## CSE585/EE555:  Digital Image Processing II

## Computer Project # 2:

## Homotopic Skeletonization & Shape Analysis

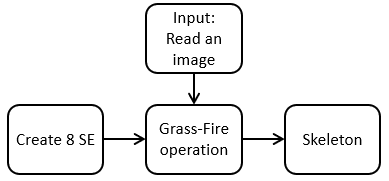
#### Hao Zhou, Fan Yang, Zerui Hua

#### Date: 02/16/2023

* + 1. **Objectives**

1. Learn how to use MATLAB for digital images processing;
2. Learn how to use simple morphological operations (erosion / dilation) to retrieve skeleton and analyze shape;
   * 1. **Methods**
3. Algorithm and Theory (Homotopic Skeletonization, Part 1)

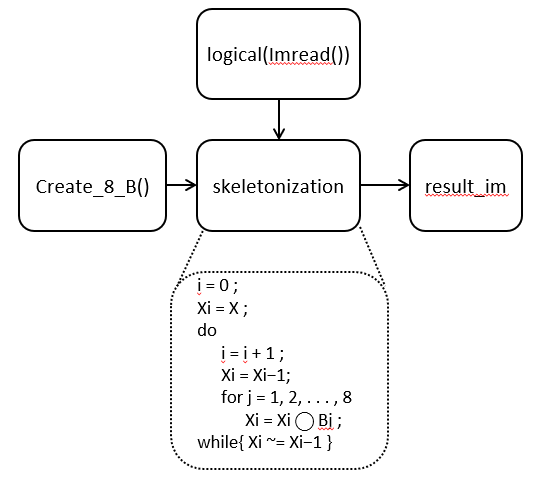
The main function of this project mainly consists of different parts, including manually creating eight structural elements, and grass-fire operation based on Hit-Or-Miss transform, as shown in the flow chart.



We will discuss these parts in the following report separately.

1. Create SE: In this part, we create eight different structural elements based on the project flyer.
2. Grass-fire operation: In this part, we used the hit-or-miss transform from project 1 and follow the algorithm provided in the project flyer to perform such operation and finally, we got skeleton of a binary image after the algorithm converged, that is there is no small details can be “burned” any more.
3. MATLAB Implementation (Homotopic Skeletonization, Part 1)

In this section, we will show the structure of the MATLAB code and how to run the code correctly.

The overall structure of the ‘main.m’ code is shown as the flow chart. The black arrow represents the process, the dotted line represents expansion of a certain function.

For part 1, there are ‘main.m’, ‘skeletonization’, ‘diff\_hit\_or\_miss.m’, ‘create\_8\_se.m’, ‘erode.m’, and‘is\_include.m’. To run the code correctly, please put all the files and image in one folder. Then just run the ‘main.m’ file, the code will automatically generate a folder named ‘results’ and put all outputs in it.

Next, we will introduce the MATLAB code in detail and analyze how to get the results step by step.

1. Read in the original image “bear.gif” or “penn256.gif
2. The ‘create\_8\_se’’ is the function to create eight SEs based on the figures given in the project flyer;
3. Next, we call the ‘skeletonization’ function, with two parameters (im,B) from step 1 and 2 respectively. In this function, we will need ‘diff\_hit\_or\_miss’ function, which is a function that performs Hit-Or-Miss and we perform X - (
4. The formula for skeletonization that scratches one layer of an image is: )
5. After there is no more layer that can be burned, the algorithm converged and we get result\_im with shape (h, w, n) which is a list of images. When n = 1, it is the original image; when n = n, we got the skeleton for the image.
   * 1. **Results**   
        1. System Output (Homotopic Skeletonization, Part 1)

In this section, we will show all the result images.

Figure 1 is ‘penn256x2.png’, the image we got for X2

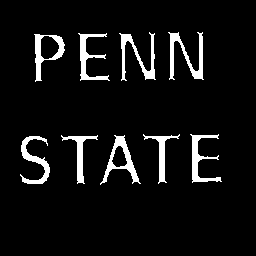


Figure 1. penn256x2.png

Figure 2 is ‘penn256x5.png’, the image we got for X5

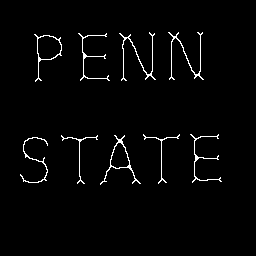


Figure 2. penn256x5.png

Figure 3 is ‘penn256x8.png’, the image we got for X8 . Note that this image is also the final skeleton.

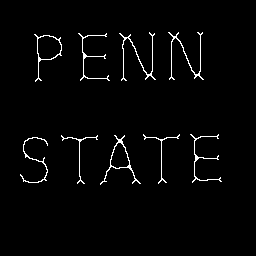


Figure 3. penn256x8.png

Figure 4 is the animation of fig. 1-3 superposed on the original image.

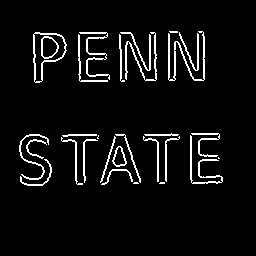


Figure 4. penn256\_superpose.gif

Figure 5 is, for simplicity, the different levels of skeletons on “bear.gif”, which contains ‘bearx2.png’, ‘bearx5.png’, and ‘bearx23.png’ (note that ‘bearx23.png’ is the final skeleton. All .png files can be found in the project submission.)

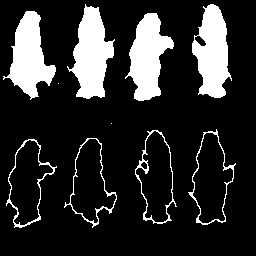


Figure 5. bear\_skeleton.gif

Figure 6 is the animation of figures in the figure 5 superposed on the original image.

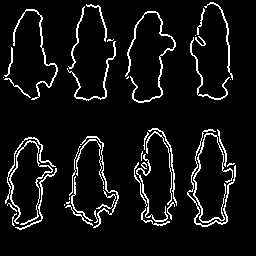


Figure 6. penn256\_superpose.gif

2. Result Analysis

(a) Penn256.png

We found that for this image, the convergence happened faster because the width of foreground is relatively small. So for grass-fire operation, the outmost layer will burn towards inner faster.

(b) B and B^c

Since we need to use B and B^c in the same hit-or-miss operation, here we did a little trick: instead of really taking the complement, we define 1 as white in B and 0 as white in B^c.

(c) Superposing

Given the images are all binary, it is hard to superpose the skeletons onto the original images. So we simply subtract the skeletons from the original image to show the changes of how we derive the final skeletons as shown in Fig. 4 and Fig. 6.

* + 1. **Conclusions**

D.1. Homotopic Skeletonization, Part 1  
We are more confident in using Matlab and Hit-or-miss transformation. From the animations, grass-fire (thinning) operation is really intuitive.