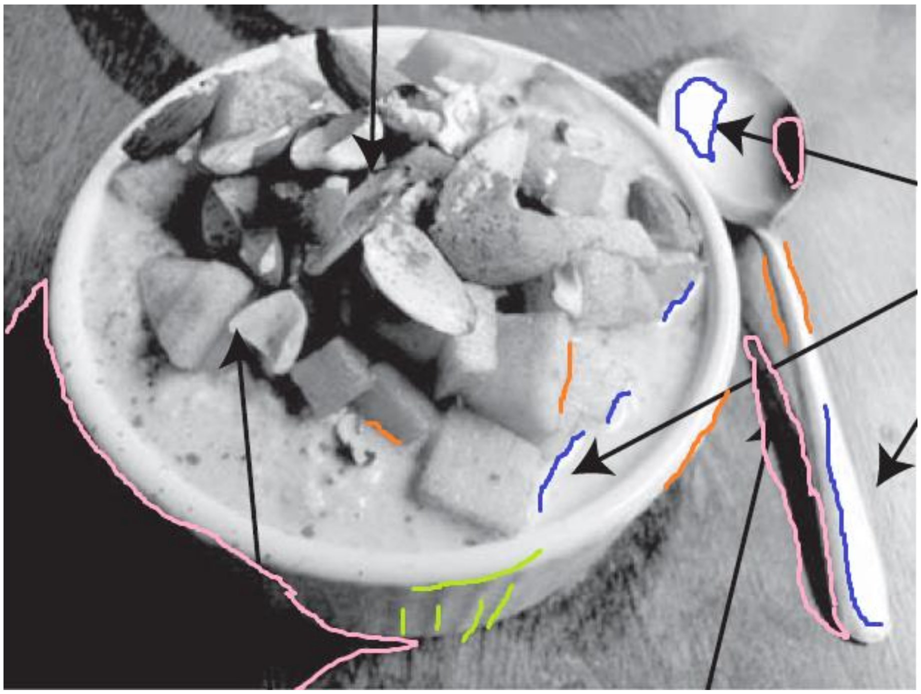
CSE 5522 Homework 4
Manjari Akella
11/27/2013

1. Do problem 24.4 in AIMA (p 969).



Discontinuity in Depth (orange)

- between the handle of the spoon and the table below
- between the rim of the bowl and the table below
- between the nuts/fruits and the liquid in the bowl
- between some of the nuts/fruits and others

Discontinuity in Surface Orientation (green)

- between the rim of the bowl and the curved surface area of the bowl
- along the curved surface area of the bowl

Discontinuity in Reflectance (blue)

- in the base of the spoon
- along the edges of some fruits where it intersects the liquid
- along the handle of the spoon

Discontinuity in Illumination (pink)

- the shadow of the handle of the spoon
- the shadow of the bowl
- in the base of the spoon

2. Implement the edge detection algorithm described in class and in the book to detect pixels that are likely edges in the images.

<u>Part I:</u> Smooth each of the images with a 2-D gaussian filter. For one of the images, try different sigmas and different filter lengths. Describe your results. Turn in the code and the smoothed images.

<u>Part II:</u> Implement a vertical line detector with a Gaussian derivative in the x-direction.

<u>Part III:</u> Implement a similar horizontal line detector, and show the output.

<u>Part IV:</u> Compute the any-orientation edge detection, given by sqrt(Gx(x,y)^2+Gy(x,y)^2). Show the output on all three images.

<u>Part V:</u> What happens if you threshold the output at a certain level? Choose one level and show the output on all three images with the same level. Discuss your findings.

<u>Extra credit:</u> Implement the final stages of Canny edge detection algorithm, where you find line segments from the edges.

<u>Part I</u>

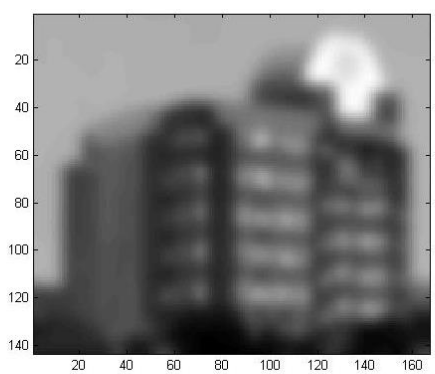


Figure 1: sigma=3, filter length=19

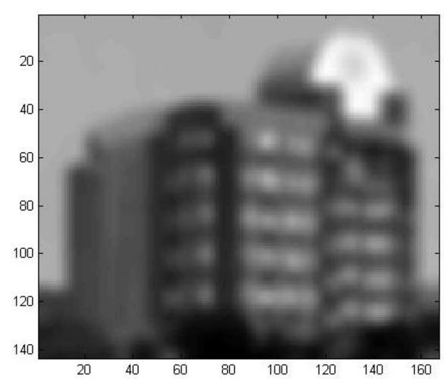


Figure 2: sigma=3, filter length=10

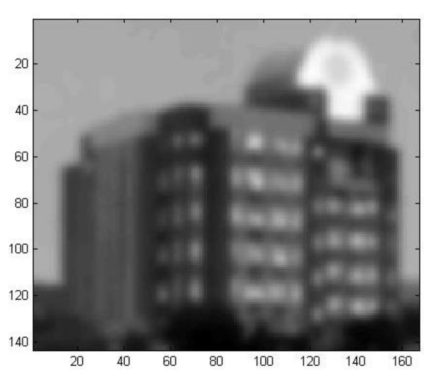


Figure 3: sigma=2, filter length=19

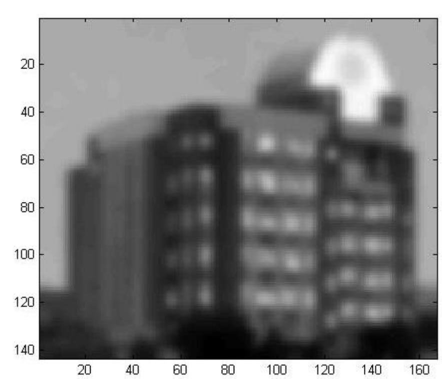


Figure 4: sigma=2, filter length=10

- As sigma is increased and filter length is kept constant, the picture becomes more blurred.
- When sigma is kept constant and filter length is increased, the picture becomes more blurred. This increase in blurriness is more apparent when sigma is itself high. That is to say the effect is more pronounced when sigma is kept at 3 and filter lengths are varied rather than when sigma is kept at 2.
- This makes sense because smoothing basically tries to smooth out noise in an image. So when we increase the value of sigma or the filter length, we tend to get a larger smoothing factor. Increasing sigma will increase the weight given to the pixels closer to the one under consideration whereas increasing filter length will look at a larger neighborhood around a pixel under consideration.

Part II

- For Parts II-Extra credit, I used sigma=3, filter length=19

$$\frac{\partial g(x,y;\sigma)}{\partial x} = \frac{-(x-x_o)}{2\pi\sigma^4} e^{-\frac{(x-x_o)^2+(y-y_o)^2}{2\sigma^2}} \qquad \frac{\partial g(x,y;\sigma)}{\partial y} = \frac{-(y-y_o)}{2\pi\sigma^4} e^{-\frac{(x-x_o)^2+(y-y_o)^2}{2\sigma^2}}$$

- Also in the formula for computing the derivative of Gaussian, I removed –ve sign from the first term {$(x-x_0)$ and $(y-y_0)$} in the formula to orient the system similar to Cartesian system (Images have a different axes system)

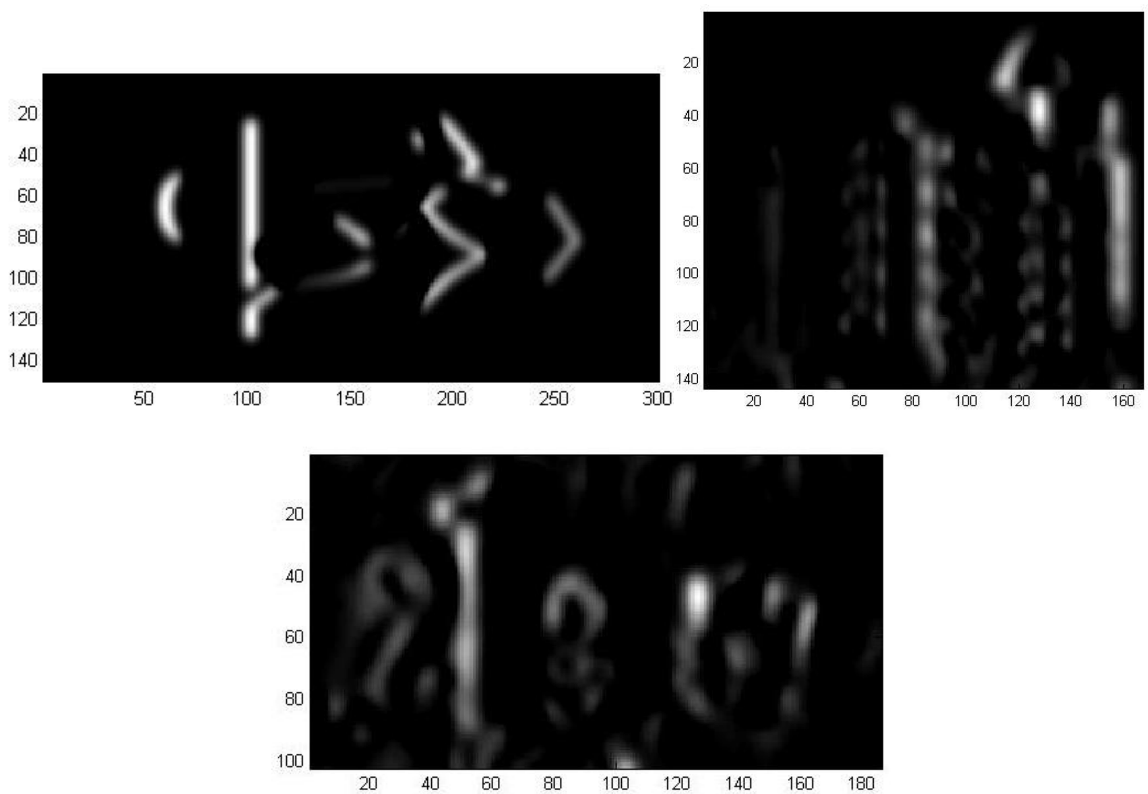


Figure 5: x-gradiants (shapes.raw,dreese.raw,num.raw)

Part III

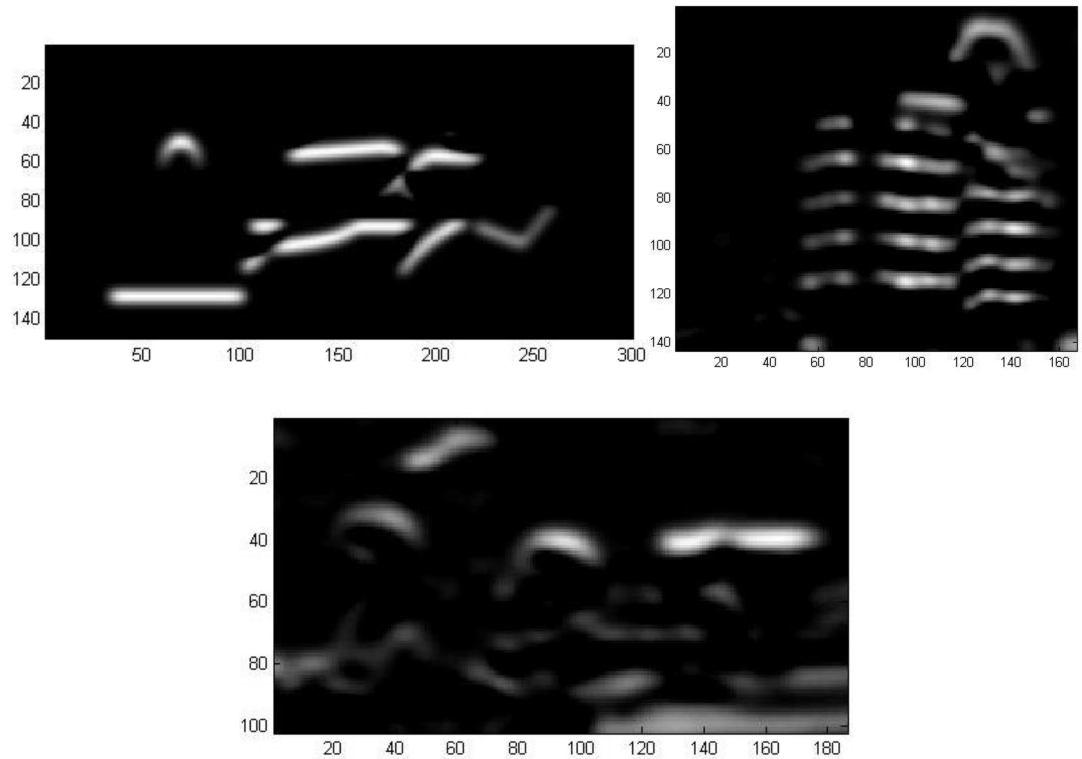
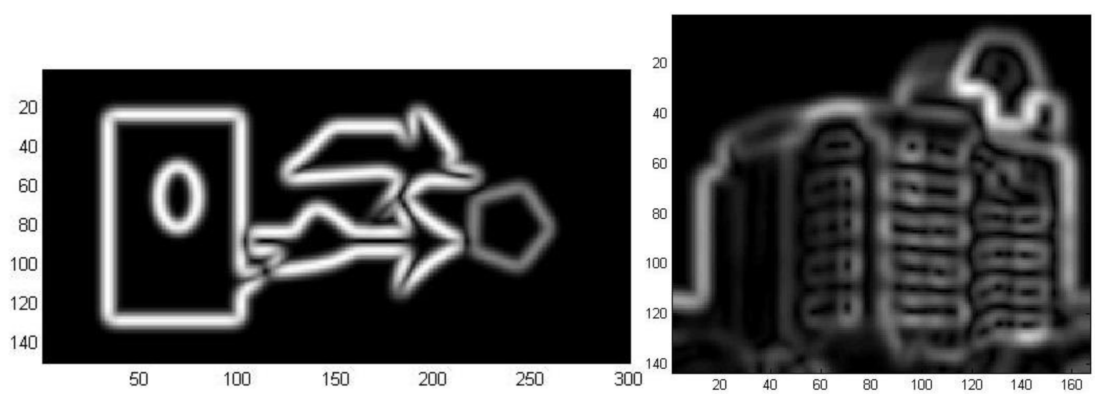


Figure 6: y-gradiants (shapes.raw,dreese.raw,num.raw)

Part IV



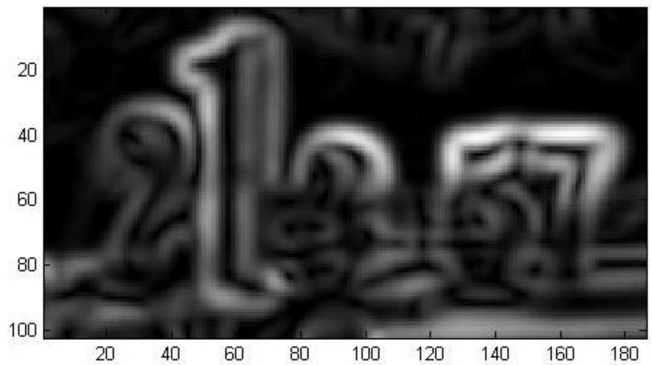


Figure 7: magnitude of gradiants (shapes.raw,dreese.raw,num.raw)

Part V

- Thresholding an image is basically saying keep pixels only above a certain threshold value and kill the rest. Thresholds should always be chosen relative to the image under consideration. As the threshold is increased, obviously we tend to lose more pixels. Increasing the threshold means a stricter cut-off.
- I chose a threshold of 0.3*max(Im). The following is the result on the three images –

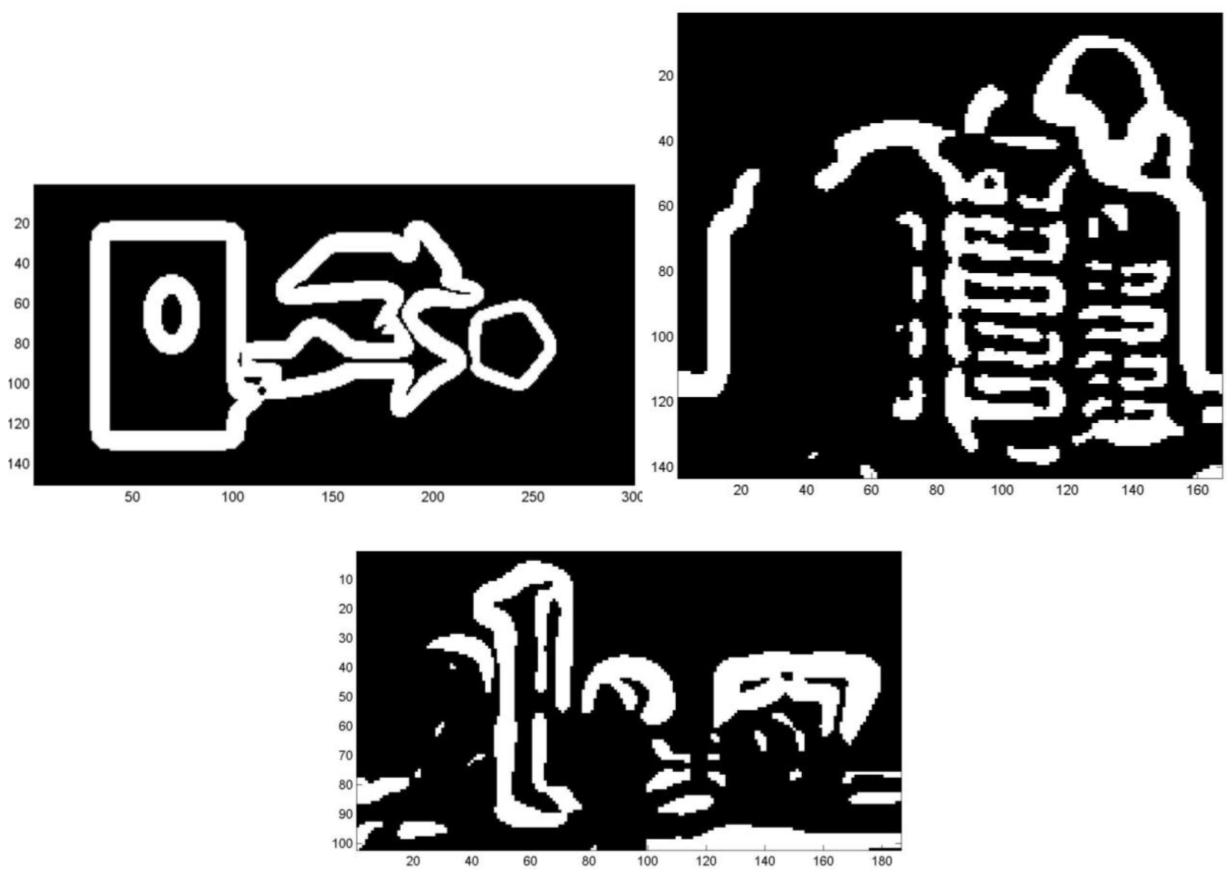


Figure 8: Thresholding (shapes.raw,dreese.raw,num.raw)

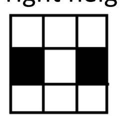
- The edges in shapes.raw hold even as the threshold is increased. The other two images quickly lose information as threshold is increased.

Extra credit:

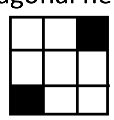
- Canny edge detection consists of 4 steps. These are –
  1. Smooth the image
  2. Compute magnitude and orientation of image
  3. Non-maximal suppression
  4. Hysteresis thresholding
- Steps 1, 2 have been done in Part II-IV of the image. To implement the complete canny edge detector, we need to implement steps 3,4.

Non-Maximal Supression

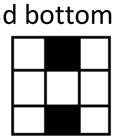
- A search is carried out to determine if the gradient magnitude assumes a local maximum in the gradient direction. In this step, those pixels in the gradient magnitude image which are not maximal (do not have a maximum value) along the pixels in direction of the gradient are suppressed (set to 0). Since there are only specific directions in which a valid pixel exists, I first discretized the direction of magnitudes to be one of 0,45,90,135. Then I used the following algorithm for non-maximal suppression in a 3x3 neighborhood –
  - If gradient direction in 0, check left and right neighbors



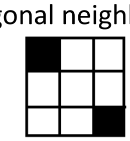
  - If gradient direction in 45, check anti-diagonal neighbors



  - If gradient direction in 90, check top and bottom neighbors



  - If gradient direction in 135, check diagonal neighbors



- Also, for invalid pixel values (where a 3x3 neighbourhood doesn't exist) I let the pixel values be as it is.

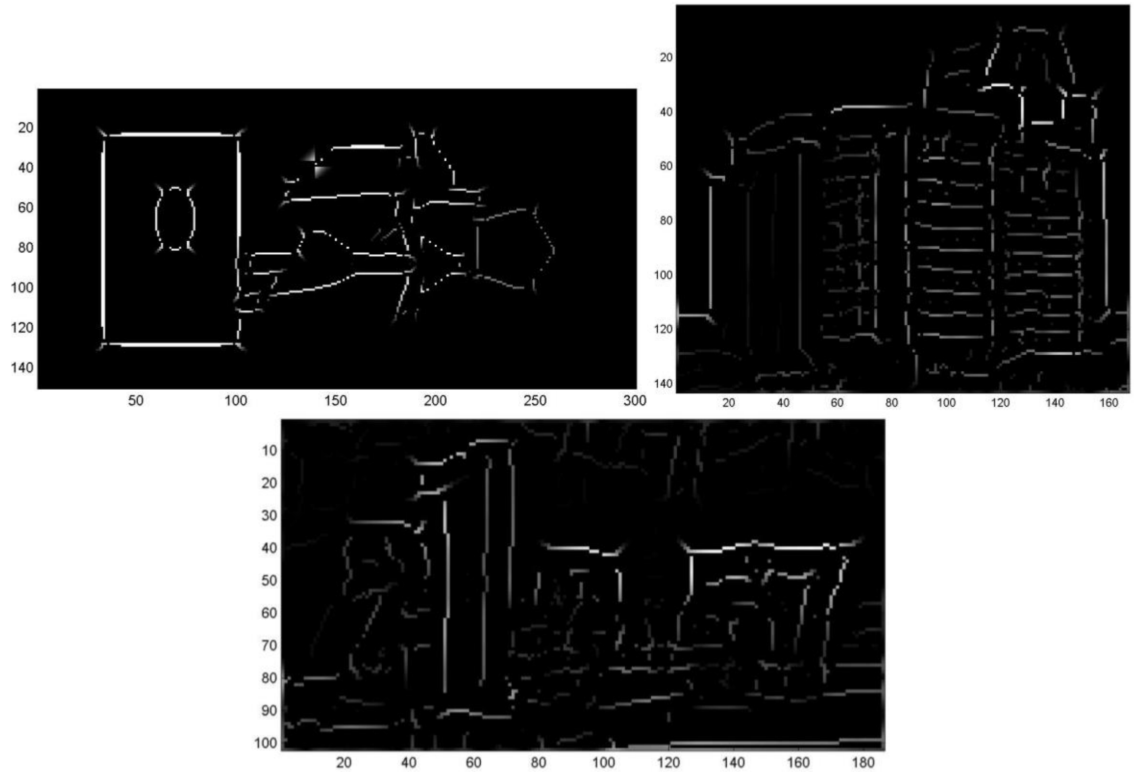


Figure 9: Non-maximal suppression (shapes.raw,dreese.raw,num.raw)

Hysteresis thresholding

- In this step we sequentially follow continuous contour segments.
- We initiate only on edge pixels where gradient magnitude meets high threshold (TH).
- A pixel whose value is less than TL is killed.
- A pixel having value between TH and TL is considered a part of the edge only if it has a neighbor (I chose to search the 8-connected neighbors) which is already a part of some edge.
- Two thresholds are used since a single threshold can cause many broken edge segments.
- The selection of threshold here is tricky. I used TH=0.1*max(Im) and TL=0.4*TH

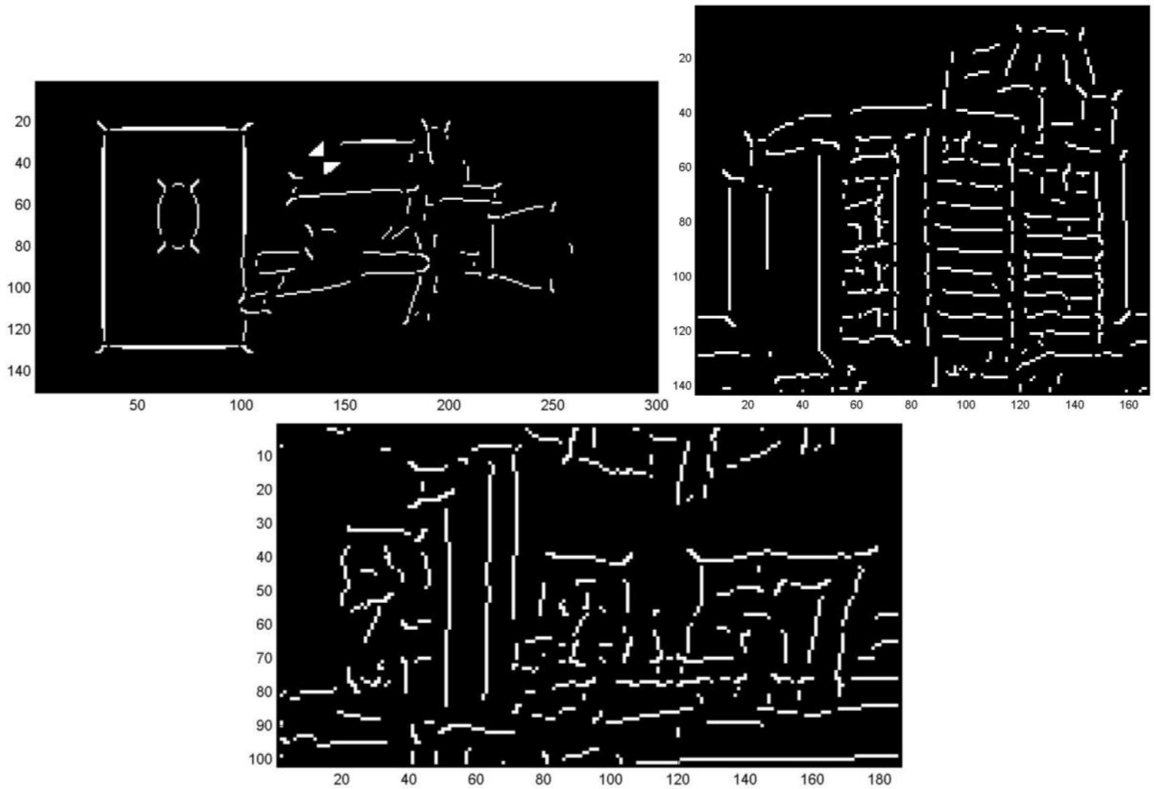


Figure 10: Hysteresis thresholding (shapes.raw,dreese.raw,num.raw)