Medical Imaging Software Report

Submitted as part of the requirements for Assessment 2 of the ACSE-5 module.

Developed by Deborah Pelacani Cruz and Hugo Coussens.

Introduction

Image processing is a widely utilised tool in the analysis of medical images due to its ability to highlight subtle features that would otherwise be unnoticeable to the naked eye. The aim for this project was to develop a C++ software that was able to read and write images of different file types, and apply a range of filters for image enhancement. We have developed a "per pixel" algorithm that is able to manipulate individual pixel properties and apply a filter kernel that produces different filtering effects. The software utilised two libraries for reading and writing BMP[1] and PNG[2] files. The algorithm to read and write PPM files was developed from scratch along with all other pixel manipulation functions.

Software Structure

The structure of the software consists of an *Image* class and three derived classes, one for each supported format. Virtual functions are *LoadImage*() and *SaveImage*(), which are unique to each type of image. The bulk of the image processing is composed by non-virtual functions of Image. This approach was chosen to guarantee that the filters would be applicable to any image format without the need of repeating the filter functions in each derived class. To implement this approach, *Image* stores a vector of *Pixels* class instances which correspond to each pixel of the loaded image. All applied filters modify each individual pixel object. Once all user-chosen filters are applied, the saved image then correspond to the modified pixels of the original image. The *Interface* class was created to implement the user interface. Making *Interface* a class makes it easier to prevent the user from giving the program invalid or bad inputs.

Using the Software

The software is packaged by an easy to use command line interface. The user is walked through the key stages: **Select image type-> Load Image -> Apply filter-> Choose strength-> repeat last 2? -> Save image to file.** The *strength* parameter represents a scaling factor to the convolution kernel for the "edge detection" and "sharpening" filters. For the blurring filters, it represents the amount of times a fixed-strength blurring filter will be applied to the image. A clear example of the array of different image filters are presented in the file “demonstration.png”. The software currently accepts PPM (P3 only), BMP (24-bit only) and PNG (24-bit only) formats. We suggest the convenient online image format converter to transfer to and from BMP(https://online-converting.com/image/convert2bmp), and GIMP for generating and viewing PPM images.

Limitations and Further Work

The software provides space for performance and utility improvements. Currently, only 24 bit BMP and PNG, and ASCII PPM images are supported, but with appropriate libraries, these categories could be expanded. The convolution kernels available are limited to a number of four and refer to a 3x3 kernel. More complex kernels can easily be implemented in the code.

In terms of performance, the software contains loop redundancies and a choice of variable types that are likely to slow down its execution and prevent the compiler from performing appropriate optimisation (more detailed comments on performance can be found throughout the source code). Performance is particularly poor when loading an image and when applying the convolution kernel to each pixel.

References

1. Partow, Arash. 2002. Bitmap Image Reader Writer Library (version 1.0). C++, http://partow.net/programming/bitmap/index.html
2. S. Barrett, STB Library, (2013), GitHub repository, https://github.com/nothings/stb.