Assignment 1: Introduction to Basic Image Processing and ROI Manipulation

Igli Duro
Department of Computer Science & Engineering
University of South Florida
Tampa, FL

I. Introduction

For this assignment we implemented basic image manipulation techniques. Along with the ability to operate within specified regions of interest (ROI) within an image. An overview of techniques implemented are a color brightness, color visualization and two different image smoothing algorithms. In addition to that, two other functionalities, add and binarize are also used in this report. These algorithms were provided in the original sample code given to work with. This report is divided into the following sections. Section 2 will give a description of each of the algorithms. Section 3 will go in depth for how each algorithm was implemented. Section 4 will discuss results and analysis. And Section 5 will be our conclusion.

II. Description of Algorithms

In this section, in-depth explanations of the implemented algorithms are given. We'll first begin by discussing the color brightness and visualization algorithms, which are used to manipulate color on an image or add color by comparing it to a threshold. Afterwards we discuss the two smoothing algorithms.

A. Color Brightness

Color brightness is a method of manipulating the color channels with an image. For a given .ppm file, there is data containing the RGB composition of each pixel. That is, the amount of red, green, and blue that is contained on a scale from 0-255. The color brightness filter adjusts these channels by taking in 3 values that act as multipliers. For example if we want to make an image more blue we can take an input of ”1 1 2”. Then we retrieve the RGB values of each pixel and multiply the current value by the input value. In this case the red and green channels stay the same while the blue channel get doubled.

B. Color Visualization

Color visualization is a method of adding color to a grayscale image. Which takes pixels of similar intensity to an input and colors them red. This algorithm takes in two values, a threshold, and a desired intensity value. If the absolute value of the input value subtracted by the pixel intensity is less than the threshold. Then the pixel has its red channel set to max for the output image. For example, if our threshold is 15 and our input intensity is 75. Then any pixel with an intensity value between 61 and 89 is colored red.

C. Regular 2D Smoothing

Regular 2D smoothing is a method of filter used to blur an image. This method uses uniform smoothing and takes in a window size as input. This window size, ”ws”, groups pixels into ws by ws neighborhoods and the intensity value of the pixels inside the neighborhoods are summed up and averaged out by diving the sum by ws squared.

D. Separable 2D Smoothing

Separable 2D smoothing is a another method of blurring an image. This method operates differently than regularly 2d smoothing but achieves a similar result. Separable smoothing operated by doing two passes through the source image. On the first pass through, pixels are averaged out in rows according to the window size. Once the first pass is complete these new values are then used for the second pass, which averages out the intensity values in columns in window size groups.

III. Description of Implementation

This program was developed using C++, several files and some starter code were provided and built upon. The most important of these files were the image.h and image.cpp files which presents the image class and some tools to read in information, manipulate and write out to and from image files. Inside the utility.h and utility.cpp files are the algorithms used to operate on image files. And inside iptool.cpp is our driver code. The driver code reads in information from a parameters.txt file which contains the following information: the source file, target file, amount of ROIs to process, the starting x and y pixel for the ROI, the size of the ROI, the function to use on the ROI, and the parameters for the function. If multiple ROIs are specified, then multiple sets of starting pixel, size, function and parameters are required. The full details regarding this format are contained in the Readme.txt file.

To process the ROIs. A data structure containing information about the ROI is used. And an array of size equal to the number of ROIs is created. Along with this a temp image file is created along with an array of images to store our ROIs after we're done processing them. The temp file is used to store a copy of the source image for reference. Before reaching a function to process the image. We copy data within the ROI from the temp image and pass that off to be processed. Then at the end, the temp file is copied to the target file, and the

ROIs get written over that. Leaving us with an image with only desired regions having been modified.

IV. Description of and Analysis of Results

A. Description of Results

This section will go over the results of the algorithms implemented. We first look at the result of defining 3 ROIs and using the previously established binarize and add filter. Then we look at and compare the results of the 2D smoothing filters. Next the color brightness and color visualization filter. And lastly we look at the results of overlapping ROIs.

Figures[1-2] show the results of processing 3 ROIs on a grey-scale image next to the original image. From here we see that the filters used to process the image inside the ROIs are self contained. Showing that our algorithm for establishing regions of interest is operating correctly.

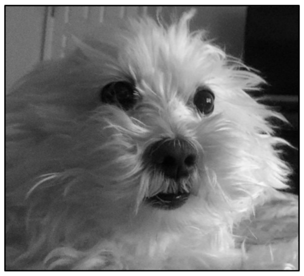
Figure 1 Original "dogFace.pgm" file

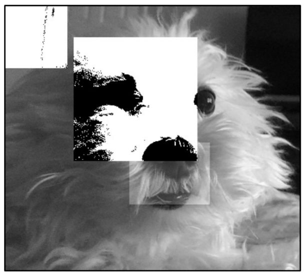
Figure 2 "dogFace.pgm" with 3 ROIs, 2 using a "binarize 75" and one "add 50" filter

Next in Figures[3-4] we see the results of both smoothing filters. With regular smoothing being on the left and separable smoothing on the right. For this operation the ROIs were kept the same to observe any differences between filters. And we can see that the results of the algorithm are the same. However, the filter does not achieve the desired blurring effect. We believe that while the algorithm operates in the correct bounds there was a missed step when averaging a windowed area. Resulting in the image simply being darkened instead.

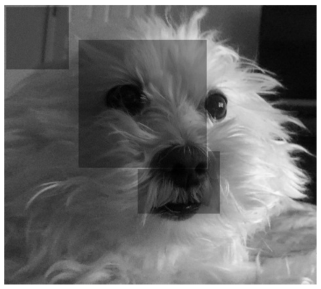
Figure 3 "dogFace.pgm" using the regular 2d smoothing filter across 3 ROIs

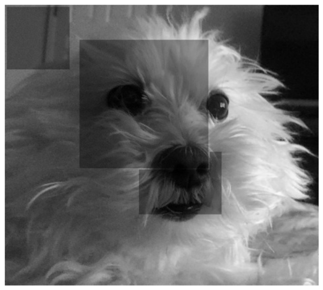
Figure 4 "dogFace.pgm" using the separable 2d smoothing filter across 3 ROIs

In Figures[5-6] we see the result of the color brightness filter next to the original image. The parameters to achieve the colors in the 3 ROIs were (0 0 2), (2 2 0), and (3 1 3). From this we can see the results were as expected. With the first ROI nullifying the red and green channels and multiplying the blue to produce a blue filter. And similarly with the second and third ROI to increase the red & green channels to produce yellow and the red & blue channels to produce magenta respectively.

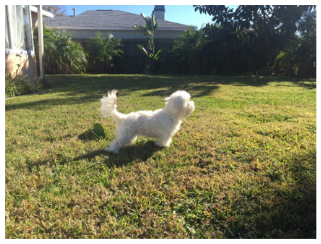
Figure 5 Original "dogYard.ppm" file

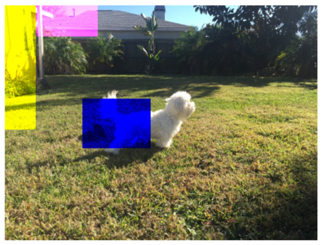
Figure 6 "dogYard.ppm" using the color brightness filter to alter 3 ROIs

For Figures[7-8] we see the result of the the color visualization filter. For this image the parameters were the same across each ROI. With the intensity being 75 and the threshold being 10. This would mean any values between 66 and 84 are set to red. This would translate to darker gray tones which line up with what portions of the image were dyed red.

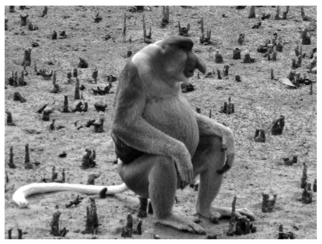
Figure 7 Original "proboscisSit.pgm" file

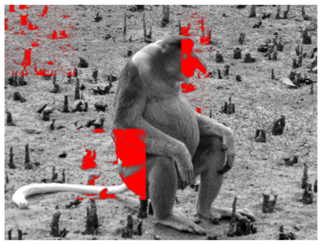
Figure 8 "proboscisSit.pgm" with a color visualization filter across 3 ROIs

Lastly in Figures[9-10] the goal was to prevent overlapped ROI regions from compounding on one another. With the first defined ROI region taking priority and the following ROI regions not operating inside them. We can see a binarize and attempted smoothing overlapping with each-other. However in the parameters for this image the 2D smoothing was defined first and takes precedence over the binaraize process.

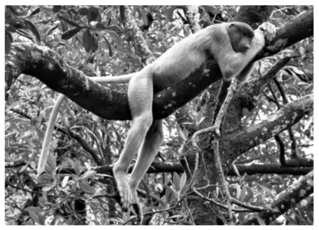
Figure 9 Original "proboscisTree.pgm" file

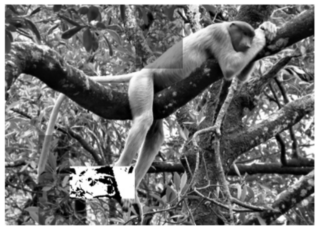
Figure 10 "proboscisTree.pgm" with overlapping smoothing and binarize filters

V. Conclusion

For this assignment, we implemented four additional image processing algorithms into our program. Along with the ability to define regions of interest to process multiple subsections of the same image in a self contained manner. From these newly implemented algorithms we gained the functionality to modify color values in an image. As-well as add color to pixels of a gray-scale image that are of similar intensity to a given input value. Along with this were two smoothing algorithms. And while they were not functioning as desired, they did operate within the desired bounds and we've built some understanding of what may have been missing to achieve full functionality.