HW7 Report

Yu Xiang Luo, B10902037

November 7, 2023

1 Introduction

This report describes a Python script developed to perform image thinning using the morphological thinning algorithm. The script applies iterative conditional marking to reduce the thickness of the shapes in a binary image to a single pixel width.

2 Algorithm Description

The script executes an image thinning algorithm, which is a morphological operation that is used to remove selected foreground pixels from binary images. It iteratively erodes away the boundaries of regions of foreground pixels (usually white on a black background) until only minimal pixel width lines remain.

3 Code Implementation

The code is written in Python, using the Pillow library for image manipulation. The algorithm consists of several functions:

- `ds(image, sfac)`: Downsamples the image by a scaling factor.
- `get_np(image, position)`: Retrieves a 3x3 neighborhood of pixels around a specified position.
- `hf(b, c, d, e)`: Applies a condition function based on pixel values.
- `ff(a1, a2, a3, a4)`: Determines the action based on conditions returned by `hf`.
- `get_yn(image)`: Generates a matrix based on conditions applied to each pixel.
- `get_im(yn)`: Creates an image from the condition matrix.
- `dilation(image)`: Performs the dilation operation on the image.
- `get_tm(image, yn, dm)`: Applies thinning to the image.

3.1 Code Snippet

Below is a snippet from the script showing the main loop of the thinning process:

```
# Main loop of the thinning process
while True:
    yn = get_yn(tm)
    im = get_im(yn)
    dm = dilation(im)
    tmp = get_tm(tm, yn, dm)
    if ImageChops.difference(tmp, tm).getbbox() is None:
        break
    tm = tmp
    Iter += 1
    print(f'Iteration: {Iter}')
tm.save(f'./img/thinning{Iter}.bmp')
```

4 Results

The algorithm was applied to a binary image, and the process was iterated until no further changes were observed. The final result is a thinned version of the original image, with all objects reduced to single-pixel width.

4.1 Thinning Process Image

Below is the thinned image after the final iteration of the algorithm:

Figure 1: The binary image after the thinning process.

5 Conclusion

The implemented thinning algorithm successfully reduces the objects in a binary image to single pixel width lines. This can be particularly useful for applications in pattern recognition, image processing, and computer vision where the structure of objects is to be analyzed without the influence of varying object thickness.