CSE585/EE555: Digital Image Processing I

Computer Project # 4:

Texture Segmentation

Lindsey Schwartz, Ke Liang, Xilun Liu

Date: 04/12/2019

A.Objectives

The goal of this project was to explore texture segmentation using Gabor filters of varying frequencies and standard deviations. The end result is a segmented image obtained by thresholding the filtered output.

B. Methods

In this project we used Gabor Filter :

$$m(x,y) = |I(x,y) * h(x,y)|$$

Where I denotes the input image, h is a GEF, and g is a circularly-symmetric Gaussian:

$$h(x,y) = g(x,y) \cdot \exp\left[j2\pi F(x\cos\theta + y\sin\theta)\right] = g(x,y) \cdot \exp\left[j2\pi(Ux + Vy)\right]$$

$$g(x,y) = \frac{1}{2\pi\sigma^2} \exp\left\{\frac{-(x^2+y^2)}{2\sigma^2}\right\}$$

Parameters (F,θ,σ) specify the GEF. You are to also implement the smoothing filter :

$$m'(x,y) = m(x,y) * g'(x,y)$$

Where g' is another circular-symmetric Gaussian using a different σ.
In the project we have 3 parts, which are shown below.

Task 1 :To apply GEF with F=0.059 cycles/pixel, θ=135deg, and σ=8 to the image “texture 1” to get m(x,y) and apply smoothing filter to m(x,y) with σ=24 to get m’(x,y). Choose the threshold to show display the segment result.
Task 2 :To apply GEF with F=0.042 cycles/pixel, θ=0deg, and σ=24 to the image “texture 2” to get m(x,y) and apply smoothing filter to m(x,y) with σ=24 to get m’(x,y). Choose the threshold to show display the segment result.
Task 3 :To apply appropriate GEF to the image “d9d77” to get m(x,y). Choose the threshold to show display the segment result.

Actually these 3 tasks can be finished by the same algorithm.

We because h(x,y) is separable in x and y, which means we can choose our own g(x,y) separated in x and y which is shown below, where σx=σy=σ.

$$g_x = \frac{1}{2\pi\sigma_x}\exp\left[-\frac{1}{2}\left(\frac{x}{\sigma_x}\right)^2\right]$$

$$g_y = \frac{1}{\sigma_y}\exp\left[-\frac{1}{2}\left(\frac{y}{\sigma_y}\right)^2\right]$$

Then we just need to calculated using the formula above, then we can get the results.

We can get the threshold by observing the 3-D figure m(x,y) along the direction representing m and in the project in order to make it easy to observe, we choose a range of threshold, and make the I(x,y) whose m(x,y) belongs to the range equal to 1. Then we can get the results of the segment result.

As for the first task, we can run code GEF_Filter_texture1.m to get the results, and in this task the threshold is range of [0.0071,0.0074].

As for the second task, we can run code GEF_Filter_texture2.m to get the results, and in this task the threshold is range of [0.0072,0.0074].

As for the third task, we apply GEF with F=0.059 cycles/pixel, θ=0deg, and σ=25 to the image "d9d77" to get m(x,y) and apply smoothing filter to m(x,y) with σ=25 to get m'(x,y). In this task the threshold is range of [0.011,0.0125].

The results are shown below.

C. Results

The original images, m (x,y) and m'(x,y) as gray-scale images and 3D plots are shown below.

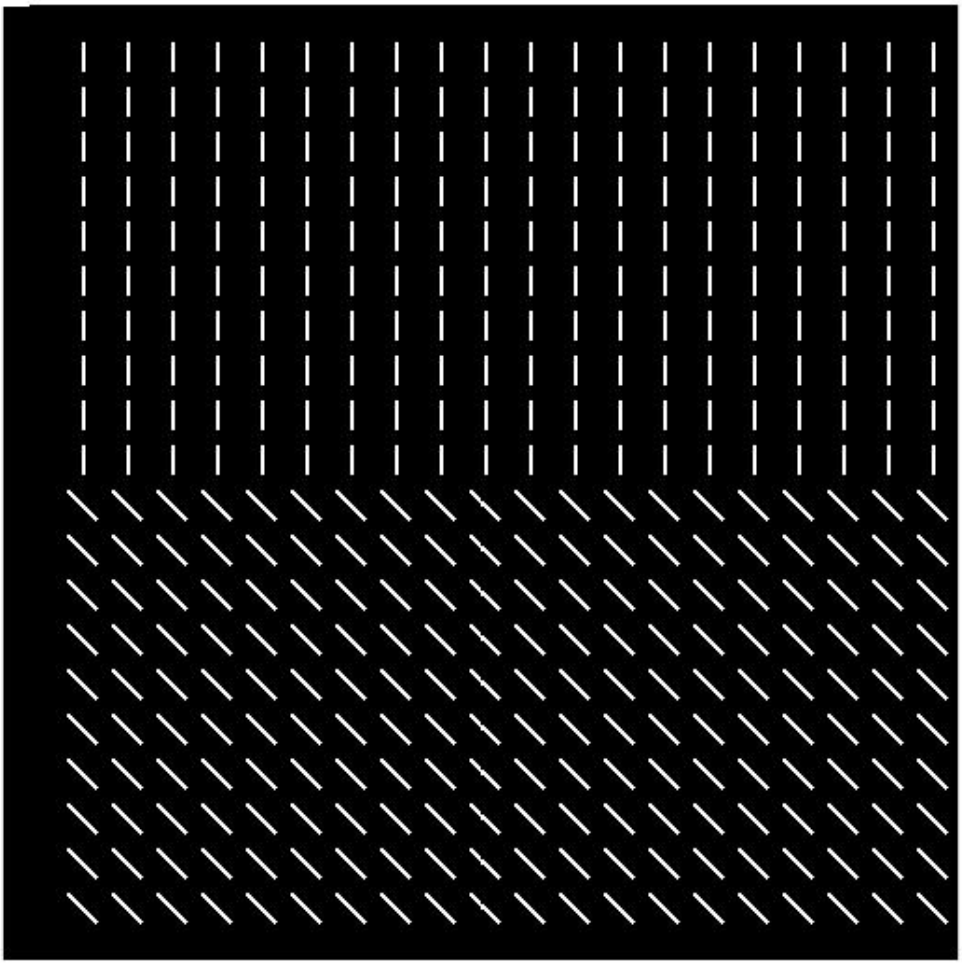


Figure 1 Original Image of “Texture 1”

Figure 2 and Figure 3 are the image applying GEF with F=0.059 cycles/pixel, θ=135deg, and σ=8 to the image “texture 1” to get m(x,y) and Figure 4 and Figure 5 are the image applying smoothing filter to m(x,y) with σ=24 to get m’(x,y).

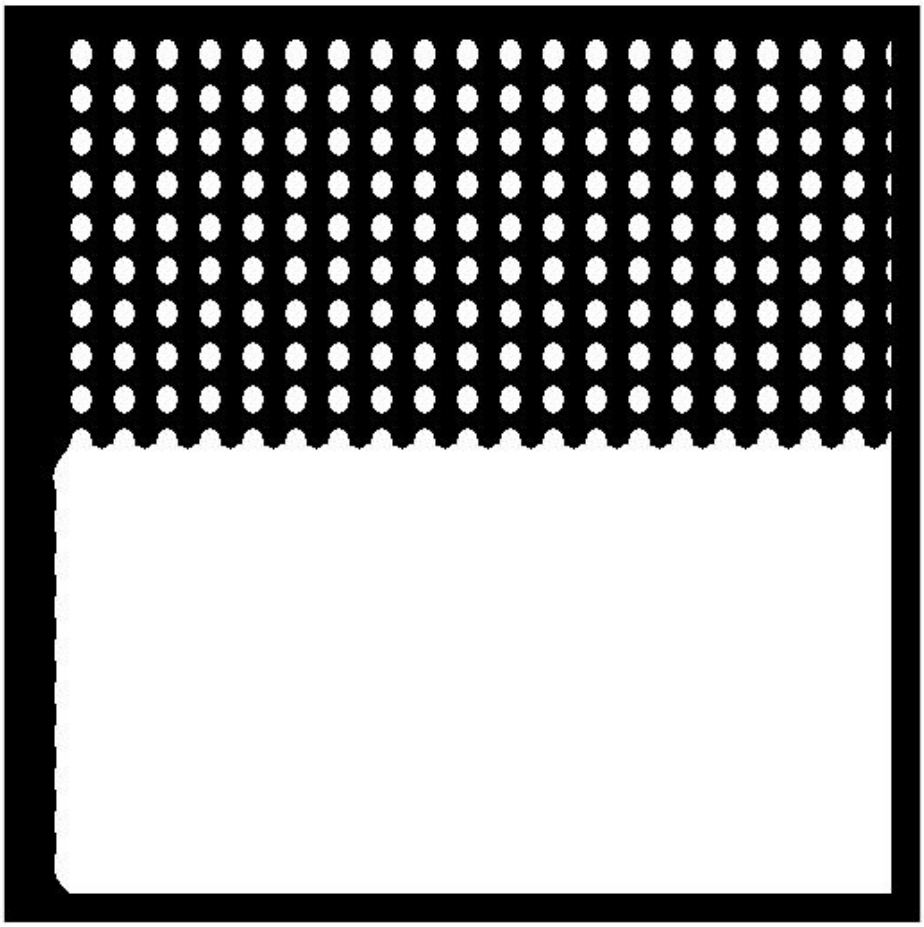


Figure 2 m(x,y) for “Texture 1” GEF not “perfect” F=0.059 cycles/pixel, θ=135deg, and σ=8 pixels shown as gray-scale image

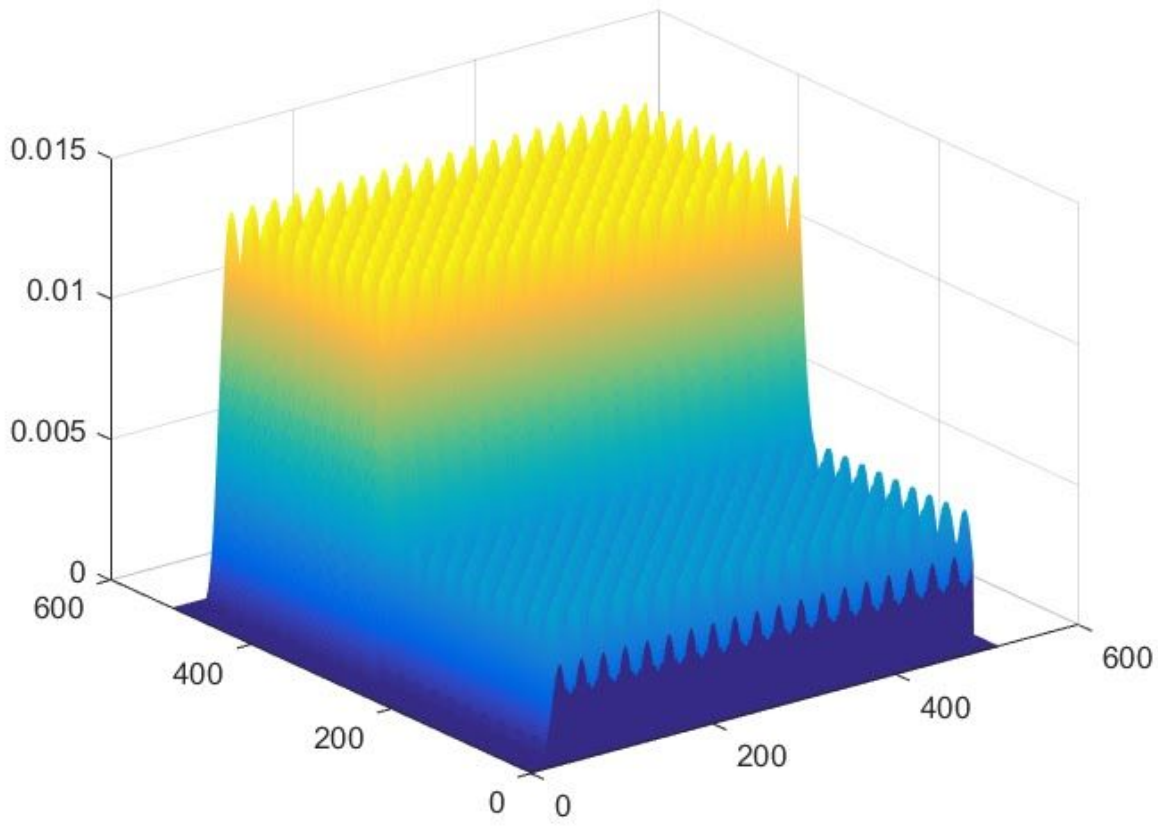


Figure 3 m(x,y) for “Texture 1” GEF not “perfect” F=0.059 cycles/pixel, θ=135deg, and σ=8 pixels shown as 3D plot

Figure 4 m'(x,y) for "Texture 1" GEF after smothing F=0.059 cycles/pixel, θ=135deg, and σ=24 pixels shown as gray-scale image

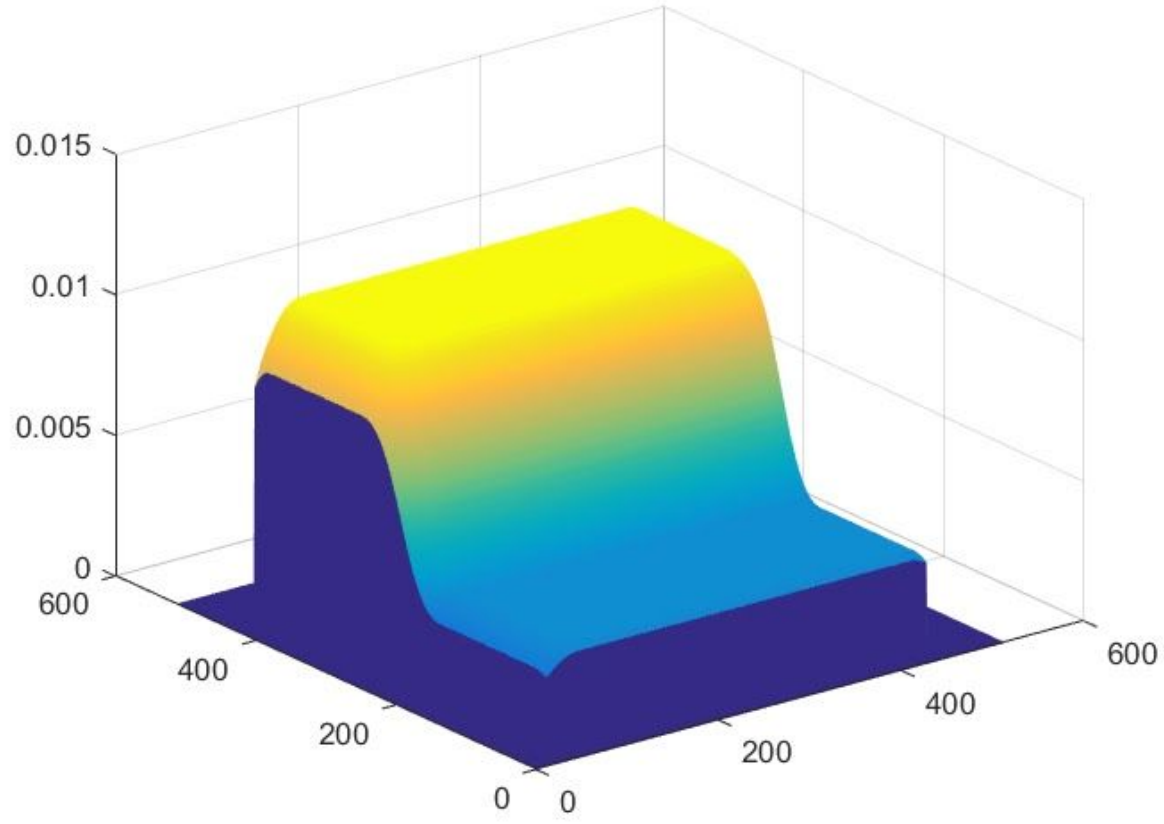


Figure 5 m'(x,y) for "Texture 1" GEF after smothing F=0.059 cycles/pixel, θ=135deg, and σ=24 pixels shown as 3D plot

Segmented image for "Texture 1" with threshold ranging from 0.0071 to 0.0074 shown below Figure 7, and the segmented line is in the middle.

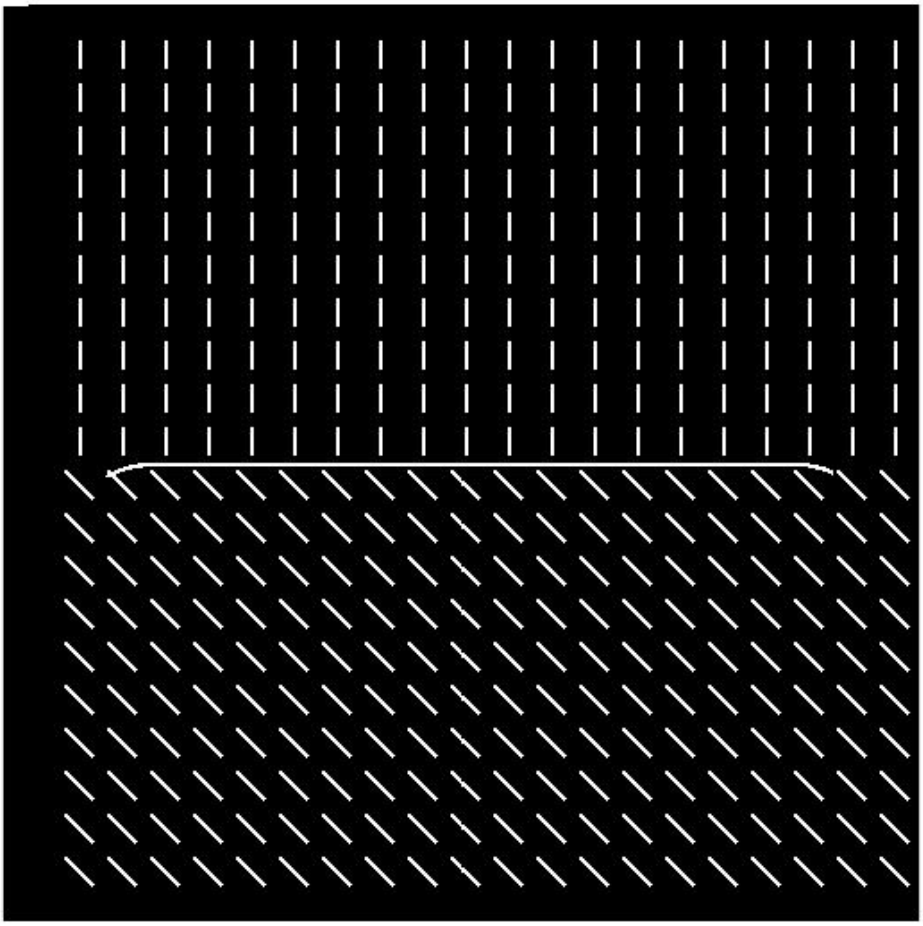


Figure 6 Segmented image for “Texture 1” with threshold ranging from 0.0071 to 0.0074

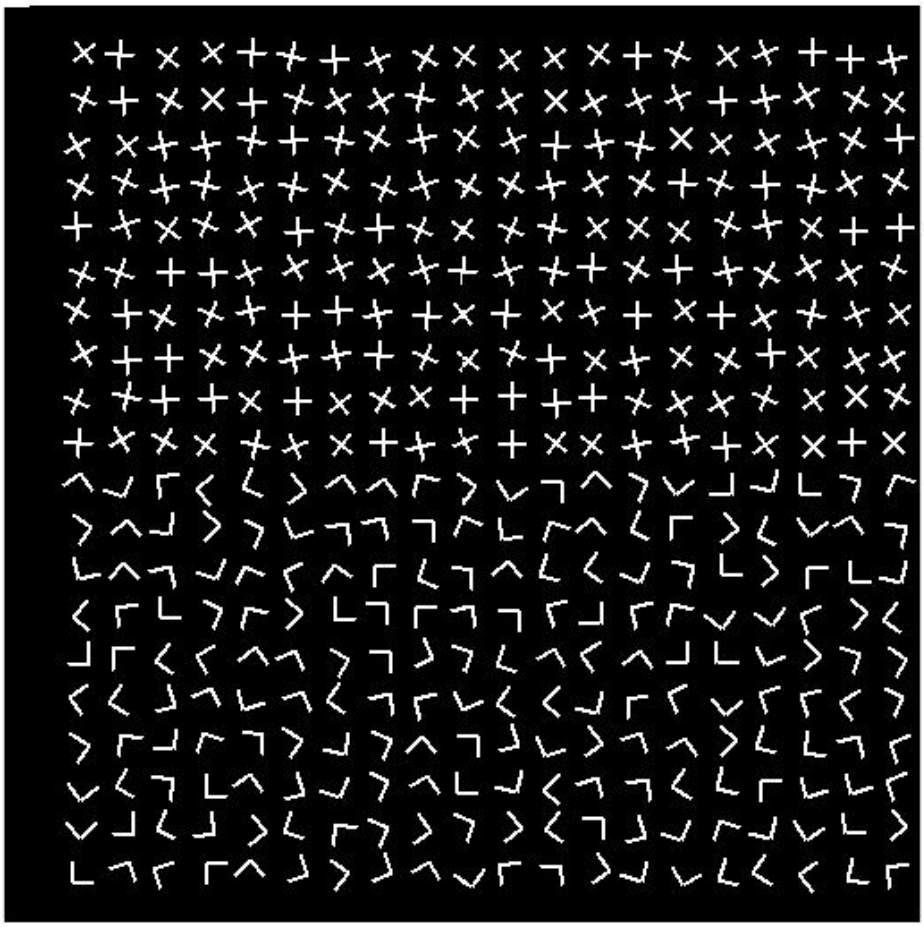


Figure 7 Original Image of “Texture 2”

Figure 8 to Figure 10 are the images applying GEF with F=0.042 cycles/pixel, θ=0deg, and σ=24 to the image “texture 2” to get m(x,y) and Figure 11 to Figure 13 are the image applyingsmoothing filter to m(x,y) with σ=24 to get m’(x,y).

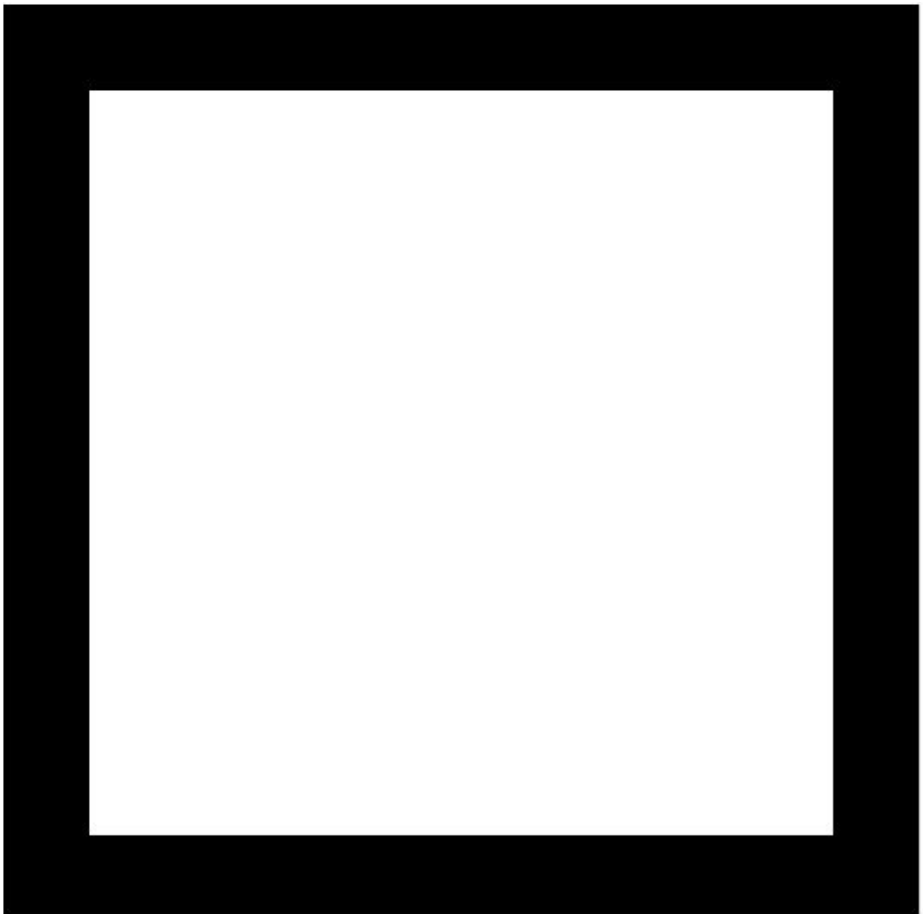


Figure 8 m(x,y) for "Texture 2" GEF not "perfect" F=0.042 cycles/pixel, θ=0deg, and σ=24 pixels shown as gray-scale 1

Because all the m of the the image is small, so we also show the image below to help us to observe the image.

Figure 9 m(x,y) for “Texture 2” GEF not “perfect” F=0.042 cycles/pixel, θ=0deg, and σ=24 pixels shown as gray-scale 2

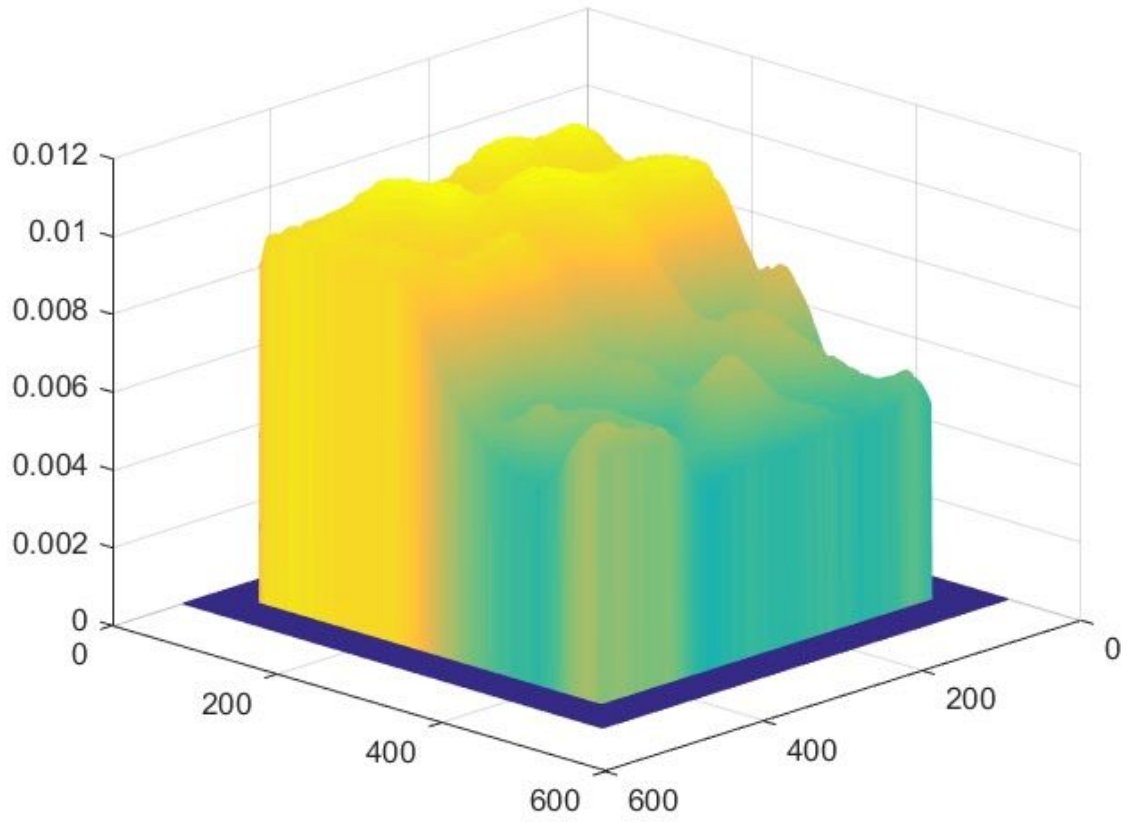


Figure 10 m(x,y) for “Texture 2” GEF not “perfect” F=0.042 cycles/pixel, θ=0deg, and σ=24 pixels

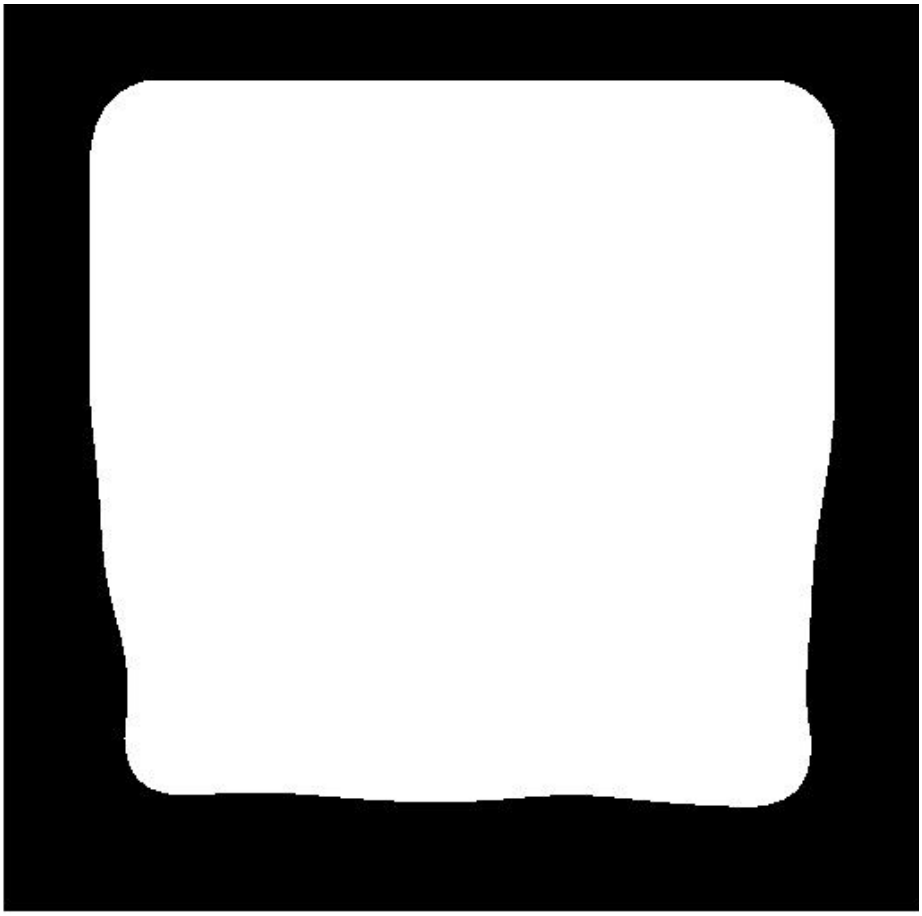


Figure 11 m'(x,y) for "Texture 2" GEF not "perfect" F=0.042 cycles/pixel, θ=0deg, and σ=24 pixels shown as gray-scale 1

Because all the m' of the the image is small, so we also show the image below to help us to observe the image.

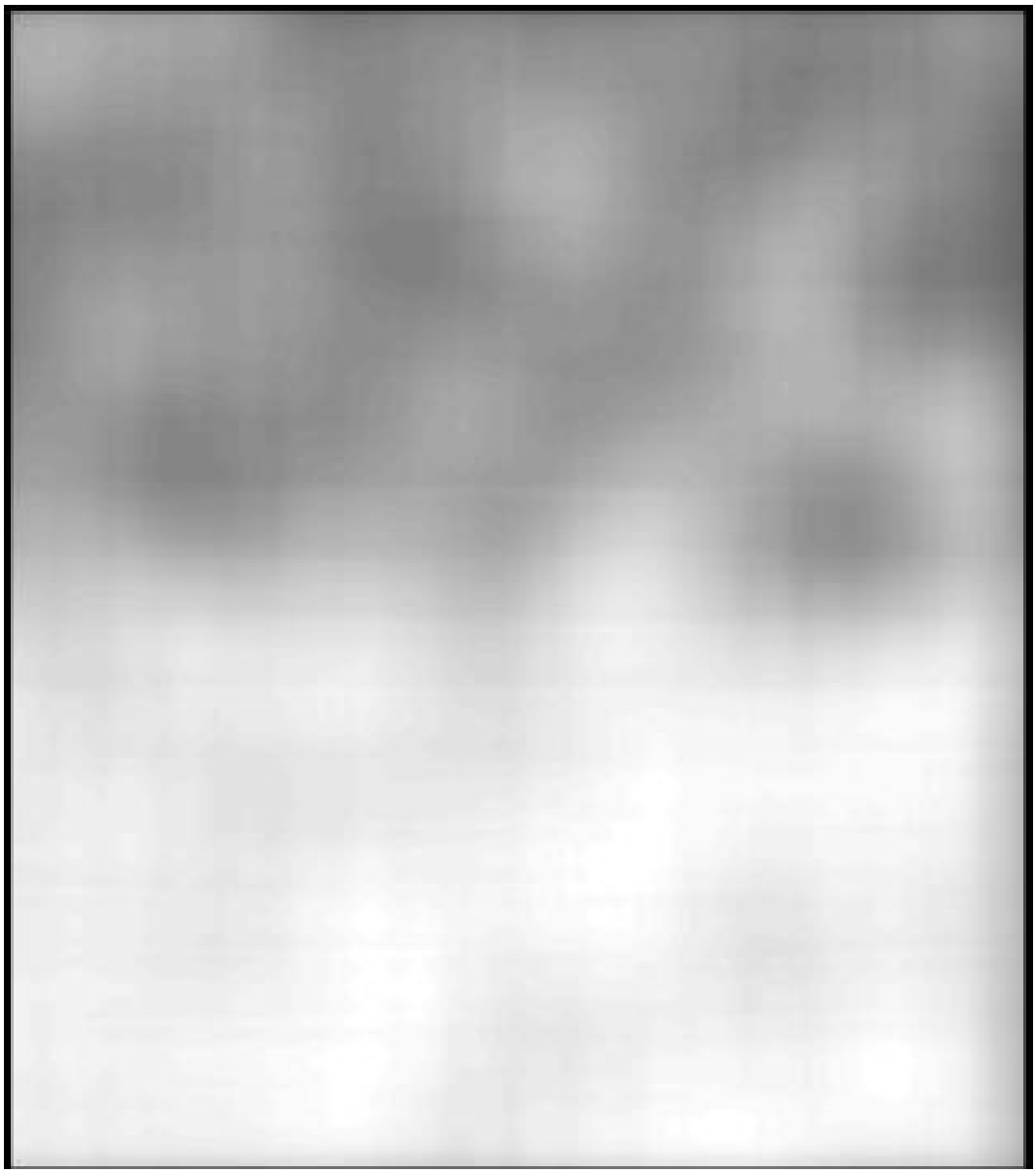


Figure 12 m'(x,y) for "Texture 2" GEF not "perfect" F=0.042 cycles/pixel, θ=0deg, and σ=24 pixels shown as gray-scale 2

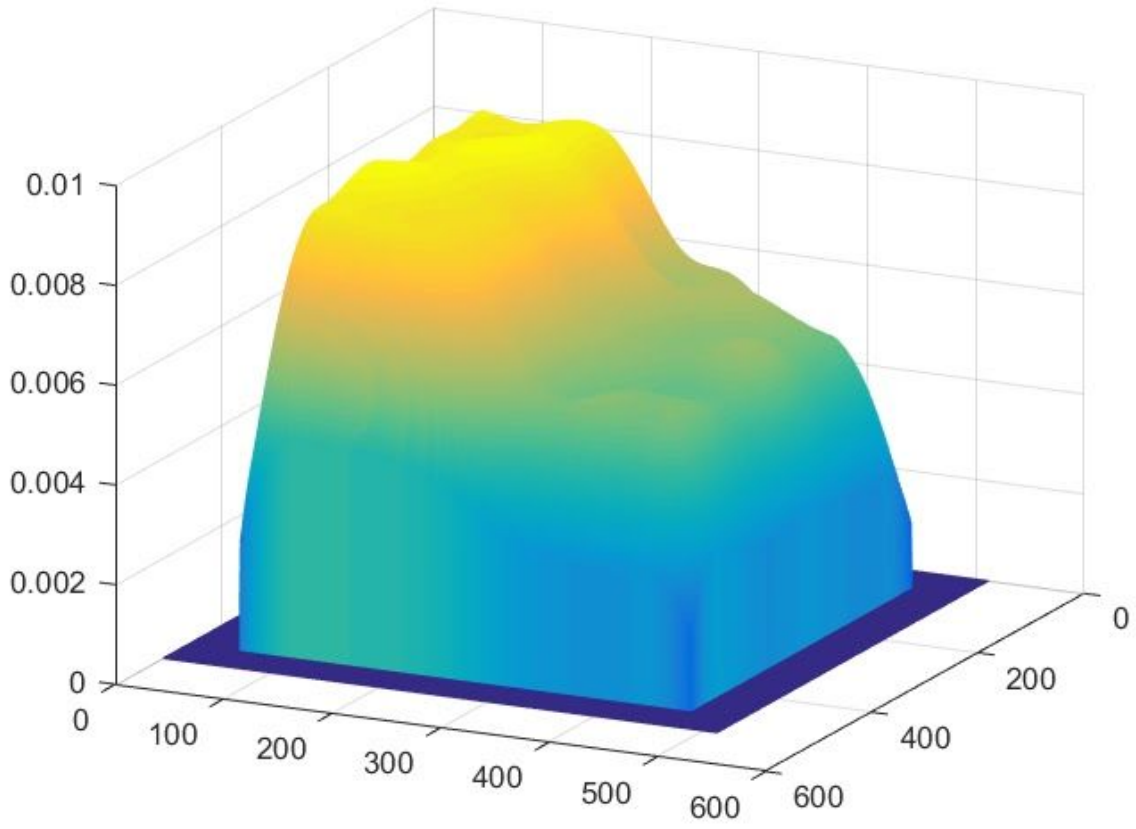


Figure 13 m'(x,y) for "Texture 2" GEF after smothing F=0.042 cycles/pixel, θ=0deg, and σ=24 pixels

Segmented image for "Texture 2" with threshold ranging from 0.0072 to 0.0074 shown below Figure 8, and the segmented line is in the middle.

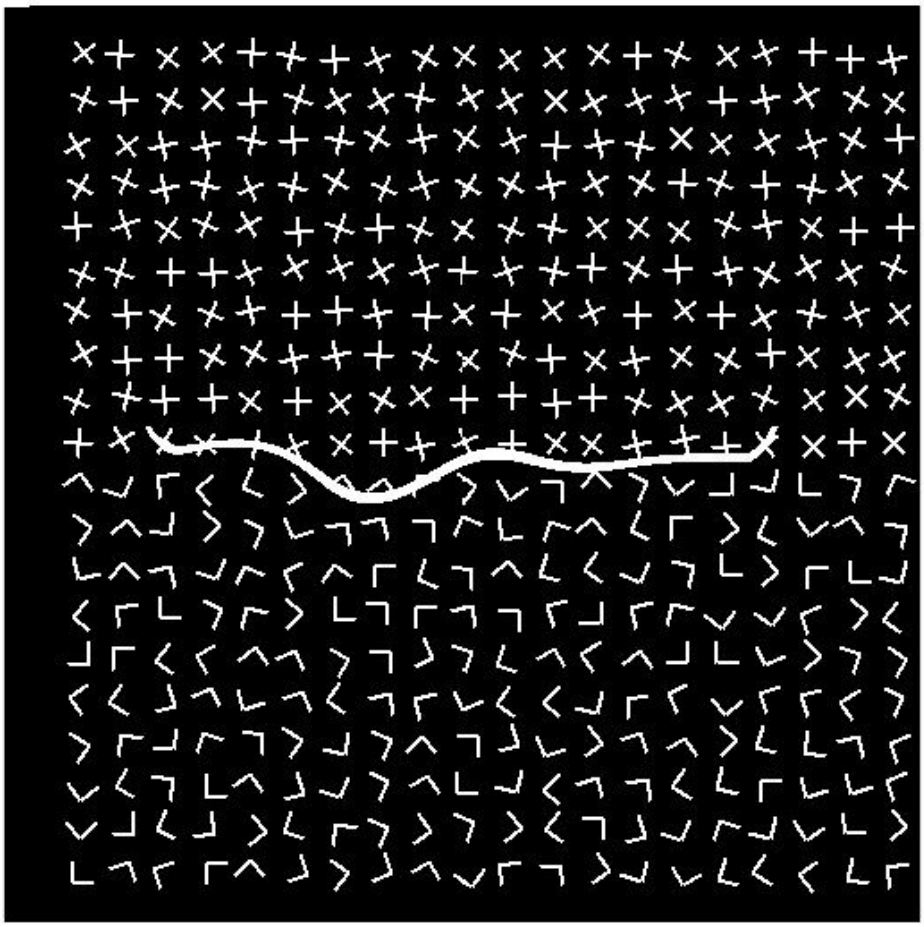


Figure 14 Segmented image for “Texture 2” with threshold ranging from 0.0072 to 0.0074

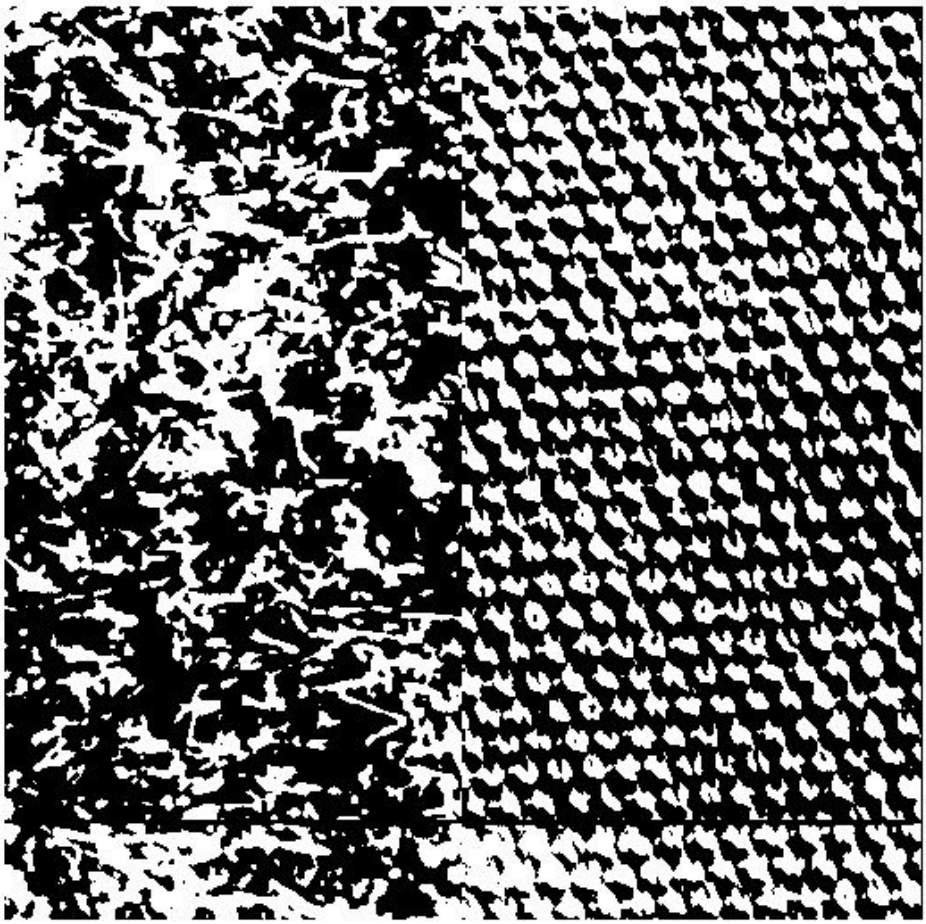


Figure 14 Original Image of “d7d99”

Figure 15 and Figure 16 are the images applying GEF with F=0.059 cycles/pixel, θ=0deg, and σ=25 to the image “d9d77” to get m(x,y) and Figure 17 and Figure 18 are the images applying smoothing filter to m(x,y) with σ=25 to get m’(x,y).

Figure 15 m(x,y) for “d7d99” GEF not “perfect” F=0.059 cycles/pixel, θ=0deg, and σ=25 pixels shown as gray-scale image

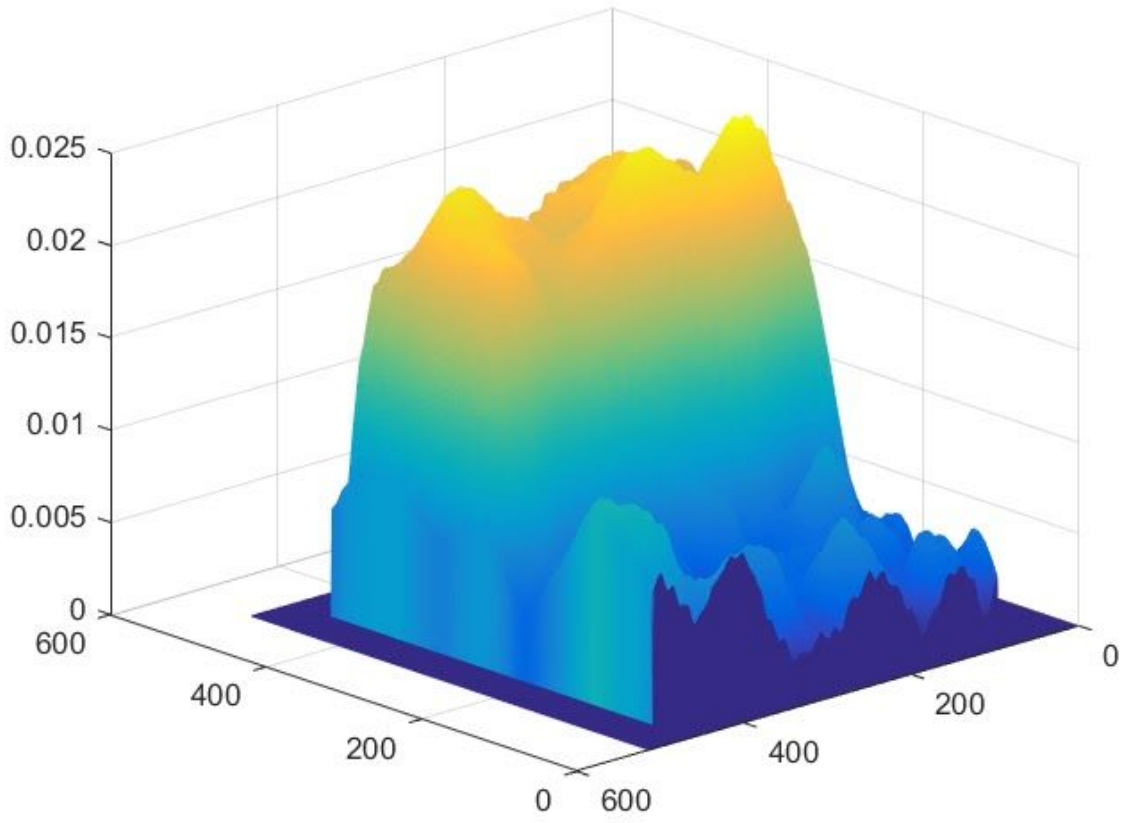


Figure 16 m(x,y) for “d7d99” GEF not “perfect” F=0.059 cycles/pixel, θ=0deg, and σ=25 pixels shown as 3D plot

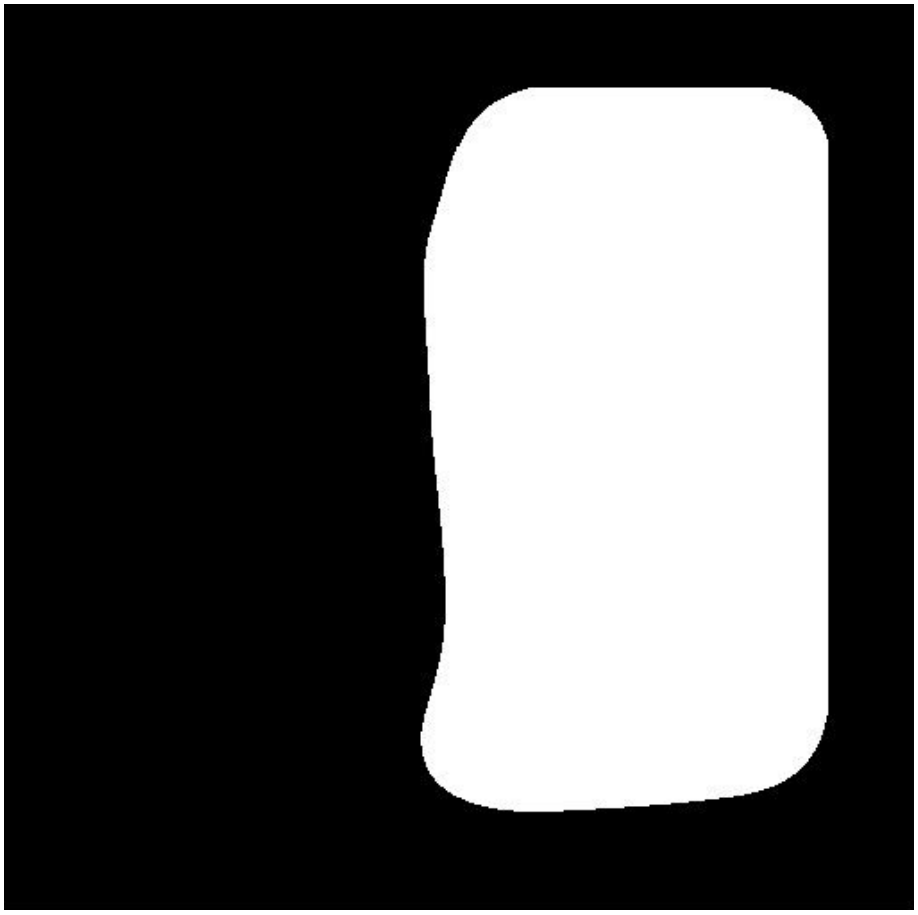


Figure 17 m'(x,y) for "d7d99" GEF after smothing F=0.059 cycles/pixel, θ=0deg, and σ=25 pixels shown as gray-scale image

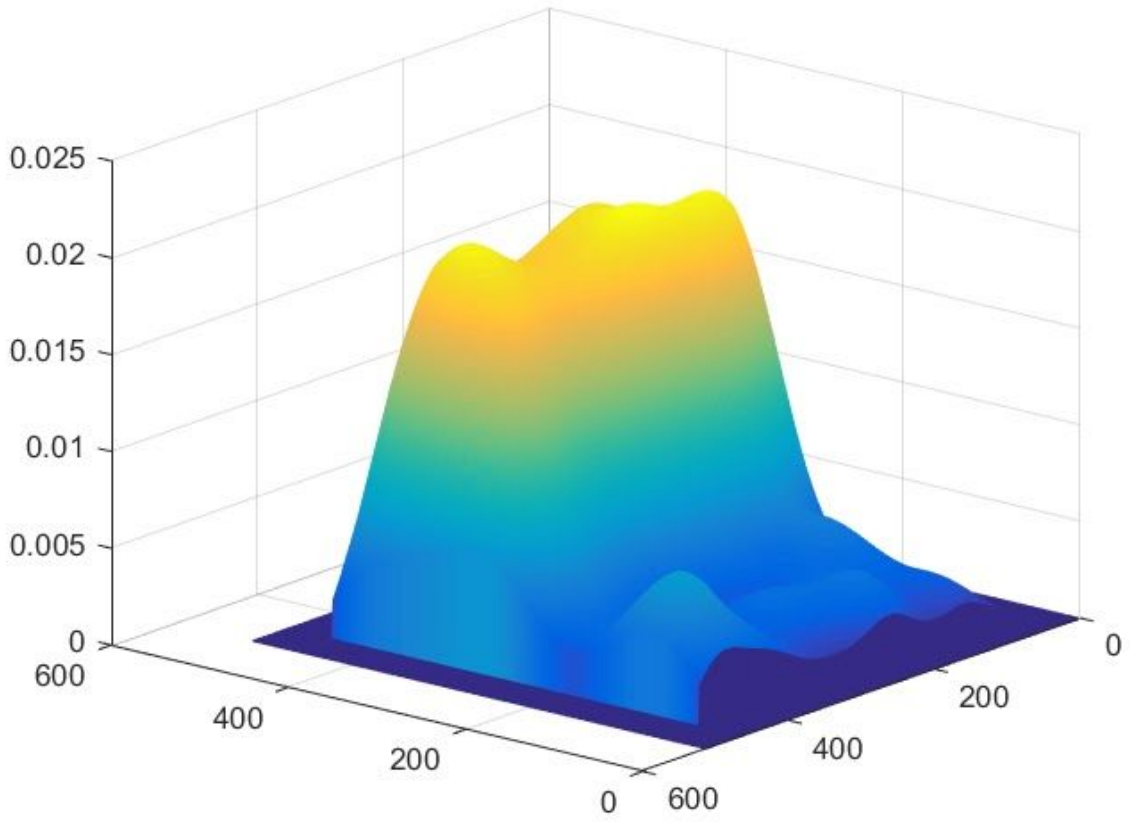


Figure 18 m'(x,y) for "d7d99" GEF after smothing F=0.059 cycles/pixel, θ=0deg, and σ=25 pixels shown as 3D plot

Segmented image for "d7d99" with threshold ranging from 0.011 to 0.0125 shown below Figure 19, and the segmented line is in the middle. In Figure 20, the segmented line the mentioned in the red box.

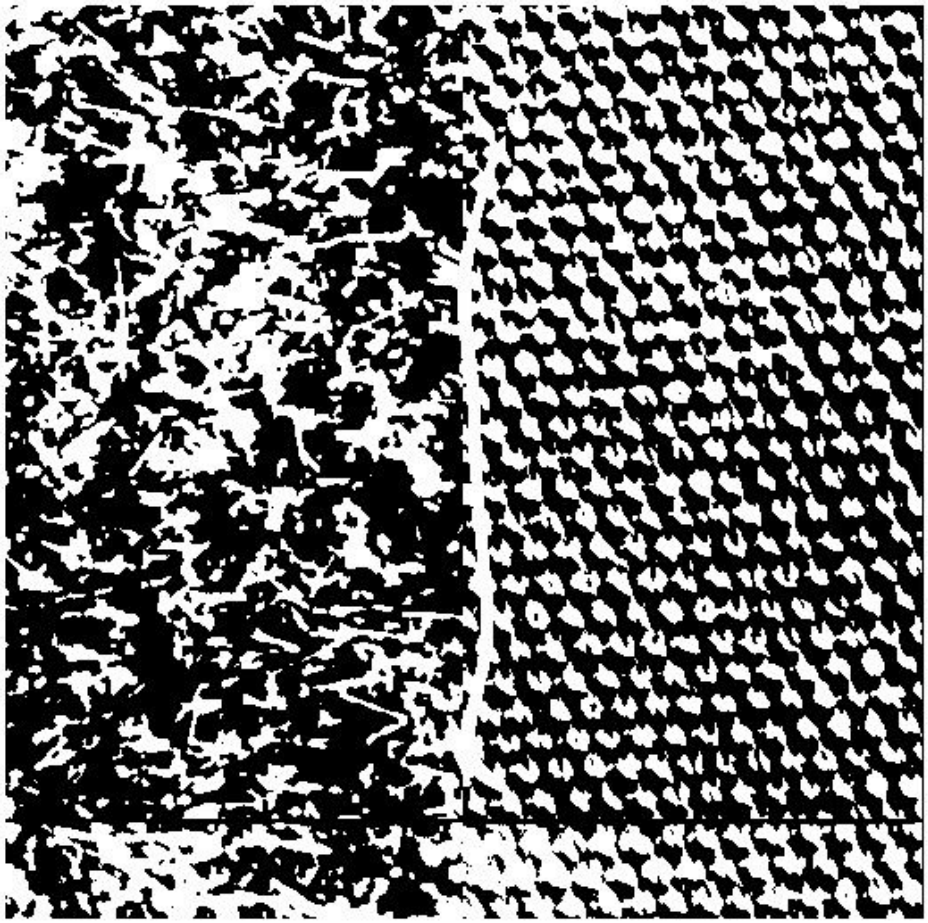


Figure 19 Segmented image for “d7d99” with threshold ranging from 0.011 to 0.0125

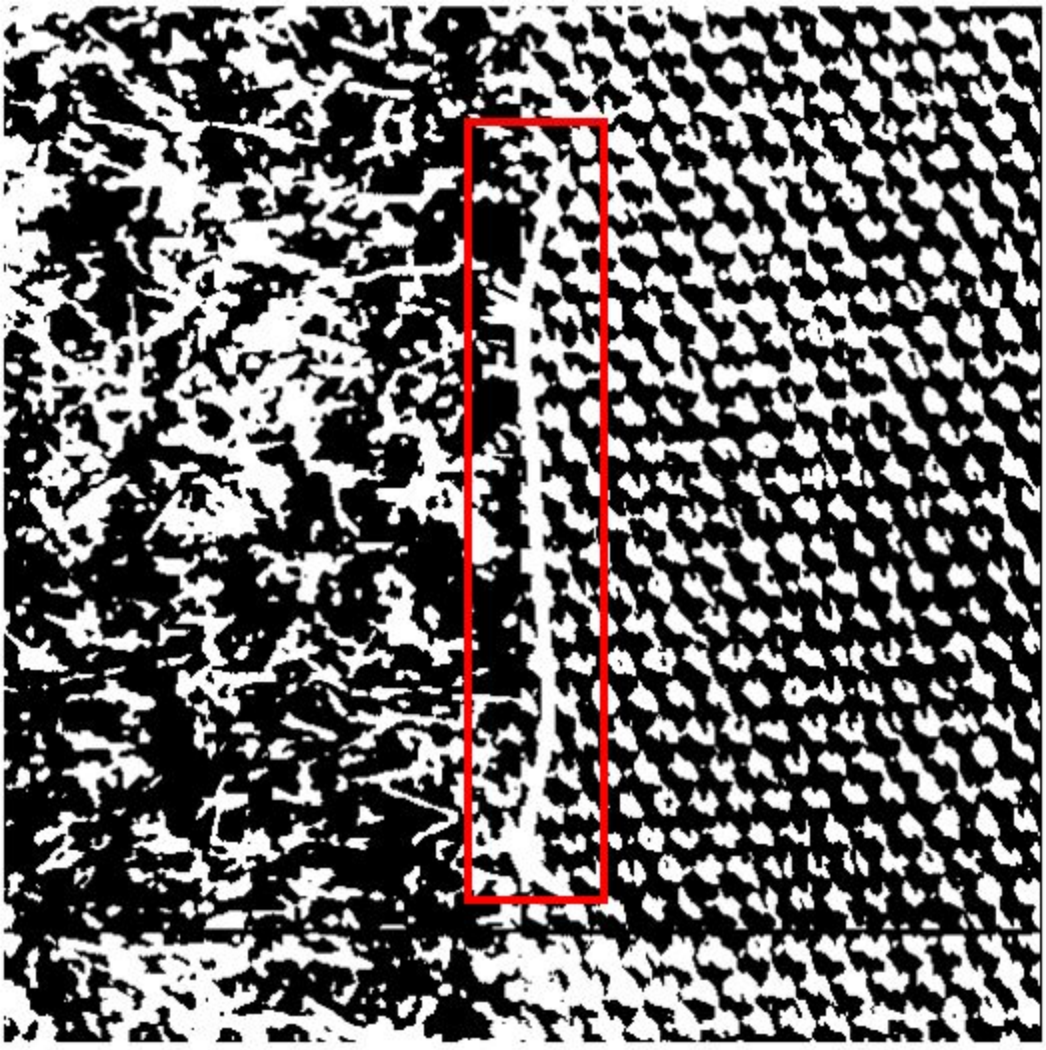


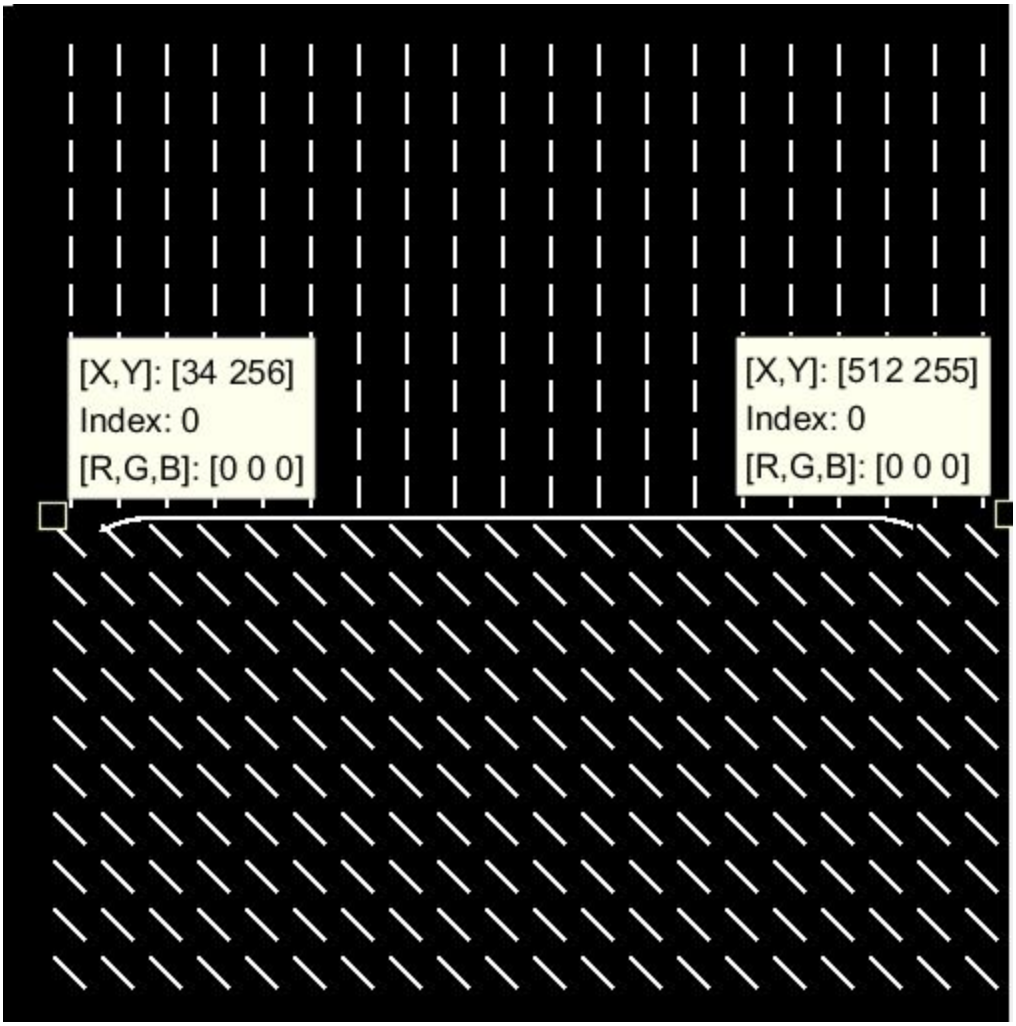
Figure 20 Segmented image for “d7d99” with threshold ranging from 0.011 to 0.0125

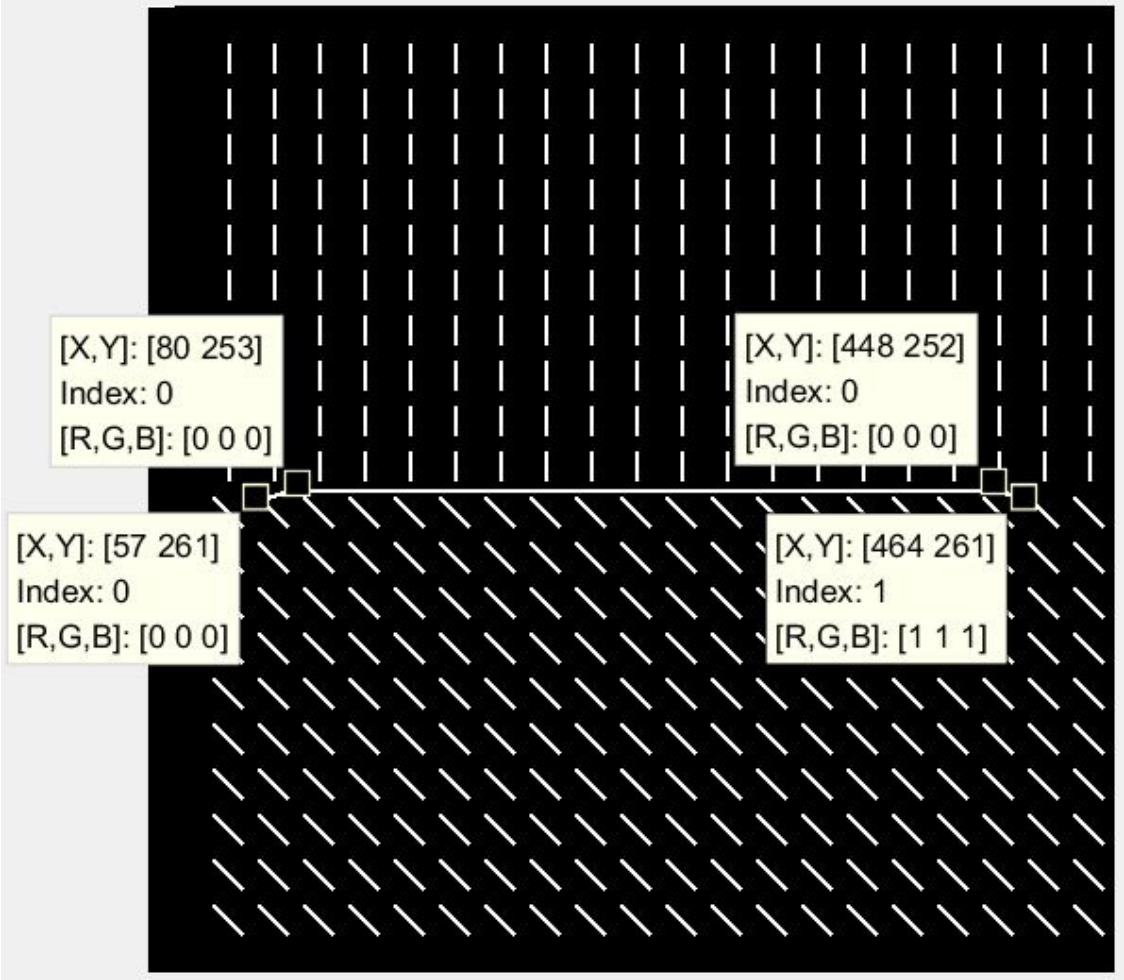
The implementation of these 3 tasks are shown in the method part, and I think the result is nearly perfect except for the segmented image.

Because the algorithm in the project will make the region of m(x,y) and m’(x,y) a little bit smaller than the original image, so the segmented lines we find are a little bit shorter in original image.

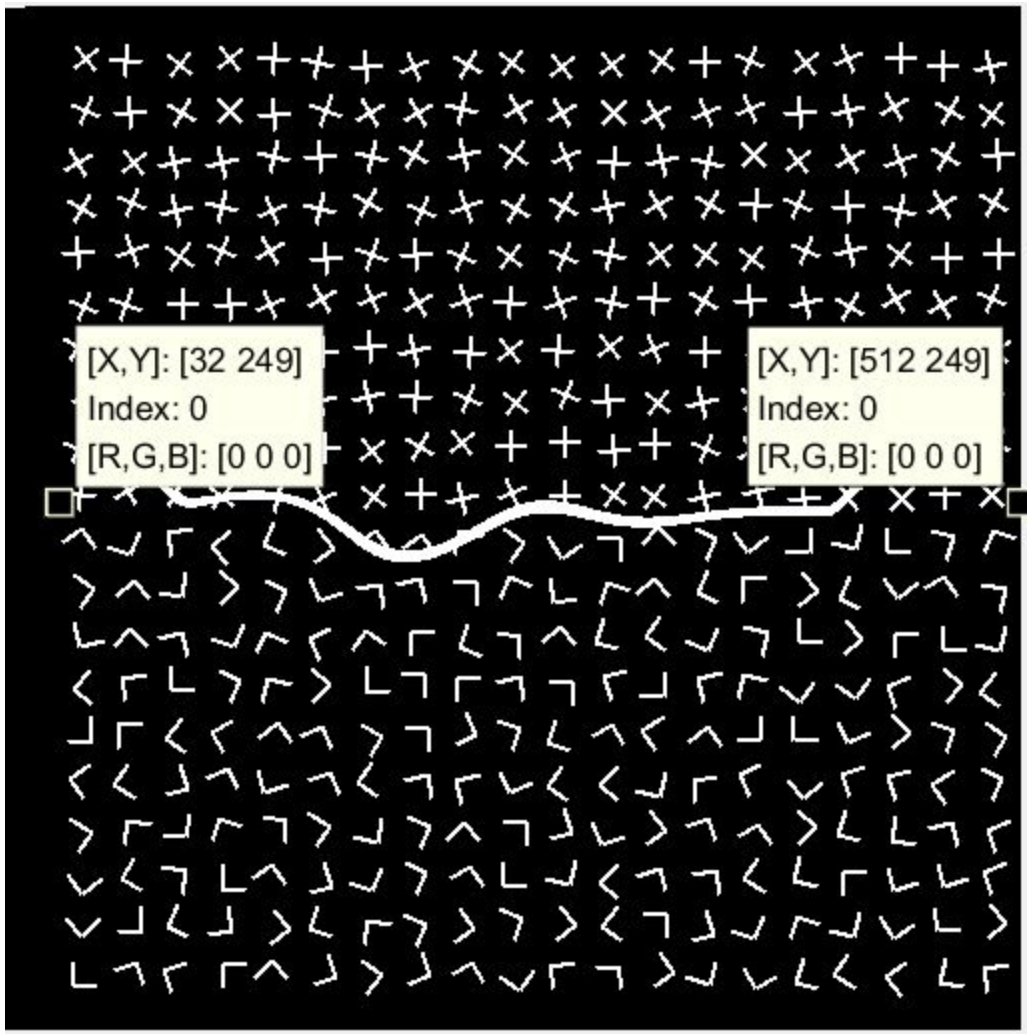
As for the portion of each image could actually segmented we can get the coordinate based on m’(x,y) as gray-scale images, which are shown below.

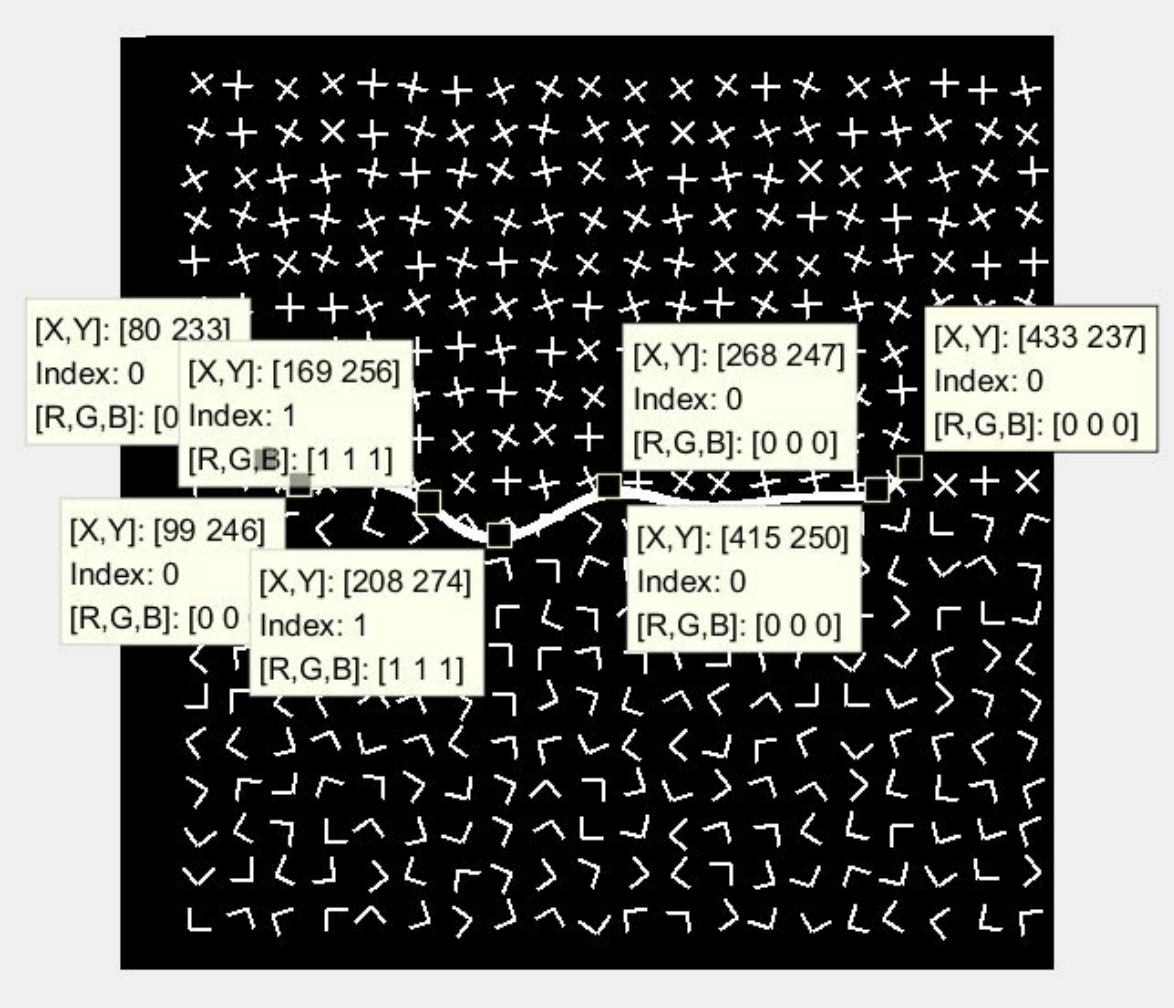
As for task 1, we can get the coordinates of the segmented line based on original images. So x is ranging from 34 to 512, and y is 256. The segmented line is the line connected by points (57,261), (80, 253), (448,252) and (464,261).



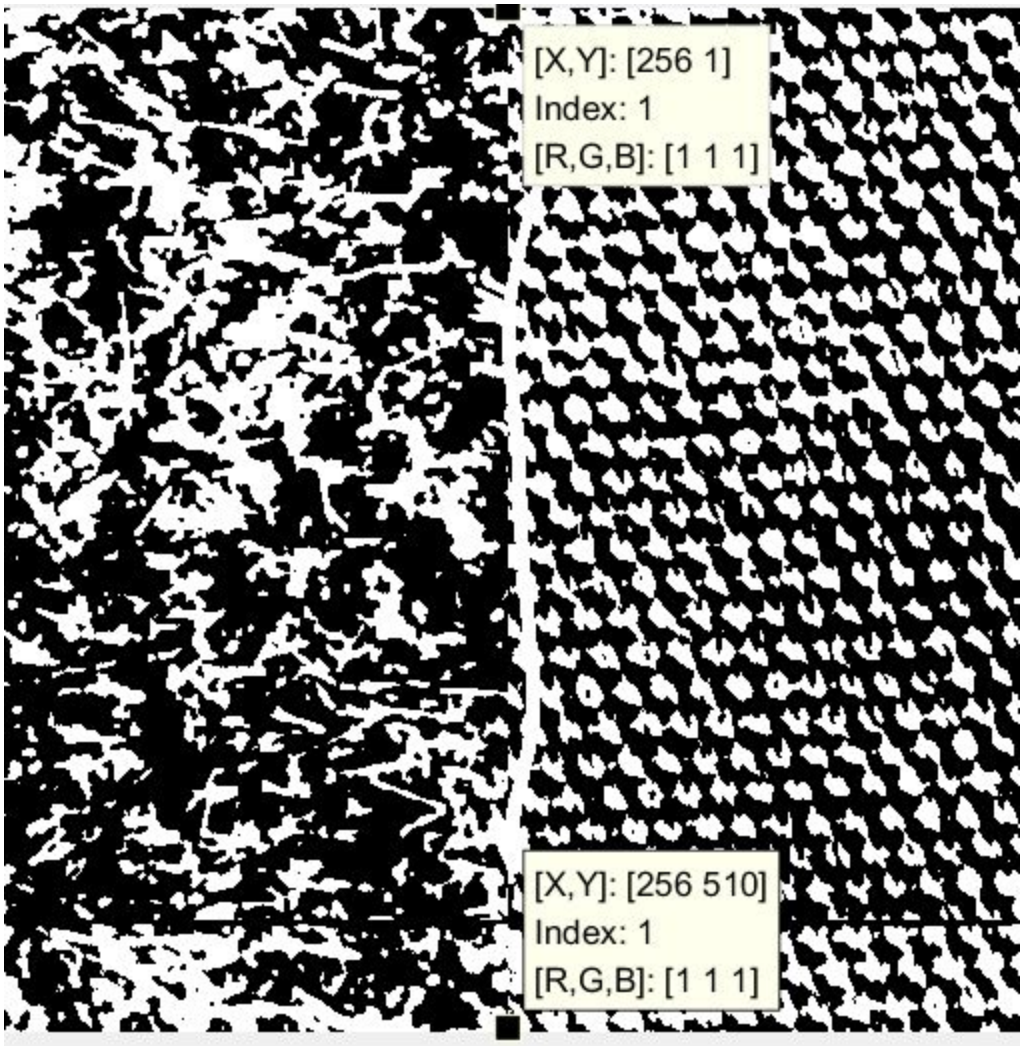


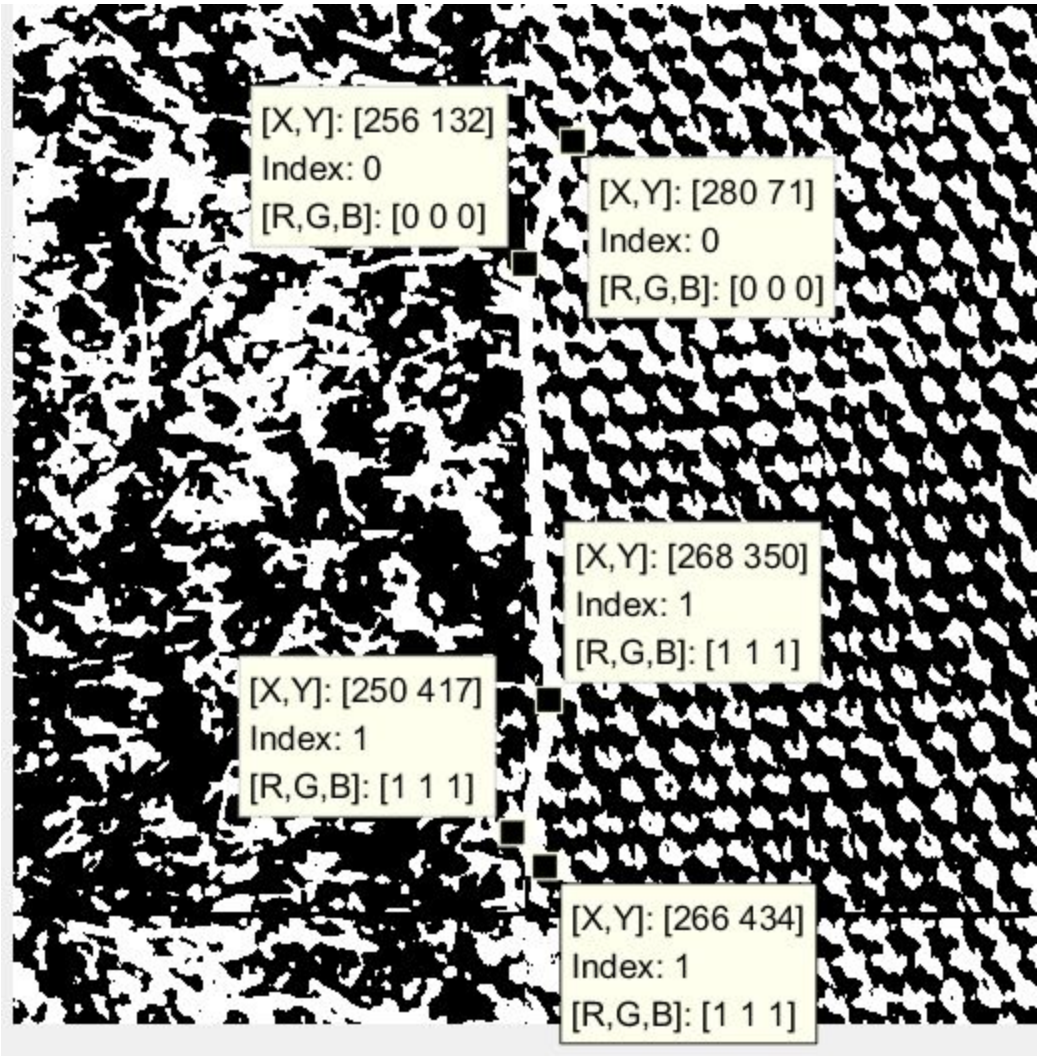
As for task 2, we can get the coordinates of the segmented line based on original images. So x is ranging from 32 to 512, and y is ranging from 249. The segmented line is the line connected by points (80,233), (99, 246), (169,208), (268, 247), (415, 250)and (433,237).


[X,Y]: [32 249]
Index: 0
[R,G,B]: [0 0 0]
[X,Y]: [512 249]
Index: 0
[R,G,B]: [0 0 0]



As for task 3, we can get the coordinates of the segmented line based on based on original images . So range of x is 256, and y is ranging from 1to 512. The segmented line is the line connected by points (256,132), (280, 71), (268,350), (250, 417) and (266,434).


[X,Y]: [256 1]
Index: 1
[R,G,B]: [1 1 1]
[X,Y]: [256 510]
Index: 1
[R,G,B]: [1 1 1]



D. Conclusions

In this project, we explored and implemented texture segmentation using Gabor filters, which is useful for solving a wide variety of problems. Overall, the segmentation algorithm worked extremely well. In each case, it is clear that there are two textures and the Gabor filters were able to easily distinguish between them. The images “Texture 2” and “d9d77” are slightly more complicated than the image “Texture 1” in that the two textures are more similar. This is evident when looking at the line that distinguishes the two classes. In the “Texture1” image, the line separating the two classes is relatively straight and does not include any of the first texture in the second

texture and vice versa. This is not the case for images “Texture2” and “d9d77”. In these images, the line separating the two textures is drawn such that some of texture 1 is said to contain texture 2 and vice versa. But, overall, the Gabor filters managed to separate the textures with minimal error.