0301-346: Project I
Image Processing using C++
Spring 2020

I. Introduction

In this project, you will be doing some image processing using C++. As an introduction to image processing using C++ you will be implementing some simple processing algorithms. You will be required to implement the following image processing routines.

- *Copy* – Make an exact copy of the original image.
- *Vertical Flip* – Flip an image about its center horizontal axis.
- *Horizontal Flip* – Flip an image about its center vertical axis.
- *2DMedian Filter* – Apply a median filter to a noisy image to reduce the noise.

Images used in this project will be provided in PGM (Portable Gray Map) because of their ease of reading and writing. Code is provided to handle reading and writing of the PGM files where all image processing procedures will be done on images that have been loaded in memory of your program.

II. Background

A. PGM Files

A PGM file is grayscale image format and is designed to be easy to read and write. Its file format provides encoding information, file comments, width/height, and max pixel value. Each file contains a header specifying information of the file followed by the image data itself. Below is an example of a PGM file (FEEP.pgm)

```
P2
# feep.pgm
24 7
15
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  3  3  3  3  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0 15  0
0  3  3  3  0  0  0  7  7  7  0  0  0 11 11 11  0  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0  0  0
0  3  0  0  0  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

Data provided by http://netpbm.sourceforge.net/doc/pgm.html,

One of the easiest ways to open and view a PGM file is to use the program IrfanView which can be downloaded from https://www.irfanview.com/. The viewer only supports Windows, for other platforms you may need to convert the PGM files to another format prior to viewing them.

B. Median Filter

The median filter is a non-linear digital filtering technique. It is commonly used in noise reduction and is highly effective at the removal of “salt and pepper” noise. The main concept of the median filters is to create a window of neighbors. Within a single window, the middle neighbor is replaced with the median value of all the neighbors in the window. The median filter is executed by sliding the window across the entire signal.

For this project, you will be implementing a 2D median filter and applying it to an image. In the 2D case, the window is n x n and it slides not only from left to right but also down. The 2D window concept does have edge boundary cases. For this project, the edges of the image are to be left as original values for simplicity. Figure 1 below, shows a visual of a 3x3 window for the median filter. A method for determining the median value in the window is to flatten the window into a 1D array, sort the values and pick the middle value.

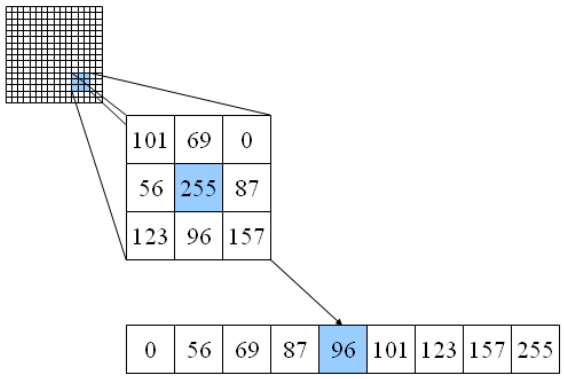

Figure 1 - Visual of median filter

0301-346: Project I
Image Processing using C++

Spring 2020

III. Getting Started

A. Provided Code

To get this project started, some code has been provided to assist you with working with PGM files. A series of functions calls and variables are available to read/write and obtain image information. Below are the function prototypes and variable provided, these can be found in PGM.h, with implementation code in PGM.cpp (see header file at end). Below is a code snippet to get your project started.

```
#include <iostream>
#include "PGM.h"
using namespace std;
int main()
{
    string fileName = "lena.pgm";

    //Open File set information
    if( openPGM(fileName) )
    {
        // Get Image Size Information
        int width = getPGMWidth();
        int height = getPGMHeight();

        // Declare and allocate memory for data
        int** original;
        original = new int*[ height];

        for(int i = 0; i < height; i++){
            original[i] = new int[width];
        }

        // Get the data
        getPGMData(original);

        // Write back out the same image
        writePGM("same.pgm", original);

        // Clean up memory
        for(int i = 0; i < height; i++){
            delete[] original[i];
        }
        delete original;
    }
}
```

0301-346: Project I
Image Processing using C++
Spring 2020

IV. Project Requirements / Scoring

A. [5pts] - Minimal files to be created
   i. main.cpp
   ii. ImageProcessing.h
   iii. ImageProcessing.cpp

B. [65pts] - Functions to be prototyped and implemented
   i. [5pts] copyImage
   ii. [15pts] flipVeritcal
   iii. [15pts] flipHorizontal
   iv. [30pts] medianFilter (9x9 window, leave edges)

C. [20 pts] - Program Execution

The program execution portion of the grade is all or nothing. If the program flow does not work with automated input then you lose this portion of the grade, because other aspects of the project will have to be graded by hand to achieve partial credit. Please follow the steps exactly:

1. Using the console prompt user for name of file.
2. Then prompt for the operation to be done.
3. Then prompt user for output file name.
4. Perform the requested operation.
5. Write out the file.
6. Exit program or ask user to start over (starting over may be tricky)

See example program input/output at the end of this document.

D. [10pts] - Code Style
   i. All prototyping done in header files
   ii. All implementation done is cpp files
   iii. Separate main file
   iv. Comment functions in header files

0301-346: Project I
Image Processing using C++
Spring 2020

V. Sample Usage

```
Enter Original File Name: lena.pgm
Reading in File
Done Reading in File
File Successfully Opened
Select Operation:
        (0) Copy Image
        (1) Flip Vertical
        (2) Flip Horizontal
        (3) Median Filter
Enter Selection: 0
Enter Save File Name: cpy.pgm
Performing Operation...
Writing out the File...
Cleaning up now!
Clean-up finished.
Perform another operation [y/n]? y

Enter Original File Name: noisy.pgm
Reading in File
Done Reading in File
File Successfully Opened
Select Operation:
        (0) Copy Image
        (1) Flip Vertical
        (2) Flip Horizontal
        (3) Median Filter
Enter Selection: 3
Enter Save File Name: filt.pgm
Performing Operation...
Writing out the File...
Cleaning up now!
Clean-up finished.
Perform another operation [y/n]? n
```