EE569 Digital Image Processing

**HOMEWORK #2**

**HOMEWORK – Introduction to Digital Image Processing**

**Issued: 31/01/2020 Due: 15/02/2020**

# Problem 1: Edge Detection (50%)

1. **Sobel Edge Detector (10%)**

**Motivation**

The Sobel Edge Detector applies the gradient of pixel intensity value of image data to detect edge.

**Approach and Procedure**

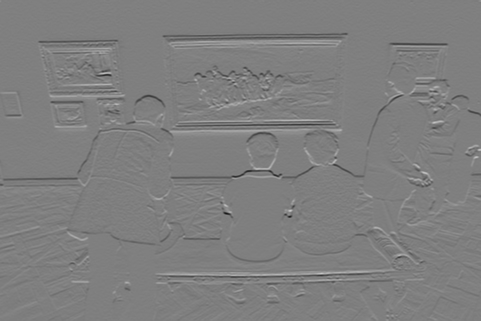
The implementation is simple. After normal procedure of allocation memory for image and

**Results**

**A close up of text on a whiteboard

Description automatically generated**

**Figure 1.1.1**

****

**Figure 1.1.2**

**A close up of a sign

Description automatically generated**

**Figure 1.1.3**



**Figure 1.1.4**

A picture containing photo, standing, outdoor, building

Description automatically generated

**Figure 1.1.5 true magnitude edge map**

A close up of a map

Description automatically generated

A group of people in a dark room

Description automatically generatedA close up of a mans face

Description automatically generated**Figure 1.1.6 false tuned result**

**Figure 1.1.9**

**Figure 1.1.8**

**Figure 1.1.7**

**A close up of a device

Description automatically generated**

**Figure 1.1.10**

A picture containing nature

Description automatically generated

**Figure 1.1.11**



**Figure 1.1.12**

A picture containing outdoor, ground, grass, black

Description automatically generated

**A picture containing tree, outdoor

Description automatically generatedA picture containing outdoor, tree

Description automatically generatedA picture containing outdoor, tree

Description automatically generatedFigure 1.1.13**

**Figure 1.1.14 Figure 1.1.15 Figure 1.1.16**

|  |  |  |  |
| --- | --- | --- | --- |
| **Trials** | **Percentage** | **Vision Performance** | **Reference** |
| **1** | **22** | **Much noise** | **Figure 1.1.7** |
| **2** | **52** | **Background turned to be white** | **Figure 1.1.8** |
| **3** | **202** | **Less edge detected** | **Figure 1.1.9** |
| **4** | **82** | **Pretty nice** | **Figure 1.1.10** |
|  |  |  |  |

**Table 1.1.1**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table 1.1.2**

The false image Figure 1.1.6 and 1.1.4 is obtained by quantization of pixel intensity too early. The quantization of computed pixel value should be in the final step. From the tuning result of the edge detector, the best value of thresholding percentage is around 80. The edge map graph shows clear and less noise. If the threshold value is high, the detail will be eliminated more. The low thresholding gives rise to the noise of the image.

**Discussion**

According to comparison of two image result of edge map of Dogs.raw and Gallery.raw, it is hard to detect the edge of image with more texture inside.

**Answers**

1. **Canny Edge Detector (10%)**

**Motivation**

**Approach and Procedure**

**Results**

**Discussion**

**Answers**

1. Flow chart for algorithm

Filter the noise

Compute the gradient magnitude

Apply non-maximum suppression to obsolete small local magnitude

Use double threshold to determine the edge

Finalize by suppressing all other weak and unconnected edges

1. **Structured Edge (15%)**

**Motivation**

**Approach and Procedure**

The setup for program needs some computer science experience.

*mex -DMX\_COMPAT\_32 private/edgeBoxesMex.cpp -outdir private*

The above command template is used to compile the C++ code for MatLab software. The difference with the old version of Matlab is the command that has additional option -DMX\_COMPAT\_32 to compile the code. Another step to run the program is to add the path of toolbox of *piotr\_toolbox*. Before addition of the folder of the toolbox, the subfolder paths of toolbox should be generated by Matlab function *genpath* to ensure all tools that will be used by Structure Edge program.

The *edgesDemo.m* file is revised to fit the requirement. The *writeraw()* function is used to get the image data of the edge image data. The edge detection file of *Dogs.raw* and *Gallery.raw* is stored as the *GalleryEdge.raw* and DogEdge.raw as a source file for performance evaluation problem. The additional transformation of edge data is implemented by formula below.

(1.3.1)

The formula ensures the *EdgeImageData* has the same graph of the comparison of the original image and edge image of Structured Edge implementation.

**Results**

**A group of people in a room

Description automatically generated**

**Figure 1.3.1 the origin image generated by Structure Edge Detector**

**A close up of text on a white background

Description automatically generated**

**Figure 3.3.2 the obtained image generated by default parameters of Structure Edge**

**A cat with its mouth open

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

**Discussion**

**Answers**

1. **Performance Evaluation (15%)**

**Motivation**

There are four basic elements to construct the performance quality measurement of edge detection. They are true positive, false positive, false negative and false positive.

**Approach and Procedure**

**Results**

**Discussion**

**Answers**

**Problem 2: Digital Half-toning (50%)**

1. **Dithering (15%)**

**Motivation**

The dithering algorithm is used to place black dots corresponding to true distribution of black dots of original image.

**Approach and Procedure**

**Results**

**Discussion**

**Answers**

1. **Error Diffusion (15%)**
2. **Color Halftoning with Error Diffusion (20%)**