Skeletonization of 2D Binary Images

David Eberly, Geometric Tools, Redmond WA 98052
https://www.geometrictools.com/

Created: June 7, 2001
Last Modified: March 2, 2008

Contents

1 Introduction

This document describes a simple algorithm for creating skeletons of binary blobs in 2D images. The blob pixels are 1-valued and the background pixels are 0-valued. The goals of the algorithm are to preserve the number of connected components in the image and to preserve the number of holes in the image.

Various definitions for pixels are important in the construction. The *interior* of a blob is defined using 4-connectedness. A 1-pixel at (x, y) is an *interior pixel* if its four neighbors at $(x+1, y)$, $(x-1, y)$, $(x, y+1)$, and $(x, y-1)$ are 1-pixels. The *boundary* of a blob is defined using 8-connectedness. A pixel is a *boundary pixel* if it is not an interior pixel and at least one of its eight neighbors is a 0-pixel.

As a whole, a blob can be viewed as an abstract undirected graph whose vertices are the 1-pixels and whose edges are defined by 8-connected adjacency. That is, if (x_0, y_0) and (x_1, y_1) are distinct 1-pixels with $|x_1 - x_0| \leq 1$ and $|y_1 - y_0| \leq 1$, then $\langle (x_0, y_0), (x_1, y_1) \rangle$ is an edge in the graph. A graph is *connected* if any two of its vertices are connected by some path through the graph. A vertex of a connected graph is an *articulation point* if its removal from the graph causes the graph to become disconnected. For the purposes of skeletonization, whether or not a 1-pixel is removed is based on the local behavior of the graph at that pixel. Of particular importance is a 1-pixel that is called a *local articulation point*. This is a pixel such that if it is removed from the graph, the 3×3 neighborhood of that pixel becomes disconnected. Observe that an articulation point is necessarily a local articulation point, but the converse is not true as the following diagram illustrates.

	$x-2$	$x-1$	x	$x+1$	$x+2$
$y-2$	0	1	1	1	0
$y-1$	0	1	0	1	0
y	0	0	1	0	0
$y+1$	0	0	0	0	0

The pixel (x, y) is a local articulation point. If it were to be removed from the graph, its 3×3 neighborhood becomes disconnected into two single-point components, the pixel at $(x-1, y-1)$ and the pixel at $(x+1, y-1)$. However, the full graph in the diagram is connected. The removal of (x, y) does not disconnect the full graph since $(x-1, y-1)$ is connected to $(x+1, y-1)$ via the pixel $(x, y-2)$.

2 Characterization of Local Articulation Points

The definition might lead one to apply a standard connected component labeler to the subgraph in the 3×3 neighborhood of (x, y) once (x, y) is removed. However, given the size of the graph, a faster method involves using a look-up table. The binary values of the eight neighbors of (x, y) can be encoded in an 8-bit index for the table. Of course, the table has 256 elements. The bit assignments are b_i for $0 \leq i \leq 7$ are shown below:

b_0	b_1	b_2
b_7	$*$	b_3
b_6	b_5	b_4

The lookup table has a value of 1 if and only if the pixel is a local articulation point. The table is

0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1	0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1	0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1	0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1	0	1	0	0	0	1	0	0
0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0
1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0
1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0

where the upper left corner corresponds to index 0 and the indices increase from left to right, then top to bottom. For example, notice that the first 1 encountered in the first row of the table is at index 5. This corresponds to the pixel (x, y) shown earlier as an example of a local articulation point. The bits b_0 and b_2 are set to 1 since the corresponding pixels at $(x-1, y-1)$ and $(x+1, y-1)$ are 1, all other bits being zero. The byte index 00000101 is the decimal number 5.

3 Large Scale Thinning

The candidate pixels for being removed are obviously the boundary pixels. But there is more to the story. A 1-pixel-thick horizontal straight line is already a skeleton of itself (according to anyone's definition for skeleton). By definition, all pixels on that line are boundary pixels, but none of them are removed to obtain the skeleton. Somewhat more complicated is the example of a 2-pixel-thick horizontal straight line. Some decision must be made about which pixels should be removed. An obvious choice is to remove the top row of pixels, but it is not clear that a greedy approach is always the best one.

The approach I take is to apply a *large scale thinning* in an iterative manner that determines which pixels are interior ones before making the decision on which boundary pixels to remove. Thus, the method is only partly greedy. After interior pixels are tagged as such, boundary pixels are removed one by one, using a top-to-bottom row scan, as long as they are (1) adjacent in the 8-connected sense to an interior pixel and (2) not a local articulation point. Multiple passes are made using this algorithm until (1) no more interior pixels

exist or (2) interior pixels exist yet no more boundary pixels can be removed. The diagram below shows the original binary image (on the left) with background pixels marked as periods for readability. The middle image shows the interior pixels marked with 2, the boundary pixels with value 1. The right image shows the image after boundary pixels are removed in a greedy manner (scan left to right, then top to bottom).

```
11111111.............. 11111111.............. 111..................
111111111............. 111122221............. 11112222.............
....111111............ ....122221............ .....2222............
.....111111........... .....112221........... .......222...........
.......11111.......... .......12221.......... ........222..........
.......11111.......... .......12221.......... ........222..........
........11111......... ........12221......... .........222.........
.........1111......... .........1221......... ..........22.........
.........1111......... .........1221......... ..........22.........
.......111111......... .......112221......... .........222.........
.....11111111111...... .....11222222111...... .......222222........
....111111111111111.... ....122222222222111.... .....22222222222....
....11111111111111111.. ....12222222211222211.. .....22222222..2222....
...1111111111..11111111 ...1222222221..11111211 ....22222222.......12.1
...1111111111.......111 ...1222212221.......111 ....2222.222.........11
...11111.1111......... ...12221.1221......... ....2221..22..........
...1111.11111......... ...1221.12221......... ....221.1222..........
..1111.11111.......... ..1221.12221.......... ...22..1222...........
..1111.11111.......... ..1221.12221.......... ...22...222...........
..111111111........... ..122212221........... ...222.222............
..11111111............ ..12222221............ ...222222............
111111111............. 112222221............. 1.222222.............
11111111.............. 11111111.............. 11...................
```

The next diagram illustrates a repeat of the process. The left image is the the right image from the previous diagram with all the 2's replaced by 1's.

```
111..................... 111..................... 111.....................
11111111................ 11111111................ 111111..................
.....1111............... .....1121............... .....112................
.......111.............. .......121.............. ........2...............
........111............. ........121............. ........2...............
........111............. ........121............. ........2...............
.........111............ .........121............ .........2............
..........11............ ..........11............ ..........1............
..........11............ ..........11............ ..........1............
.........111............ .........121............ .........2............
.......111111........... .......112221........... .........222............
.....11111111111........ .....11222222111........ .......222222111.......
.....11111111..1111..... .....12222221..1111..... ......222222...1111....
....11111111.......11.1 ....12221221.......11.1 .....222.22........11.1
....1111.111.........11 ....1221.121.........11 .....22...2..........11
....1111..11............ ....1221..11............ .....221...1............
....111.1111............ ....111.1121............ .....11.1.2............
...11..1111............. ...11..1221............. ...11..122.............
...11...111............. ...11...121............. ....1....2.............
...111.111.............. ...121.121.............. ....2...2..............
...111111............... ...122121............... ....22.2...............
1.111111................ 1.111111................ 1.11..1................
11...................... 11...................... 11......................
```

Another pass.

```
111..................... 111..................... 111.....................
111111.................. 111111.................. 111111..................
.....111................ .....111................ .....111................
........1............... ........1............... ........1...............
.........1.............. .........1.............. .........1..............
.........1.............. .........1.............. .........1..............
..........1............. ..........1............. ..........1.............
...........1............ ...........1............ ...........1............
...........1............ ...........1............ ...........1............
..........1............. ..........1............. ..........1.............
.........111............ .........121............ ..........2............
.......111111111........ .......112221111........ .........2221111.......
......111111...1111..... ......121221...1111..... .......2122....1111....
.....111.11........11.1 .....121.11........11.1 ......2...1........11.1
.....11...1..........11 .....11...1..........11 ......1...1..........11
.....111...1............ .....121...1............ ......21...1............
.....11.1.1............. .....11.1.1............. .....11.1.1.............
...11..111.............. ...11..111.............. ...11..111..............
....1....1.............. ....1....1.............. ....1....1..............
....1...1............... ....1...1............... ....1...1...............
....11.1................ ....11.1................ ....11.1................
1.11..1................. 1.11..1................. 1.11..1.................
11...................... 11...................... 11......................
```

Another pass.

```
111..................... 111..................... 111.....................
111111.................. 111111.................. 111111..................
.....111............... .....111............... .....111...............
........1.............. ........1.............. ........1..............
.........1............. .........1............. .........1.............
.........1............. .........1............. .........1.............
..........1............ ..........1............ ..........1............
...........1........... ...........1........... ...........1...........
...........1........... ...........1........... ...........1...........
..........1............ ..........1............ ..........1............
..........1............ ..........1............ ..........1............
.........1111111....... .........1211111....... ..........211111.......
.......1111....1111.... .......1111....1111.... .......111.....1111....
......1...1........11.1 ......1...1........11.1 ......1...1........11.1
......1...1..........11 ......1...1..........11 ......1...1..........11
......11...1........... ......11...1........... ......11...1...........
.....11.1.1............ .....11.1.1............ .....11.1.1............
...11..111............. ...11..111............. ...11..111.............
....1....1............. ....1....1............. ....1....1.............
....1...1.............. ....1...1.............. ....1...1..............
....11.1............... ....11.1............... ....11.1...............
1.11..1................ 1.11..1................ 1.11..1................
11...................... 11...................... 11......................
```

The final result after all pixels are relabed to 1 is shown below.

```
111.....................
111111..................
.....111...............
........1..............
.........1.............
.........1.............
..........1............
...........1...........
...........1...........
..........1............
..........1............
..........111111.......
.......111.....1111....
......1...1........11.1
......1...1..........11
......11...1...........
.....11.1.1............
...11..111.............
....1....1.............
....1...1..............
....11.1...............
1.11..1................
11......................
```

The stopping criterion in this example is that there are no more interior pixels. The other stopping criterion that was mentioned earlier is that there are interior pixels, but no boundary pixels can be removed. The only way this is possible is if every boundary pixel is a local articulation point. This is a situation where the small scale information is effectively noise that affects the larger scale aspects of the blob. The following example illustrates how pathological the problems can be. The interior pixels are marked by 4, the boundary pixels are marked by 1, and the background pixels are marked by 0.

$$\begin{array}{ccccc} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{array}$$

Effectively this examples shows lots of single-pixel-sized "holes" surrounded by the 1-pixels. You can imagine taking this type of configuration and scattering it throughout an image to create a very large image with lots of interior pixels, but the image cannot be trimmed. Potentially the worst case is a checkerboard image where (x, y) pixels are 1 for $x+y$ even and 0 for $x+y$ odd except for an arbitrary set of odd-sum components that are set to 1.

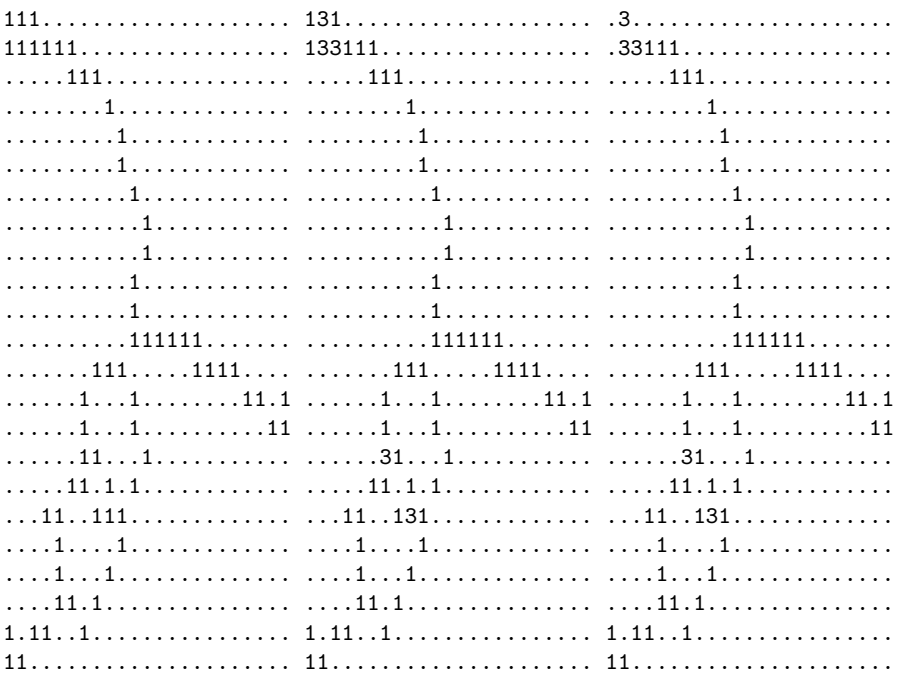
If the number of single-pixel-sized holes is expected to be small, a preprocess of the image can be made to fill them with 1-pixel values, effectively a dilation except that the "outermost" 1-pixels are not expanded.

4 Smaller Scale Thinning

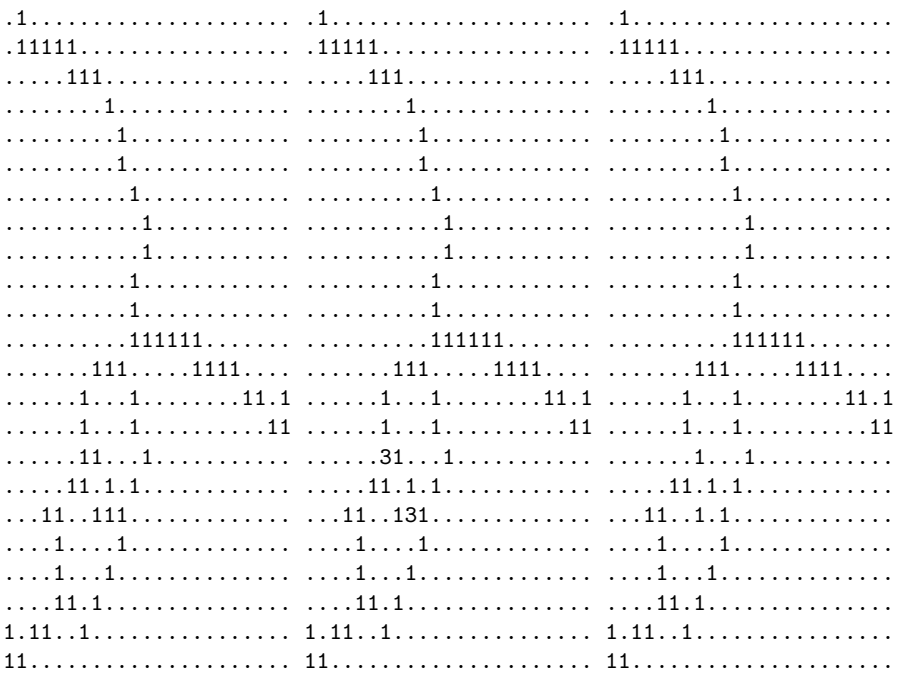
In the large image example shown earlier, the large scale thinning was applied until the thinned binary blob had no more interior pixels. If skeleton thickness is not an issue, you can stop here and accept this set as the skeleton. On the other hand, many applications desire a 1-pixel thick skeleton that can be segmented into disjoint polylines. The interiorless thinned blob must be reduced even further.

The simplest way to do this is to repeat the large scale thinning algorithm, but by marking all those pixels with exactly three adjacent neighbors in the 4-connected sense (only north, south, east, and west neighbors are counted). Call these points 3-interior points. After marking such pixels, a pass is made to remove boundary pixels that are adjacent to the marked pixels. Essentially the thinned blob is not thick enough to have a true interior point (a 4-interior point), but it is almost thick enough because it has the 3-interior points. As in the large scale thinning, multiple passes are made using this algorithm until (1) no more 3-interior pixels exist or (2) 3-interior pixels exist yet no more boundary pixels can be removed. If condition (2) is the one that stops the process, it is safe to remove any remaining 3-interior pixels (using a greedy removal) because, by definition, they cannot be local articulation points.

Continuing with the previous example, the diagram below shows the image resulting from the large scale thinning (image on the left). The middle image shows the marked pixels. The right image shows the image after boundary pixels are removed in a greedy manner (scan left to right, then top to bottom).



The next pass yields 3-interior points, but no boundary pixels can be removed. Stopping condition (2) is met and the remaining 3-interior pixels are removed in a greedy manner. The right image below is the result of that removal.



Although in this example, all 3-interior pixels were removed, the following shows a simple situation where neither 3-interior pixel is a local articulation point *until* the other one is removed. The removal must check for local articulation. The original image is on the left with the 3-interior pixels marked. The middle image is the left image with the first 3-interior pixel removed. The other 3-interior pixel becomes a local articulation point, so it cannot be removed. The right image is the final result of the removal.

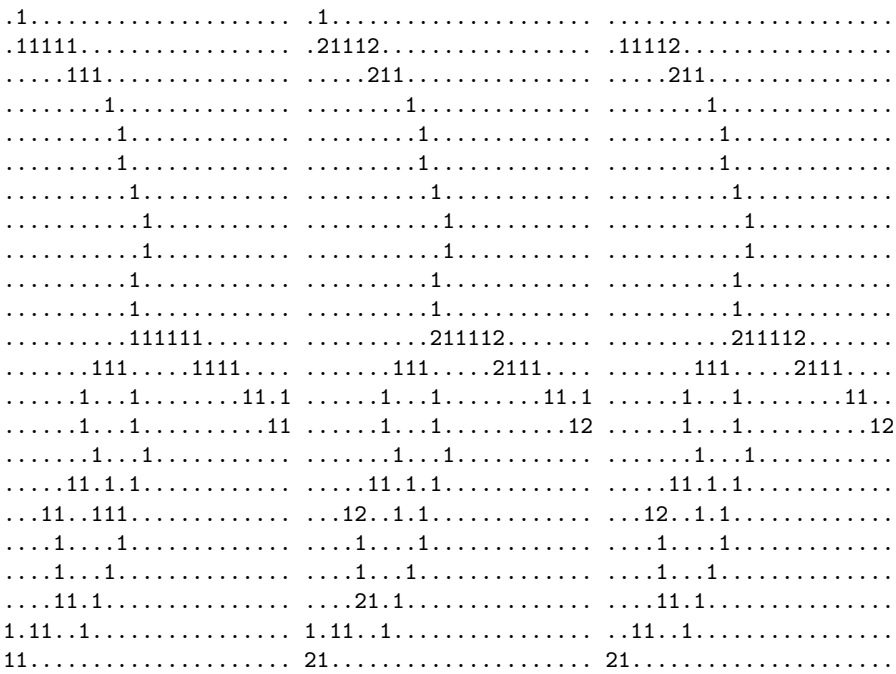
$$\begin{matrix} 0 & 1 & 0 \\ 0 & 3 & 1 \\ 1 & 3 & 0 \\ 0 & 1 & 0 \end{matrix} \quad \rightarrow \quad \begin{matrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 3 & 0 \\ 0 & 1 & 0 \end{matrix} \quad \rightarrow \quad \begin{matrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{matrix}$$

5 Smallest Scale Thinning

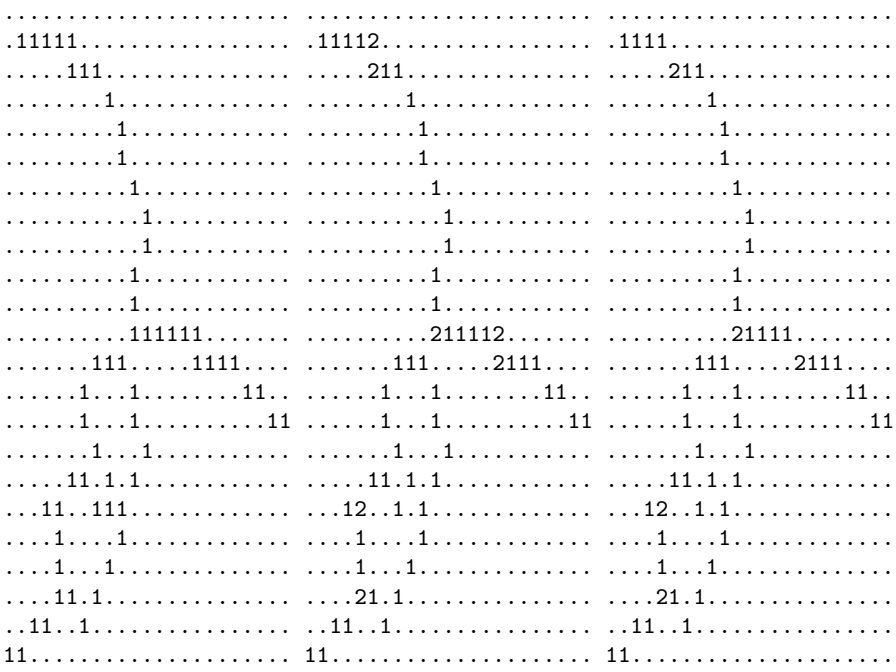
The final thinned blob after the two thinning operations may still not be 1-pixel thick (if possible). In the event that no more 4-interior or 3-interior pixels exist, there may be what I call 2-interior pixels. These pixels have one neighbor in the north/south direction *and* one neighbor in the east/west direction. Such a pixel forms an "L" with the two neighbors.

An algorithm identical to the 3-interior processing may be applied for the 2-interior case. The 2-interior pixels are marked. Boundary pixels that are adjacent to 2-interior pixels are removed as long as they are not local articulation points. Multiple passes are made until (1) no more 2-interior pixels exist or (2) 2-interior pixels exist yet no more boundary pixels can be removed.

Continuing with the previous example, the diagram below shows the image resulting from the large scale thinning (image on the left). The middle image shows the marked pixels. The right image shows the image after boundary pixels are removed in a greedy manner (scan left to right, then top to bottom).



The next pass yields 2-interior points, but no boundary pixels can be removed. Stopping condition (2) is met and the remaining 2-interior pixels are removed in a greedy manner. The right image below is the result of that removal.



The diagram below shows the original image and the final 1-pixel thick skeleton.