by F.W.M. Stentiford and R.G. Mortimer. Its main pixel survival condition is based on the similarity to the following templates (*Figure 6.0*):

Figure 6.0 Stentiford Templates

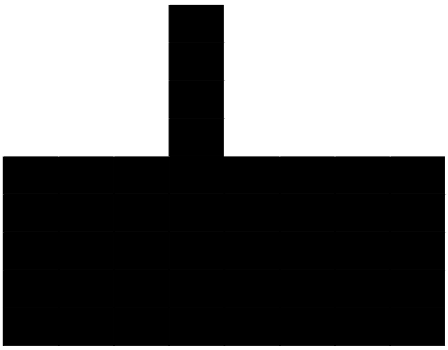
When going through the pixels representing the binary image first it checks every black pixel and its 8-neighbours on similarity to template M1. Where an empty square represents an arbitrary pixel, a white circle represents a white pixel and a black circle represents a black pixel. If the 8-neighbours match template M1 and it is not an end-point (see *2.2 Basics and Terminology*) it gets marked for deletion. After one iteration over the whole array the same process is repeated now with template M2 instead of M1 and ignoring pixels that have already been marked for deletion. After every template has been used parsing through the array, marked pixels get deleted. The algorithm iterates until no pixels are marked for deletion anymore and then finishes.

3.1.1 Pre-Processing

After implementing the Stentiford Thinning Algorithm into Mathematica different artefacts could be observed in the processed pictures. (see *7.0 Appendix: Necking and Spurious Projections, Figure 35.0*) To reduce or eliminate these flaws I try to enhance the performance by implementing two pre-processing methods.

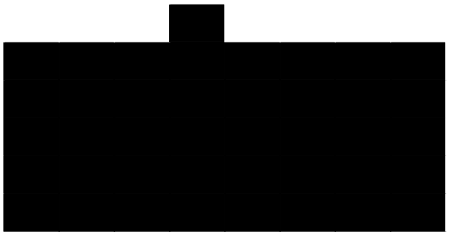
The pre-processing method preventing spurious projections is an algorithms that basically works the same as any iterative morphological method (see *2.3 Basic Functionality of Iterative Morphological Methods*). The condition for deletion of the black pixel is having less than three 8-neighbours and a CN less than two. The algorithm is supposed to reduce thin unwanted continuations, so called ”hairs“, of the object in the image that cross the actual border of the object. The effect is demonstrated using a very simple case of a rectangle having a hair on the top in the following figure (see *Figure 7.0, 8.0 and Figure 20.0 and Figure 35.0*).

Figure 7.0 Spurious Projections



Original Image

Figure 8.0 Spurious Projections Reduced



Processed Image

The pre-processing method preventing necking is even more similar to the Stentiford algorithm than the method for spurious projections because it uses templates as well (see *Figure 10.0 and Figure 21.0*) to eliminate the necking effects (see *7.0 Appendix: Necking and Spurious Projections*). The method searches and eliminates acute angles in the image by following the following conditions for deletion.

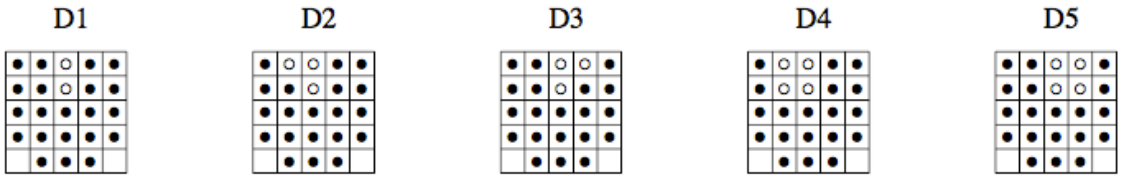


Figure 9.0 Necking Templates

The templates displayed in Figure 10.0 and their 90°-, 180°- and 270°- rotated equivalents must match the 5x5 pixel window with the currently checked pixel in the centre for the algorithm to mark a pixel for deletion. Doing this acute angles can be removed. The effect of the algorithm is shown in the following example image.

Figure 10.0 Pre-processing Necking

Original Image

Processed Image

3.2.0 Zhang-Suen Algorithm

In contrast to the Stentiford algorithm, the Zhang-Suen algorithm does not make use of specific templates to determine the pixel survival. Secondly, the Zhang-Suen algorithm consists of two sub-iterations. In the first sub-iteration the condition for pixel deletion is the following: The current central pixel has a CN equal to one, between two and six 8-neighbours and one or more of the vertical and the right D-neighbours and one or more of

the vertical and the left D-neighbours is white. After pixel deletion the second sub-iteration with the following condition for deletion starts: The current central pixel has a CN equal to one, between two and six 8-neighbours and one or more of the horizontal and the bottom and one or more of the horizontal and the top D-neighbours is white. The algorithm again deletes all marked pixels and starts to iterate until no pixel is marked and then terminates.

3.2.1 Post-Processing – Staircase Removal Algorithm

The staircase removal algorithm introduced by Christopher M. Holt is a post-processing method that can improve the thinning images produced by the Zhang-Suen algorithm by eliminating central pixels matching one of the following templates or their 180° rotated equivalent and having exactly one of its D-neighbours white. (see *Figure 11.0*). ”x” denotes an arbitrary value.

$$\begin{pmatrix} 1 & 0 & x \\ 0 & 0 & x \\ x & x & 1 \end{pmatrix} \qquad \begin{pmatrix} x & 0 & 1 \\ x & 0 & 0 \\ 1 & x & x \end{pmatrix}$$

Figure 11.0 Holt Templates for Staircase Removal

The effect is demonstrated in the following image (see *Figure 12.0*).

3.3.0 Charged Pixel Method (CPM)

After implementing these two standard thinning algorithms I start implementing a force-based thinning method into Mathematica. The CPM-algorithm introduced by Ching Y. Suen is based on the idea that dark and white pixels carry an opposite charge and effect each other according to the theory of Coulomb‘s Law:

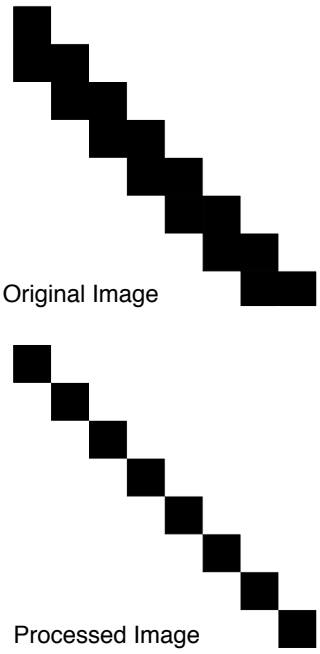


Figure 12.0 Post-Processing Staircase Removal

$$F = \frac{Q_1 Q_2}{4\pi\epsilon R^2} = \mathrm{k}\frac{Q_1 Q_2}{R^2} \qquad (2)$$

Where Q_1 and Q_2 denote the charges of the pixels P_1 and P_2 respectively and R denotes the distance between pixels P_1 and P_2 expressed by the equation:

$$D_{ji} = \sqrt{(\mathrm{P}_{xi} - \mathrm{P}_{xj})^2 + (\mathrm{P}_{yi} - \mathrm{P}_{yj})^2} \qquad (3)$$

Where P_{xi}, P_{xj}, P_{yi} and P_{yj} denote the position of pixel i and
j in the x- and y- direction. For a central pixel the CPM algorithm only considers the pixel‘s 8-neighbours exerting a force on the central pixel. Equations (2) and (3) result in the the following equations describing the force exerted on one pixel. Later on, the value of the force is used as a criterion for the deletion of the pixel. The algorithm consists of four main parts. To decide whether a pixel should be marked for deletion or not it uses the following criteria: The current pixel has to be an edge-point for the algorithm to further consider it for deletion. Furthermore, it cannot be an end-point, its CN has to be equal to one and the net

$$F_{xi} = k \sum_{j=1}^{j=8} Q_j Q_i (\mathrm{P}_{xi} - \mathrm{P}_{xj}) / D_{ji}{}^3 \quad (4)$$

$$F_{yi} = k \sum_{j=1}^{j=8} Q_j Q_i (\mathrm{P}_{yi} - \mathrm{P}_{yj}) / D_{ji}{}^3 \quad (5)$$

force calculated by (4) and (5) must not be zero. Then the pixel will be marked for deletion. The process will be iterated until no pixel is marked for deletion anymore. A process similar to the staircase removal follows (see *3.2.1 Post-Processing Staircase Removal*).

$$\begin{pmatrix} 1 & 0 & x \\ 0 & 0 & 1 \\ x & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} x & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & x \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & x \\ 1 & 0 & 0 \\ x & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} x & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & x \end{pmatrix}$$

Figure 13.0 Post-Processing CPM Templates

If a 3x3 window around a black pixel matches on of the templates in *Figure 13.0* the inner pixel gets marked for deletion.

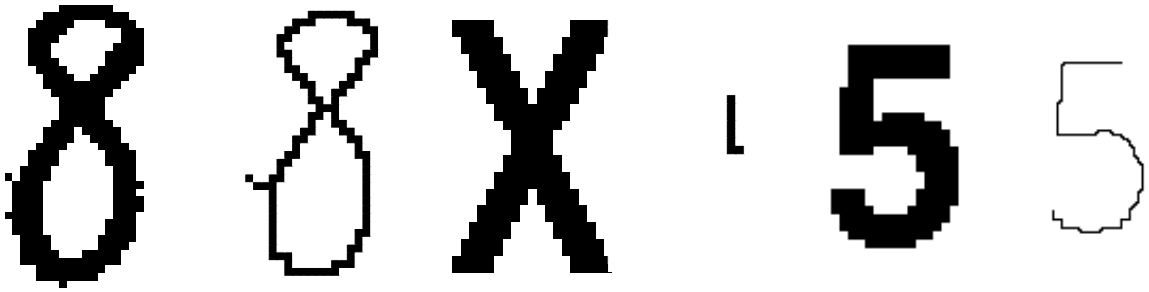
3.4.0 Analysing the Algorithms

By carrying out several test-runs I analyse the algorithms‘ efficiency meaning the time every algorithm takes to return an image, the quality of the image returned and the impact of the pre- and post-processing methods.

4.0 Results

4.1.0 Stentiford Thinning Algorithm

Resulting images of the Stentiford thinning algorithm without pre-processing methods:



1.57s 6,58s 85,27s

Figure 14.0 Stentiford Processed Images

One can observe decent thinning images with the 8-image and the 5-image. In the contrary the thinning of the X-image totally failed. The processed 8-image shows some spurious projection on the left side. Other than that the returned images of the eight and

the five look a little bit rough but this could be due to the low amount of pixels the picture s consist of. The running times are 1.57, 6.58 and 85.27 seconds. See appendix for more example images.

4.1.1 Stentiford Thinning Algorithm With Pre-Processing

Resulting images of the Stentiford thinning algorithm with pre-processing methods:

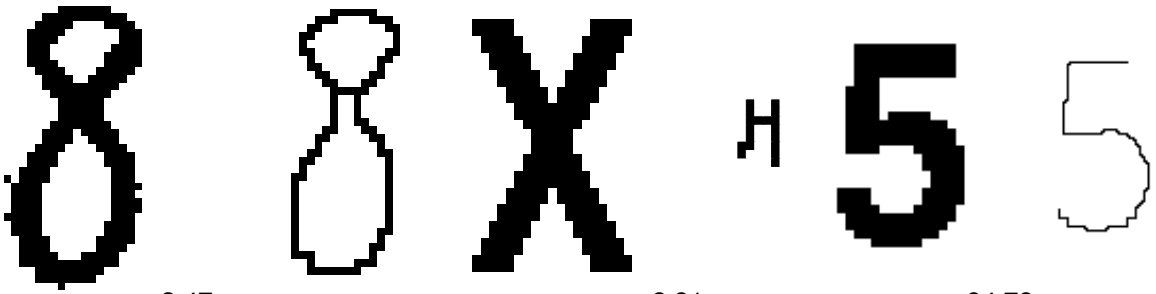


Figure 15.0 Stentifort Processed Images With Pre-Processing

By using the pre-processing methods the spurious projection on the left side of the eight got removed, the processed image of the X looks better than without pre-processing but is still of not acceptable quality. The necking pre-processing does not manage to absolutely prevent necking since there is a small horizontal necking artefact visible inside the eight. The five looks unchanged. The running times are 3.47, 8.81 and 94.78 seconds. See appendix for more example images.

4.2.0 Zhang-Suen Algorithm

Resulting images of the Zhang-Suen algorithm without pre- and post-processing methods:

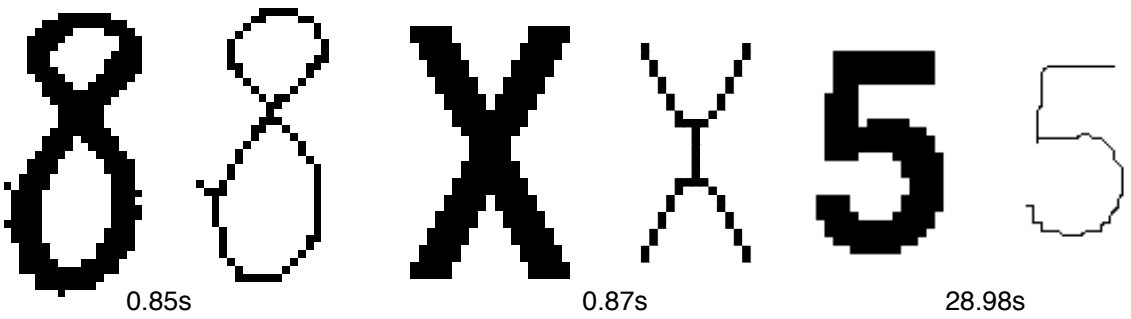


Figure 16.0 Zhang-Suen Processed Images

All three returned images are of decent quality. The image of the eight has a horizontal hair on the left side and the X-image shows strong necking. The 5-image has a small tail on its left central side. The running times are 0.85, 0.87 and 28.98 seconds.

4.2.1 Zhang-Suen Algorithm With Post-Processing

Resulting images of the Zhang-Suen algorithm with staircase removal:

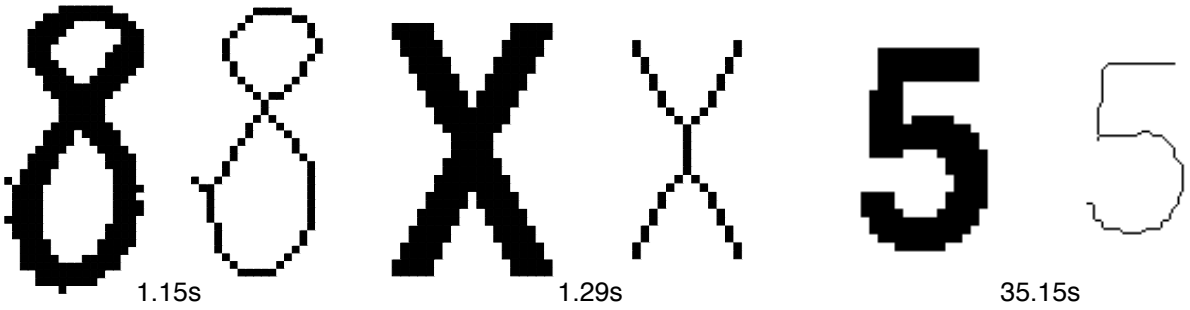


Figure 17.0 Zhang-Suen Processed Images With Staircase Removal

The staircase removal deleted some ”staircase” pixels clearing the resulting images up a bit. The X-image is now symmetric. Spurious projections, necking and tailing still exists. The running times are 1.15, 1.29 and 35.15 seconds.

4.2.2 Zhang-Suen Algorithm With Pre- and Post-Processing

Resulting images of the Zhang-Suen algorithm with Stentiford pre-processing methods and staircase removal:

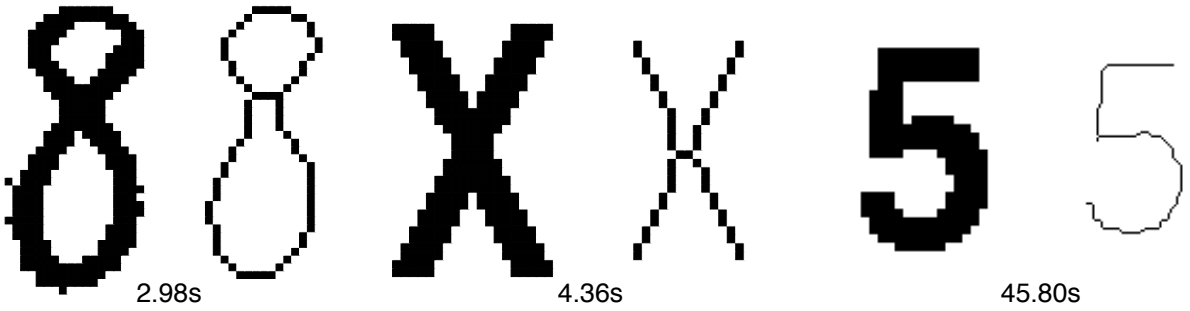


Figure 18.0 Zhang-Suen Processed Images With Pre- and Post-Processing

The necking and the spurious projections got removed almost totally leaving the 8-image with a slight horizontal necking and the X-image nearly with only one central pixel where the different parts of the “X“ touch. The running times are 2.98, 4.36 and 45.80 seconds.

4.3.0 Charged Pixel Method - Algorithm

Resulting images of the CPM algorithm:

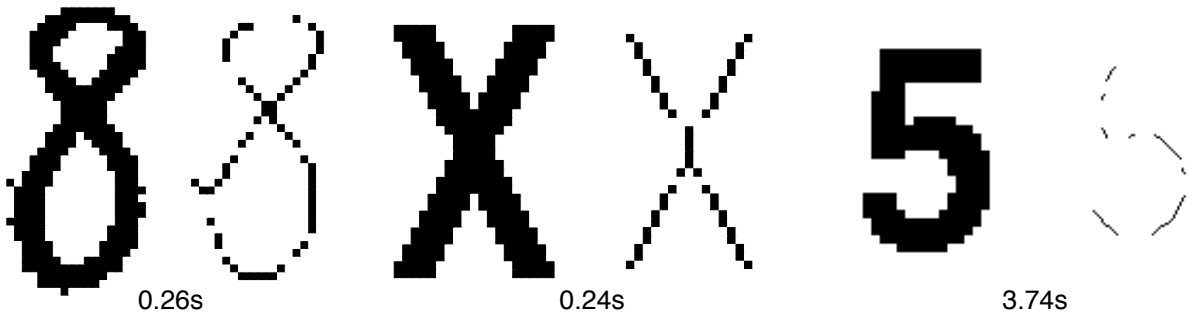


Figure 19.0 CPM Processed Images

The images are much more strongly thinned out. Often the main skeleton is cut of and has gaps but at least for the 8-image and the X-image the structures are well recognisable. The processed 5-image misses some essential pixels. The same spurious projection in the processed 8-image and some necking in the X-image are visible. The running times are 0.26, 0.24, and 3.74 seconds.

4.4.0 Pre- and Post-Processing Methods Only

These are the resulting images of the **spurious projections** pre-processing algorithm with using different connectivity types. In general, the 8-neighbours connectivity type is used.

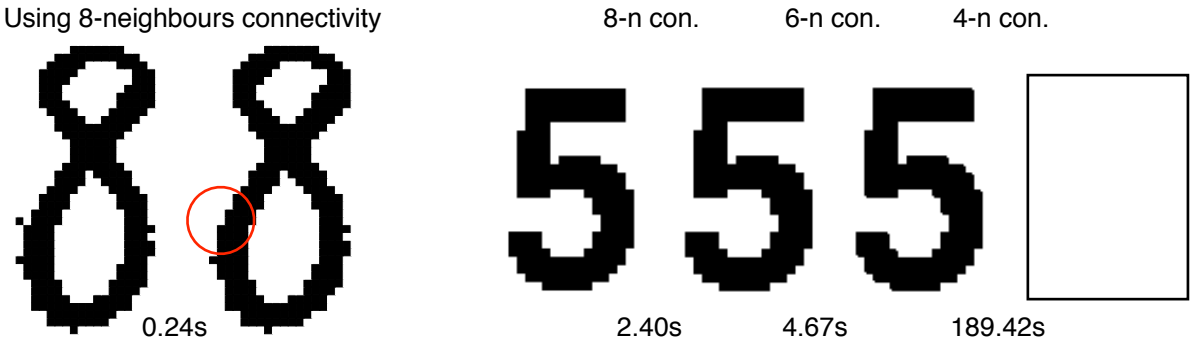


Figure 20.0 Spurious Projection Pre-Processing

One can slightly see the effect of the projections pre-processing method removing one pixel in the 8-image using the 8-neighbours connectivity. For the 5-image I use the three different types of connectivity to emphasise their importance. The pre-processing with 8-neighbours connectivity does not change anything. Whereas, using the 6-neighbours connectivity the algorithms slightly smoothes some corners and with the 4-neighbours connectivity the algorithm simply returns a totally blank image having removed all black pixels. For 8-neighbour connectivity the running times are 0.24 and 2.40 seconds. For 6-neighbour connectivity the running time is 4.67 seconds in this case and for 4-neighbour connectivity 189.42 seconds.

Resulting images of the **necking** pre-processing algorithm:

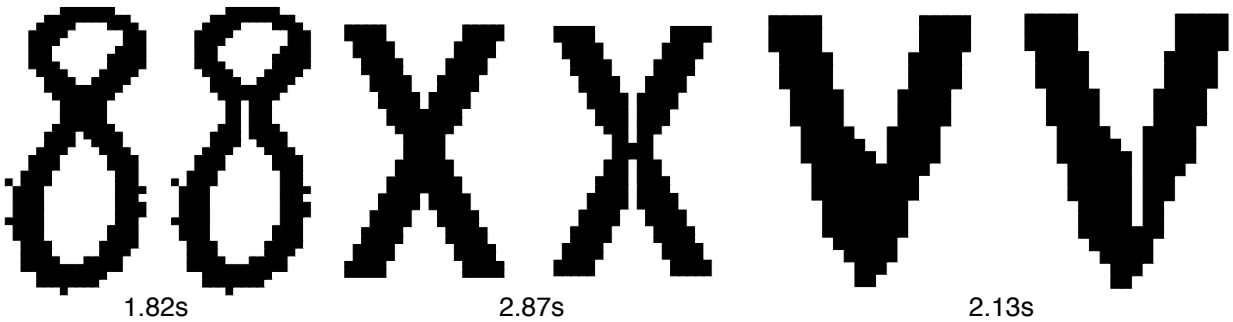


Figure 21.0 Necking Pre-Processing

In every image the cut-in at a acute angle pixel caused by the pre-processing method is clearly distinguishable. This can help preventing necking often caused by acute angles when thinned by an algorithm as explained earlier (see *3.1.1 Pre-Processing*). The running times are 1.82, 2.87 and 2.13 seconds.

Resulting images of the **staircase removal** post-processing algorithm:

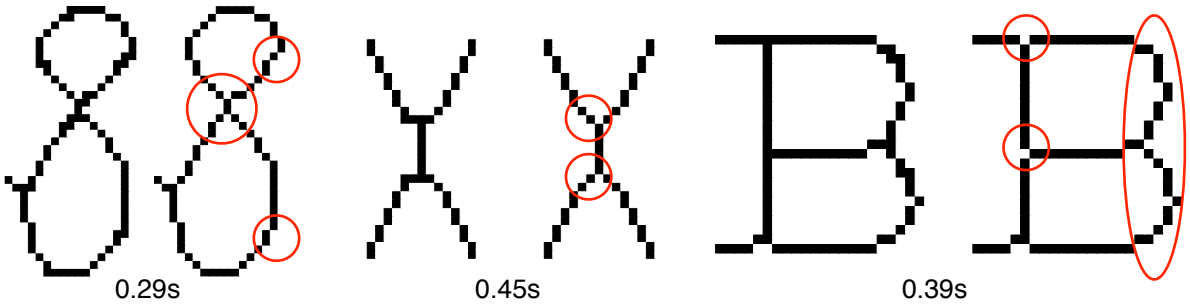


Figure 22.0 Staircase Removal Post-Processing

With red marks I indicate where the staircase removal algorithm removes redundant pixels. Those changes are clearly visible. The running times are 0.29, 0.45 and 0.39 seconds.

5.0 Analysis and Discussion

In the following I analyse, list and compare the strength and weaknesses of the three thinning algorithms putting the work done into perspective and suggest several starting points and ideas for further work that could be done to possibly improve the examined algorithms.

5.1.0 Strength and Weaknesses

5.1.1 Stentiford Thinning Algorithm

The work done very well demonstrates that the quality of the images returned by the Stentiford algorithm is reasonable for some simple objects and unacceptable for others (see *Figure 14.0)*. While the image quality of this algorithm can be improved significantly by using the two suggested pre-processing methods (see *Figure 15.0*), it nonetheless stays mediocre to bad in general, because the algorithm returns at least one image in which the original shape of the object is drastically changed making the skeleton of the original picture almost non-recognisable (X-image). Furthermore, minor cases of necking are still present within the 8-image. The quality of other images processed is fine. Looking at the raw Stentiford algorithm and the one including pre-procession methods one can see that in both cases the average running time of 5.32ms per pixel without pre-procession and 7.07ms per pixel with pre-procession between the three pictures is extremely high. This makes this version of the Stentiford algorithm almost useless in application as running times would rise insanely for slightly more complex images.

5.1.2 Zhang-Suen Algorithm

The pure Zhang-Suen algorithm without pre- and post-procession returns all test-images in medium quality by thinning down to the right skeleton in every case keeping the original shape recognisable or at least guessable. Artefacts like spurious projections and necking are visible in some cases (see *Figure 17.0*). The running times are fine with an average running time of approximately 1.32ms per pixel between the three presented test images.

When the staircase removal algorithm is appended to the main algorithm, image quality gets improved without exception coping with a running time detriment of an average 0.41ms per pixel between the three test images. This results in a total of 1.73ms per pixel average running time. Thus, the staircase removal is very effectively improving the performance of the Zhang-Suen algorithm and well implemented.
When the Stentiford pre-processing methods are added to the Zhang-Suen algorithm most of the ”hairs“ and necking artefacts are removed. Though, in some cases slight artefacts are created like the short horizontal necking in the 8-image (see *Figure 18.0).* Therefor, these pre-processing methods are not recommended for the Zheng-Suen algorithm unrestrictedly. Even more so, if the additional running time is considered. The average running time for the Zhang-Suen algorithm with pre- and post-processing is 3.83ms per pixel, which is remarkably longer than without with a detriment of 2.1ms average running time per pixel. That is more than double the initial average running time per pixel!

5.1.3 Charged Pixel Method

Significantly different results demonstrates the CPM algorithm by very radically deleting pixels partly destroying the skeletons of the original shapes. Interestingly, the original structure stays guessable even if the thin line of the image skeleton is broken at some or even many points (see *Figure 19.0*). From a technical point of view the images are of very bad quality since the algorithm does not conserve the inner and important coherent main structure of the shape. Some unwanted artefacts are showing. The running time is relatively short with 0.27ms average running time per pixel.

5.2.0 Comparing the Algorithms

Overall, the Zheng-Suen algorithm with staircase removal implemented shows the best results and performance. Depending on individual cases the Stentiford pre-processing methods should be added. The algorithm carries out relatively fast and reliable thinning of simple binary images. The information collected are not enough for giving a good approximation of how well these algorithms will do with larger images. Letting the Zheng-Suen algorithm run over a larger image with around $1x10^6$ pixels would result in a running time of at least 22min under the reckless assumption that the average running time per pixel is constant in all variables except image size, which would underestimate the relation immensely.
Regarding the potential for improvement the CPM algorithm seems to be the most promising. Even if technically the results are by far not acceptable, for a human on the other hand or an advanced neural network a complex image could be reduced in image size but stay recognisable. Furthermore, additional conditions for deletion of a pixel can prevent the algorithm from deleting connecting pixels. The shortest average running time per pixel of all the three algorithms and even the pre- and post-processing methods makes the algorithm interesting and it makes room for further changes which could improve image quality while reducing efficiency.
The Stentiford algorithm looses against the Zhang-Suen algorithm in both aspects, quality and speed. Comparing templates to the 8-neighbours of a central pixel does not seem to be the fastest solution for determining pixel deletion. The pre-processing methods improve the algorithm but themselves take additional time which further slows down the algorithm. Thus, the Stentiford algorithm drops out of the competition.

5.3 Improving the Algorithms: Possible Starting Points, Ideas

- **Hole Removal:** Even though in none of all the test images was a case where a small embedded hole inside the object was causing a major flaw in the thinning process it could be necessary to implement the hole removal pre-processing method as original introduced by F. W. M Stentiford where white holes are filled in. The holes are identified according to the templates in *Figure 23.0* and marked for removal when matching the templates in *Figure 24.0.*

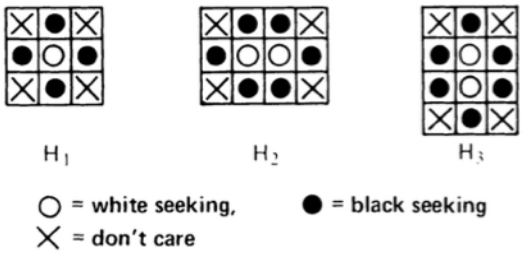


Figure 23.0 Identifying Templates

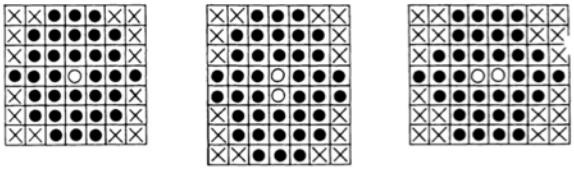


Figure 24.0 Filling In Criterion Templates

- **Connection Point Removal Prevention:** One should definitely try to improve the CPM-algorithm either by increasing the distance from which pixels are said to be influenced by other pixels or change the algorithm in that way that it does not remove important connection points.
- **6-neighbour connectivity:** It could lead to some interesting results to try implementing different connectivity types into the algorithms using the connectivity number as a criterion. As shown in *Figure 20.0* different connectivity types can smooth out the edges for images with a comparably high pixel density and could have a large impact on results.
- **Efficiency:** A in-depth analysis of the running times is recommended. This would allow a a targeted approach for improving the efficiency of the algorithms possibly making them viable for future application or further improvement.
- **Zhang-Suen Algorithm:** Even though the Zhang-Suen algorithm is a lot faster than the Stentiford algorithm its efficiency could very easily be improved further by removing the need for sub-iterations within the algorithm by enlarging the window and thus considering edge-points information of the central pixels neighbours.

6.0 Bibliography

Stentiford, F. W. M. and R.G Mortimer, ”Some New Heuristics for Thinning Binary Handprinted Characters”, OCR, 1984

Holt, Christpher M et al., ”An Improved Parallel Thinning Algorithm”, Communications of the ACM, Frbruary 1987

Suen, C. Y. And Akila Arumugam and T. Radhakrishnan, ”A Thinning Algorithm Based On The Force Between Charged Particles”, World Scientific, 1994

Sreejini, K.S. And V.K. Govidan, ”Improved multiscale matched filter for retina vessel segmentation using PSO algorithm”, National Institute of Technology, Calicut, Kerala, India, 9th of October 2014

University of Tartu, ”Digital Image Processing”, University of Tartu, January 2014

Khare Ashish, ”Image Pre-Processing”, Ashish Kumar, Dec 6 2012

Open Source, ”Topological Skeleton”, Wikipedia, 16 February 2018

Ferrer Romulo et al., ”Techniques of Binarization, Thinning and Feature Extraction Applied to a Fingerprint System”, International Journal of Computer Science, Oct 2014

7.0 Appendix

All used code is available in the logbook under the last section: ”Final Versions“.

7.1 Further Processed Test Images

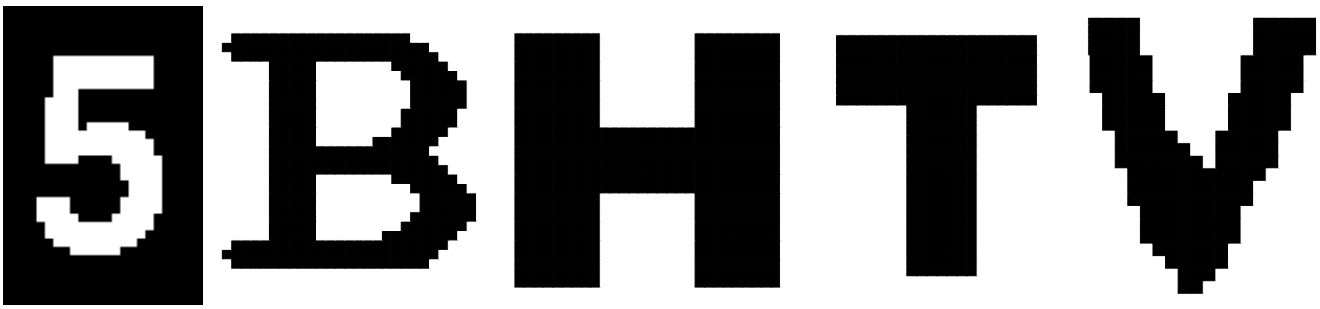


Figure 25.0 Original Images

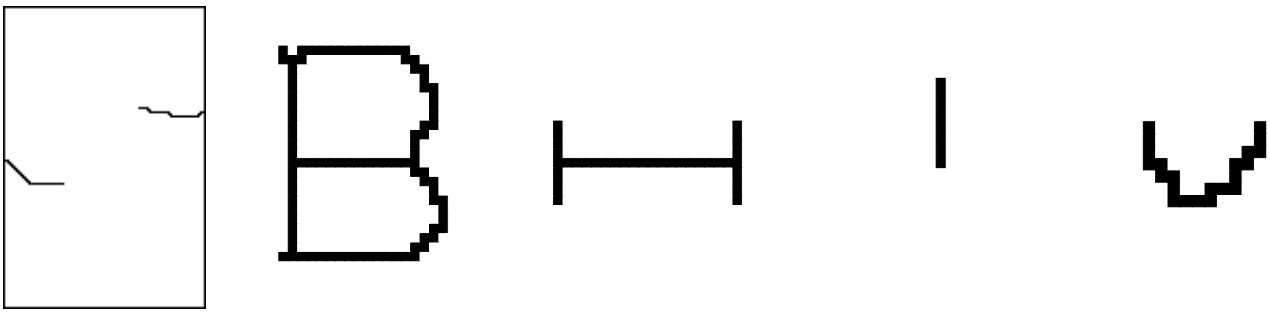


Figure 26.0 Stentiford

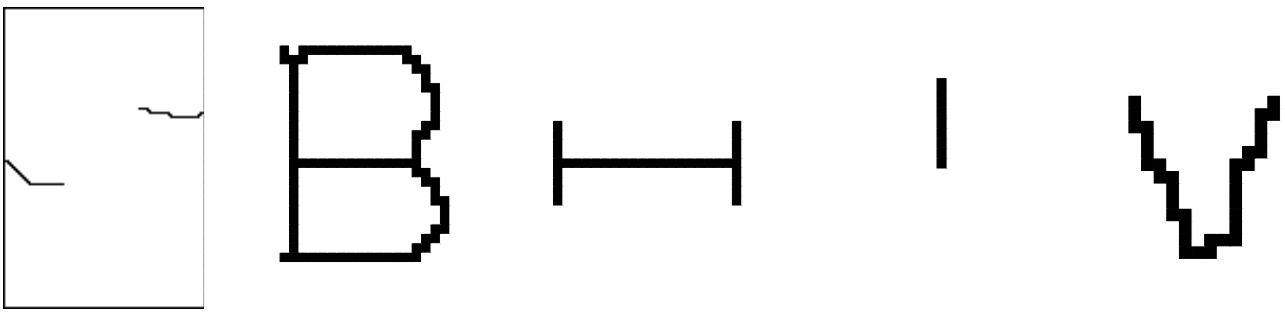


Figure 27.0 Stentiford With Pre-Processing

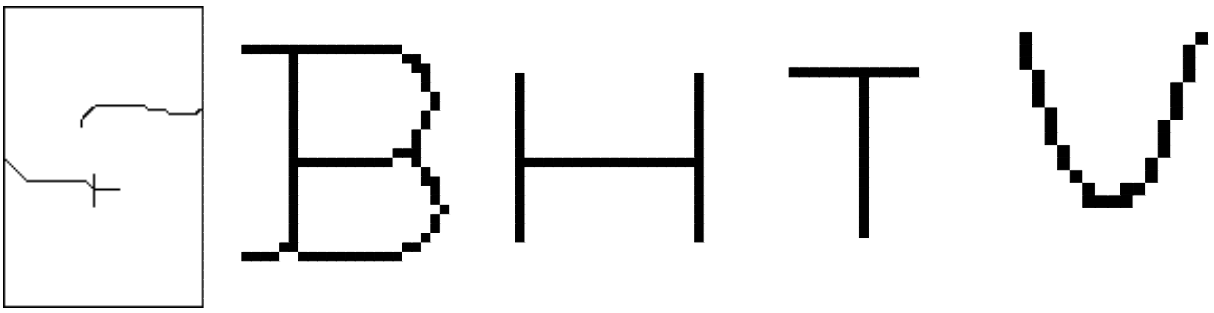


Figure 28.0 Zhang-Suen

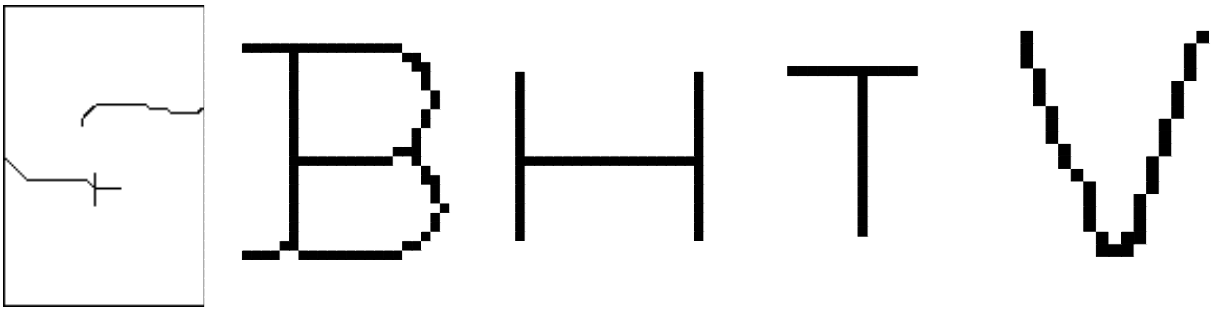


Figure 29.0 Zhang-Suen With Pre-Processing Methods

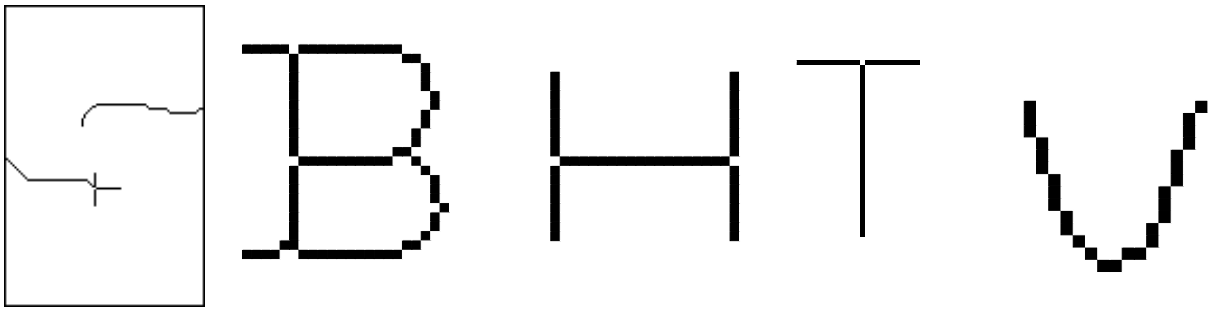


Figure 30.0 Zhang-Suen With Staircase Removal

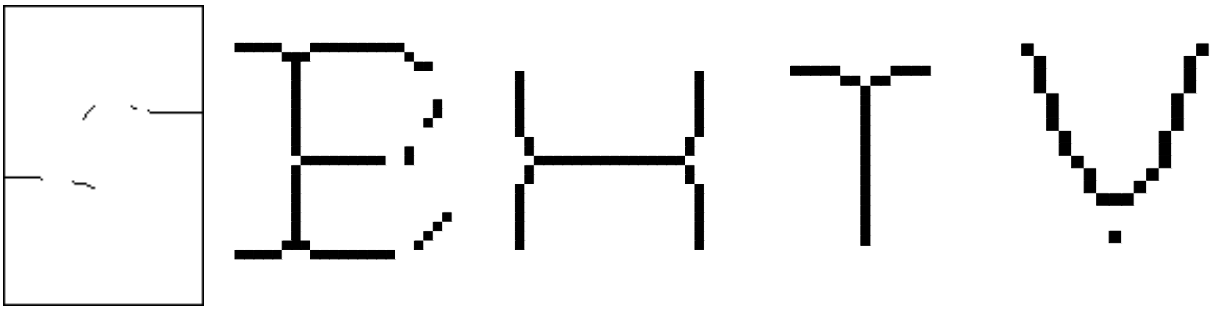


Figure 31.0 CPM Algorithm

Some Examples of Images Without Specific Object

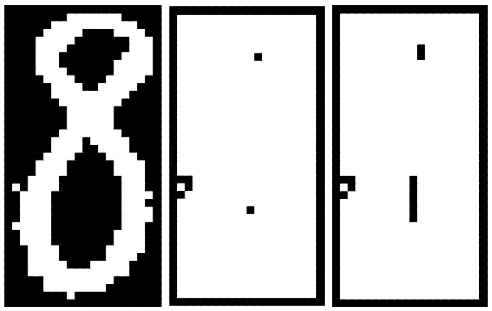


Figure 32.0 Left to Right : Original, Zhang-Suen,Stentiford

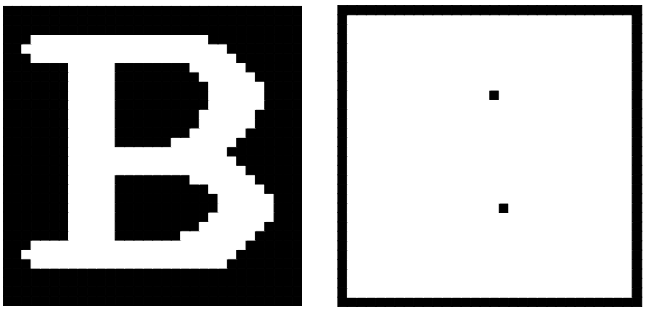


Figure 33.0 Stentiford with Pre-Processing Methods

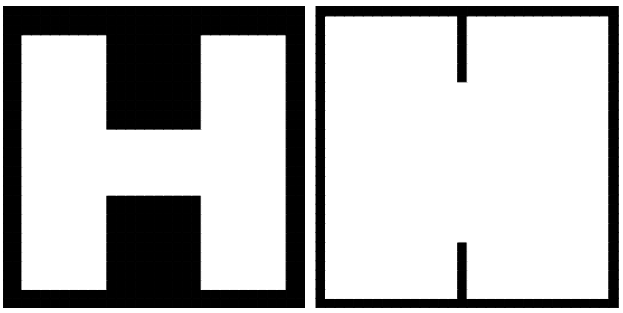


Figure 33.0 Stentiford

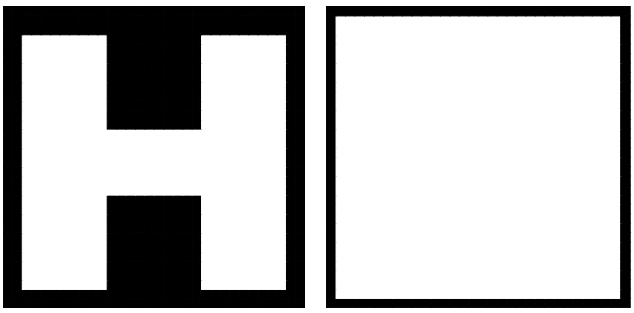


Figure 34.0 CPM Algorithm

Necking and Spurious Projections

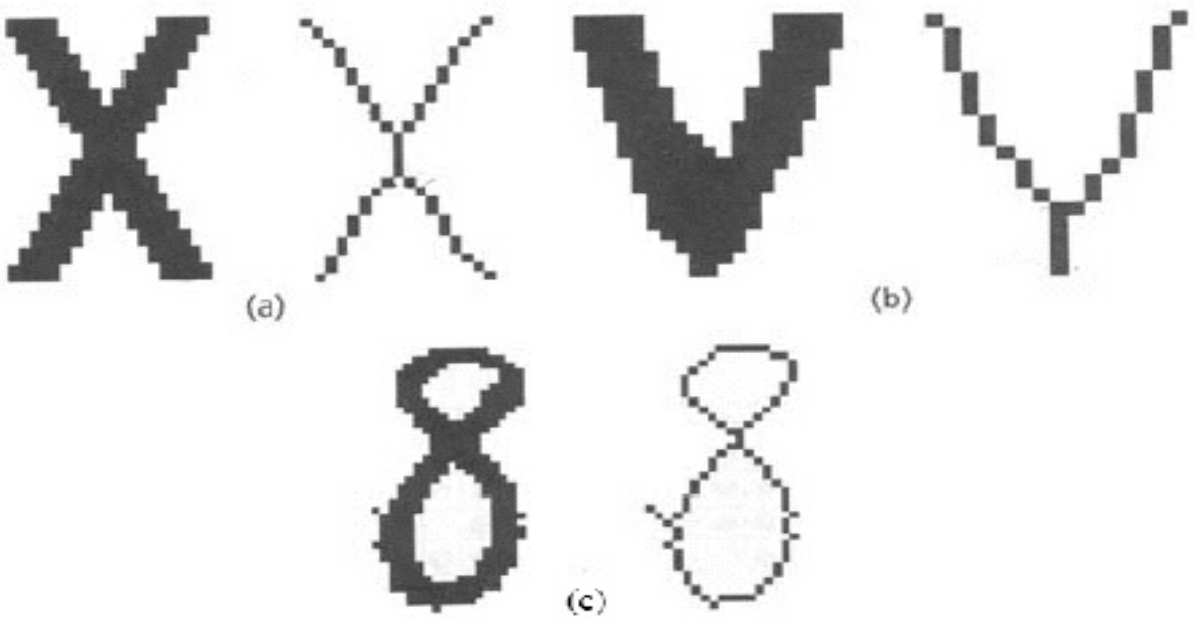


Figure 35.0 Artefacts: a) Necking, b) Tailing,
c) hairs/spurious projections