Edge Detection Comparison Report

University of North Texas

CSCE4240.001

Ruotian Liu

# Introduction

Edge detection is a widely used image processing technique for image segmentation and data extraction. By detecting the changes of brightness in an image, objects and important events are able to be captured. Ideally, edges, the boundaries of objects in an image, can be indicated by set of continuous (connected) curves. Edge detection is important since it is one of the basic steps in processing images.

This project is to implement several edge detection methods as script within Matlab and compare the results by inputting some images. Additionally, four edge detection methods are implemented in this project: Sobel filtering, Canny filtering, Fuzzy Logic Image processing, and a self-designed filtering technique.

# Implementation

All of the methods that are used in this project are implemented using Matlab application. A sample image, called test.png, is used throughout the testing (the sample image is downloaded from Project Files on Blackboard; file name: A LiDAR image).

Figure . Simple Image (test.png)

## Sobel Filtering

The embedded function, edge(), is used with a label ‘sobel’ for this specific task.

* Code simple: edge(img, ‘sobel’).

The result image is showing on the right:

Figure . Sobel Filter

## Canny Filtering

The embedded function, edge(), is used with a label ‘canny’ for this specific task.

* Code simple: edge(img, ‘canny’).

The result image is shown on the right:

Figure . Canny Filter

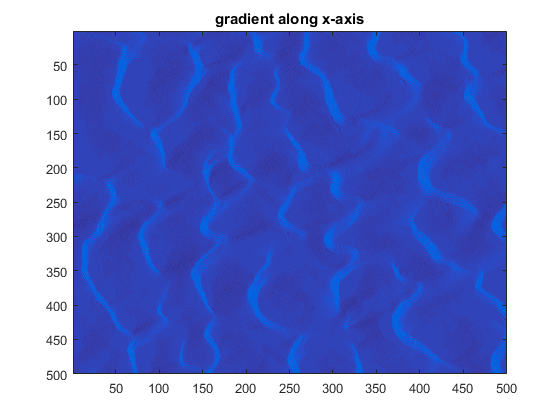
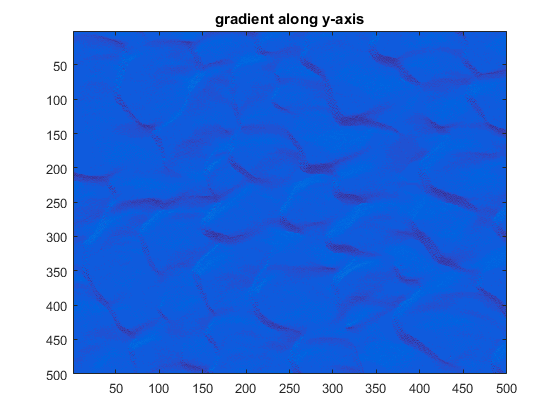
## Fuzzy Logic Processing

Fuzzy logic is a logical system which is an extension of multivalued logic. In the multivalued system, the logic emphasis that not only two possible values, true or false, are counted, but also an extension to classical two valued logic is an n valued logic.

The concept of fuzzy logic is to use set of if-then statements as rules to evaluate the output space which is formatted, or mapped, by the input.

Simple steps and implementation is listed below:

* Make the input image a double array using: im2double()
* Configure gradient on x-axis and y-axis with: [-1 1]
* Calculate image gradient on x-axis and y-axis with: conv2(A, B, ’same’)
* Create new fuzzy inference system with: newfis(’name of the system’)
* Add variables(two variables are needed for this task) to the system with: addvar(system, ‘input’, X gradient, [-1 1]) / addvar(system, ‘input’, Y gradient, [-1 1])
* Add membership functions to the system with: addmf(system, ‘input’, 1, ‘zero’, ‘gaussmf’, [X 0])
* Configure functions and setup rules, such as: rule1 = 'If IMGx is zero and IMGy is zero then IMGout is white'
* Using parser to detect rules and add rules with: char(a, b); and parsrule(sys, addrule)
* Apply the edge detector for each row of pixels to evaluate the output.

Results for steps during processing:

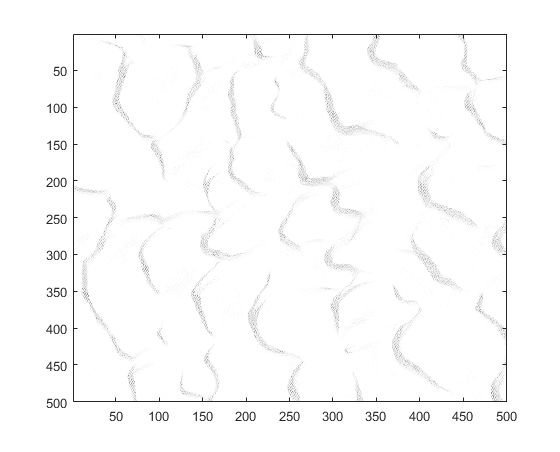


Figure 5. Gradient Along Y-axis

Figure 4. Gradient Along X-axis

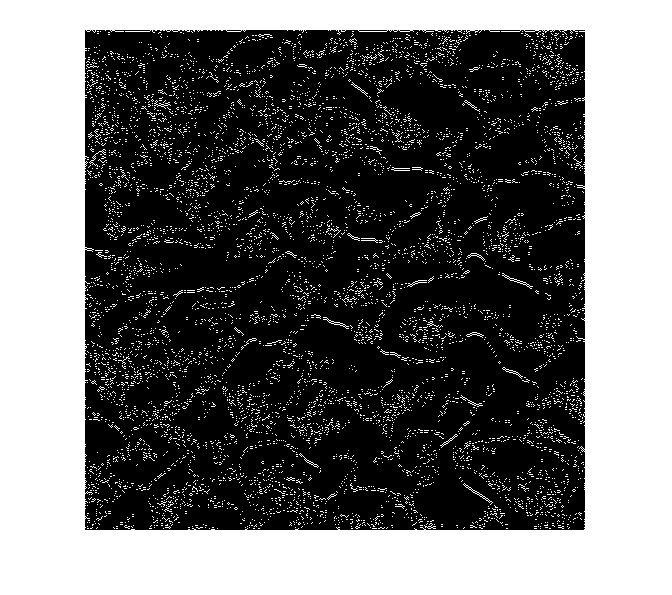
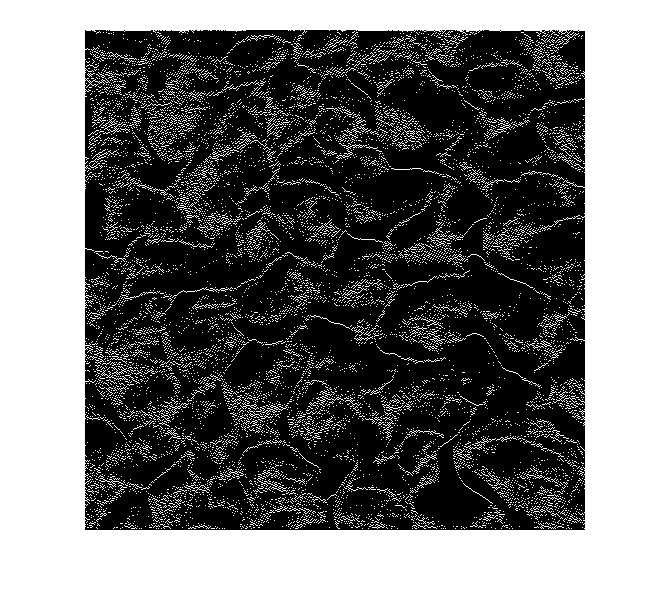
Figure 6. Final Result of FIS

## Self-Designed Filter

For this self-designed filter, techniques like finding peaks, detecting edges, and morphological operations are used.

Simple steps and implementation is listed below:

* Get the size of the original image with: [x,y,z] = size(img)
* Create an empty 2D matrix based on the size of original image with: sample = zeros(x,y)
* Locate local maxima (the peaks) and record their indexes with: [peakValue, indexes] = findpeaks()
* Get size of the indexes array with: a = size(indexes)
* Find and mark the corresponding position of each peaks in the original image with for loop
  + Use fix()to locate specific rows in the empty 2D matrix
  + Use rem() to locate specific columns in the empty 2D matrix
* Filter the marked 2D matrix first time with sobel filter
* Filter the filtered 2D matrix with clean method

Results for steps during processing:

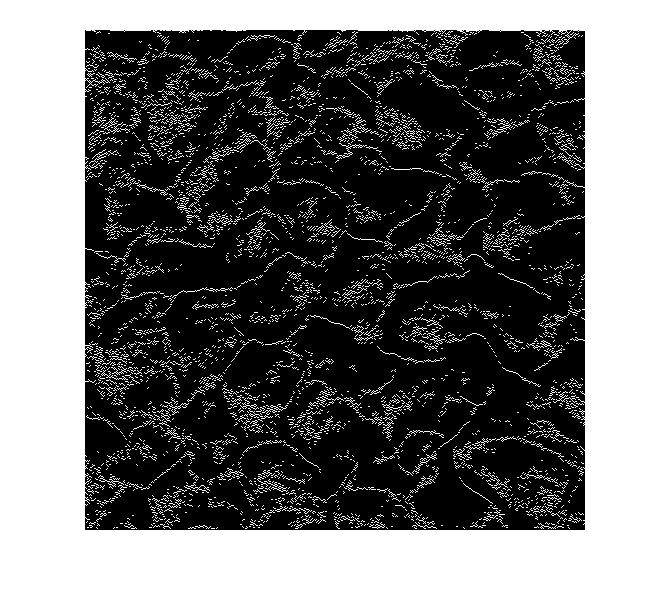


Figure 4. Empty 2D Matrix After Marked

Figure . First Filtering

Figure . Second Filtering

## Conclusion

From the result, it is easy to tell that sobel filtering and fuzzy logic system are good for precise edge detection. Additionally, edges that are generated by sobel filtering are sharper than those are generated by fuzzy logic. Canny filtering is better when detecting general structural information of an object. Our self-designed filter is used to detecting the edges with shadow areas on the image.