CS 361 - Homework 05

Professor Mark W. Boady

1 Overview

Implement a threaded generation of the Mandelbrot set. The algorithm to generate the set is described in Lecture materials. How generate and manipulate images is also covered in a set of lecture material.

2 Programming Assignment

Develop a command line program in C++ that generate the Mandelbrot set as a Bitmap image. The program will take exactly 5 command line arguments. The compiled program will be named `mandelbrot`. The inputs are

1. x_1 - A **long double** representing the smallest x to display
2. x_2 - A **long double** representing the largest x to display
3. y_1 - A **long double** representing the smallest y to display
4. y_2 - A **long double** representing the largest y to display
5. filename - The name to save the file as.

Use a threshold of 100 to test all points.

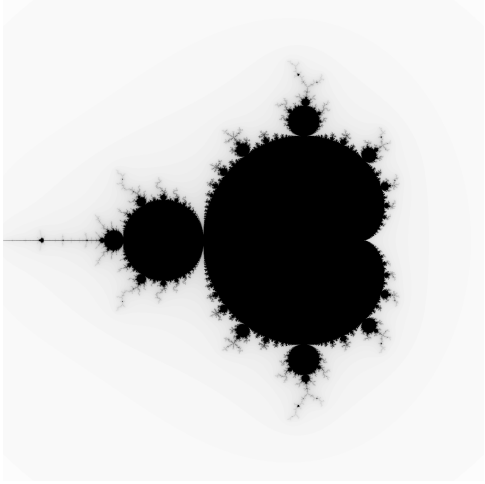
The **width** of all images should be 1500 pixels. The **height** should be determined by the number of points on the y axis. The image needs to scale if the x and y are not of equal length. Base the scaling on the fact that the width is always 1500 pixels.

You will make a 24 color image (1 byte red, 1 byte green 1 byte blue). The image's dpi will be 150.

For example, we could call

./mandelbrot −2 1 −1.5 1.5 full.bmp

to generate the image shown below (your colors do not need to match, but the shape must be accurate)



The program will exit with an error message on bad command line inputs. Otherwise, it will generate an image and exit.

You **must** use threads to generate the pixels of the image. You may decide on the number of threads yourself, but it must be **at least** 4.

You may chose your own color scheme. You may use greyscale or a variety of colors. The color must be based on the when you detected divergence or when the point was in the set.

You **must** follow design requirements listed below:

- A **manager** thread must assign (x, y) points to **worker** threads.
- All **worker** threads must compute the color of points they are assigned.

The rest of the implementation details are up to you.

2.1 Implementation

You are expected to write professional code. Use good variable and function names. Provide comments explaining how the code works. Documents each function in the header file with comments. You will be graded on style as well as functionality.

2.2 Citations

If you use any outside resources, talk about algorithm design with other students, or get help on assignments you **must** cite your resources in the comments. Uncited sources are an Academic Honesty Violation. Cited sources may lead to a deduction depending on the amount of code used, but will not violate Academic Honesty Policies.

You are expect to write the majority of the code yourself and use resources for things like looking up commands. For example, if you forgot how to test if a file can be opened for reading you could look it up and cite a source. If you copy a critical algorithm and cite the code, you may still get a deduction for not developing the code yourself.

2.3 Makefile

You **must** provide a makefile to compile your code. We will type make mandelbrot and it **must** build your program. If there are any compile errors or a makefile is not provided we cannot test your code.

These images are shown in the lecture slides.

You must have the following make targets:

1. make mandelbrot - Builds the Program
2. make e0.bmp - ./mandelbrot −2 1 −1.5 1.5 e0.bmp
3. make e1.bmp - ./mandelbrot −0.5 0 0.3 1.2 e1.bmp
4. make e2.bmp - ./mandelbrot 0.3 0.4 0.6 0.7 e2.bmp
5. make e3.bmp - ./mandelbrot −0.2 0.0 −1.0 −0.9 e3.bmp
6. make e4.bmp - ./mandelbrot −0.05 −0.01 −1.01 −0.97 e4.bmp
7. make clean - Remove images and compiled code.

2.4 Other Requirements

If you submission does not meet the following guidelines we will not be able to grade it.

- You **must** use the C++ 17 Standard threads. No other thread libraries (pthreads, boost, etc) may be used. `https://en.cppreference.com/w/cpp/header/thread`
- Code **must** run on tux and be compiled with g++.
- All code **must** compile using the C++ 17 or above standard. (−−std=c++17)
- All code **must** be submitted in a single zip file. (No tar.gz, rar, etc)
- A working makefile **must** be provided.
- You may use libraries in the C++ 17 standard unless noted elsewhere in the instructions. `https://en.cppreference.com/w/cpp/header`
- Your code **must** compile. You should always submit stable code, we will not debug code that does not compile.

3 Grading

This homework is worth 100 points.

Question 1 : 15 points
Main Program
(a) (5 points) Handles Command Line Arguments
(b) (5 points) Error Message on Bad Command Line Arguments
(c) (5 points) Overall Style

Question 2 : 20 points
Bitmap Header
(a) (10 points) Writes Bitmap Header corrrectly
(b) (10 points) Writes Bitmap DPI information correctly

Question 3 : 10 points
Mandelbrot
(a) (10 points) Computes value of z at given point $x + yi$

Question 4 : 50 points
Image Generation
(a) (20 points) Manager Thread Design
(b) (30 points) Worker Thread Design

Question 5 : 5 points
Following Instructions
(a) (5 points) Single Zip file with all required files submitted