

William Taylor | Compute Games Technology | February 12, 2017

Mandelbrot CUDA Assignment

coursework part 1

Table of Contents

[Introduction 2](#_Toc474702220)

[Problem 2](#_Toc474702221)

[Hardware 2](#_Toc474702222)

[Software 2](#_Toc474702223)

[Goals 3](#_Toc474702224)

[CPU Analysis 3](#_Toc474702225)

[Analysis 3](#_Toc474702226)

[Timings 3](#_Toc474702227)

[Profiling 3](#_Toc474702228)

[GPU Implementation 3](#_Toc474702229)

[Port Overview 3](#_Toc474702230)

[Optimisations 3](#_Toc474702231)

[Timings 3](#_Toc474702232)

[Project Results 3](#_Toc474702233)

[Testing 3](#_Toc474702234)

[Comparison 3](#_Toc474702235)

[Conclusion 3](#_Toc474702236)

# Introduction

## Problem

A snippet of code that renders a section of the Mandelbrot set into an image was given. However, the code is sequential and the code could be much improved by using parallel compute to generate the image. In this report I set out the hardware and software I used, the steps I took to port the code to CUDA and conclude how the approach I took resulted in a more efficient program that would not only be faster but scale better with larger image sizes.

## Hardware

As I am not using university lab equipment I thought I would have a quick run through of the hardware on my laptop which is what I will be benchmarking on. My laptop is equipped with a top of the line GPU and CPU. The CPU is a sixth generation core i7 and is still one of the fastest mobile processors out right now. While it is one generation behind the newly released 7th generation intel processors this CPU is no slouch and will make sure any GPU solution gets a run for its money. My GPU is an NVidia 970m while not a workstation card like the ones found in the labs is still a very fast card beating out an NVidia 960 desktop class graphics card. So there will be plenty of power to exploit using CUDA.



## Software

**Insert Paragraph**

* Visual Studio
* CUDA Toolkit 8.0
* GitHub
* Visual Studio Profiling Tools
* NSight NVidia Profiler

## Goals

Insert Paragraph

* Identical Output
* Speed Improvement
* Identical Interface

# CPU Analysis

## Analysis

Expensive code

## Timings

One graph for all image sizes

## Profiling

Show profiling results

# GPU Implementation

## Port Overview

## Optimisations

## Timings

# Project Results

## Testing

## Comparison

## Conclusion