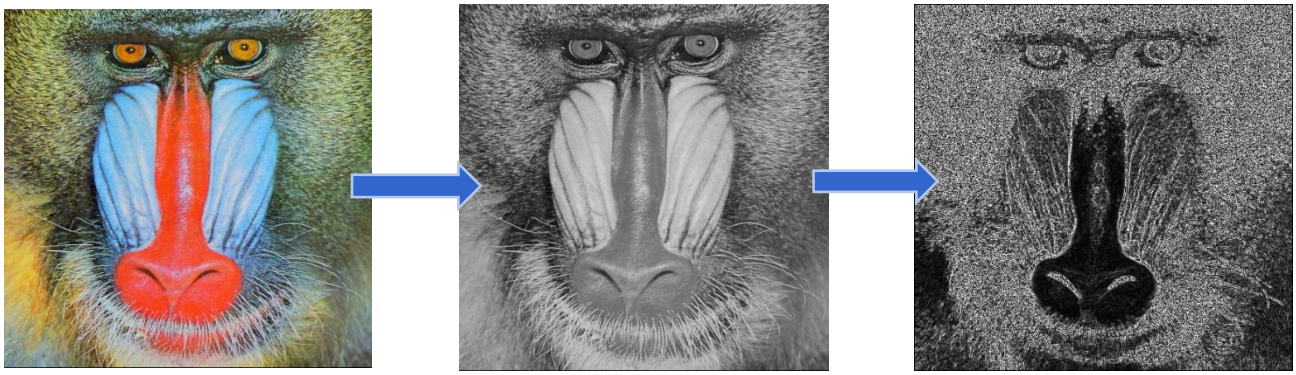
UNIVERSITY OF TEHRAN
Electrical and Computer Engineering Department
Object-Oriented Modeling of Electronic Circuits, Spring 1401
Computer Assignment 4
Image Processing and Use of Channels

In this assignment, you are to design an image processing system with edge detection and median filter. Edge Detection filter is commonly used as a first step in procedures to define discrete objects (such as buildings or agricultural fields) within images and median filter is the filtering technique used for noise removal from images.



Original image | Grayscale image | Edge detection result

Fig.1) Edge detection filter

Fig.2) Median filter

This assignment has two parts. In first part, an edge detection filter is developed and in second part median accelerator will be added to the system.

Part 1) This system comprises of four modules (as shown in fig. 3). The first one (*File Reader*) is responsible for opening RGB text files and send pixels to *Gray Scaler* via a channel. The RGB pixel colors' values of input picture are written in three separated files, and the picture size is 512×512. This module reads all pixel values from external image files and stores them in separated

arrays. Every pixel has three colors, (Red, Green, and Blue) each specified by an 8-bit word. Use an 8-bit channel and send each value in one clock cycle, so one pixel's data is sent in 3 cycles.

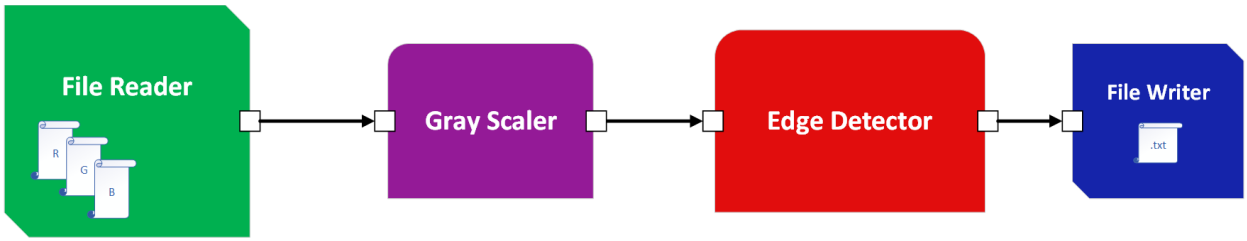


Fig.3

Below is a summary of steps taken by the *File Reader*:

a. Read {red_pixel, green_pixel, blue_pixel}.txt files and store their values in separate arrays.
b. Send first pixel's data to *Gray Scaler*.
c. Go to next pixel until pixels are not finished.

The *Gray Scaler* converts an RGB image to grayscale using the following *average method*. The grayscale equivalent is the average of the three colors. This module read every 3-bytes from channel and calculate grayscale value.

$$grayscale = \frac{R + G + B}{3}$$

The image is divided into 64 segments each of which consisting of 8 image rows. The *Gray Scaler* sends a segment that has been processed to the *Edge Detector*. This is done through an 8-bit channel that connects the two accelerators. While *Gray Scaler* is waiting to a segment sent to the *Edge Detector* to be received, it processes the next segment of data. The channel that you develop handles this transfer of data.

Below is a summary of steps taken by the *Gray Scaler*:

a. Read first segment of the image from the channel.
b. Grayscale the received segment based on the *average* method.
c. Send the gray scaled segment to *Edge Detector*.
d. If segments are not finished, return to step a.

The third module, named *Edge Detector*, is responsible for detecting edges in the grayscale image. It convolves the received segment of the image with a of 3×3 convolution kernels to detect the edges (as shown in fig.4).

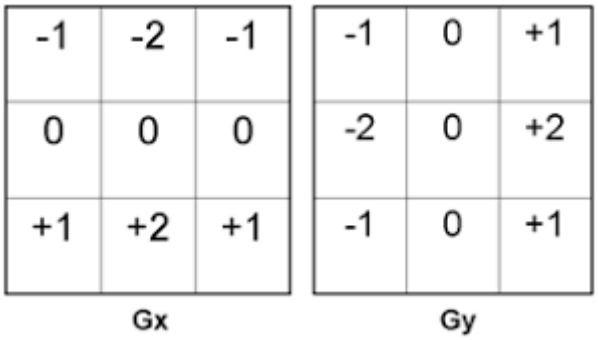


Fig.4

These kernels are designed to respond maximally to edges running vertically and horizontally. The kernels can be applied separately to the input segment, to produce separate measurements of the gradient component in each orientation (call these *Gx* and *Gy*). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude is given by:

$$|G| = \sqrt{Gx^2 + Gy^2}$$

Typically, an approximate magnitude is computed using:

$$|G| = |Gx| + |Gy|$$

which is much faster to compute.

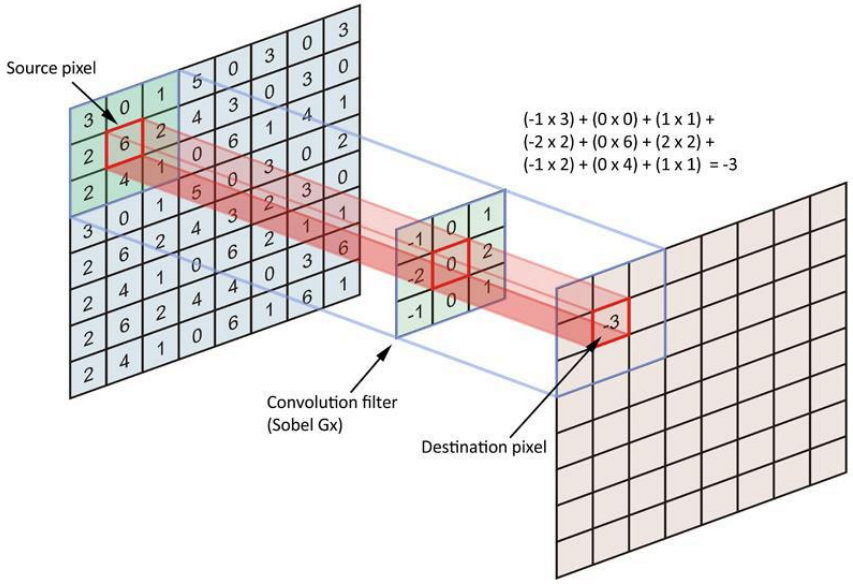


Fig.5

After receiving a segment of a picture, this module performs matrix convolution, as depicted in fig.5. When the processing reaches a row that belongs to the next segment of an image, the *Edge Detector* requests another segment from the channel connecting the two accelerators. The data of the previous segment that is no longer needed will be overwritten and processing of edge detection continues. This continues until all data have been processed by the *Edge Detector*. each completed segment will be sent to last module named *File Writer*.

Below is a summary of steps taken by the *Edge Detector*:

a. Get the first segment.
b. Perform convolutions and save the results internally. (consider $|Gx|+|Gy|$ for this assignment)
c. Send the processed segment to next module.
d. Continue receiving the segments until all segments have been received.

The *File Writer* reads pixels in bytes and from an 8-bit channel which connects *Edge Detector* and *File Writer*. The result data will be dumped to an external file at the end of the processing. Use *Python* files uploaded on *elearn* to save output image.

Part 2) Bonused Part

In this part you are to add *Median* accelerator to your system (as shown in fig.6). This module reads each byte from channel and process every segment due to the algorithm shown in fig.7.

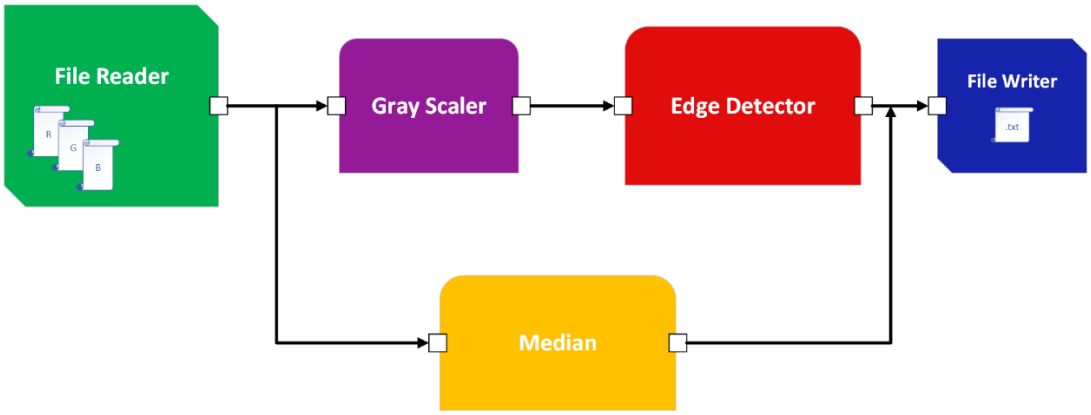


Fig.6

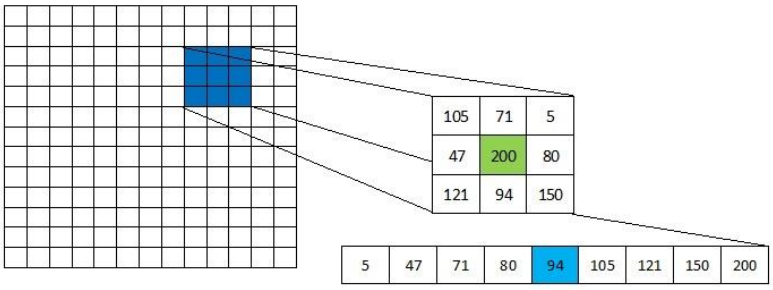


Fig.7

After processing each segment, it writes pixels on the 8-bit channel between *Median* and *File Writer*. *File Writer* will start reading this data after completion of saving edge detection values and it will output the data in three different text files for red, green and blue values. Add new signals to part 1 system in order to handle both filters (e.g., start or ready port).

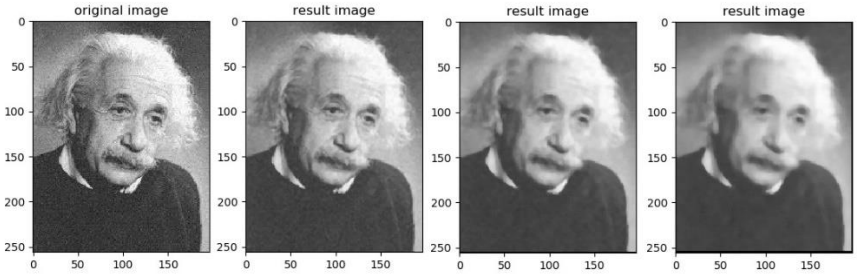


Fig.8) Median filter result

Deliverables:

a) Generate a report that includes item discussed below:
   1) Explain your design decisions completely.
   2) Show result images. (Edge detection result for part 1 and median result for part 2)

b) Include your codes in a folder beside your report.

Make a .zip file and name it with the format shown below:

LastName_Studentnumber_CAnn-ECE01