DE HEER Karel S N

Low-Level Programming  Staffordshire University

Ray Tracer Optimization Documentation

Contents

[**Introduction** 2](#_Toc534838740)

[**Optimizations** 2](#_Toc534838741)

[**Use of C++ Standard Library – Chrono** 2](#_Toc534838742)

[**Memory Management** 2](#_Toc534838743)

[**Heap Manipulation** 2](#_Toc534838744)

[**Framework** 2](#_Toc534838745)

[**Ray Tracer Class** 2](#_Toc534838746)

[**Renderer Class** 2](#_Toc534838747)

[**Animation Classes** 2](#_Toc534838748)

[**Threading** 2](#_Toc534838749)

[**File Input & Output** 2](#_Toc534838750)

[**Data Structures** 2](#_Toc534838751)

[**Conclusion** 2](#_Toc534838752)

[**References** 2](#_Toc534838753)

**Ray Tracer Optimization**

# **Introduction**

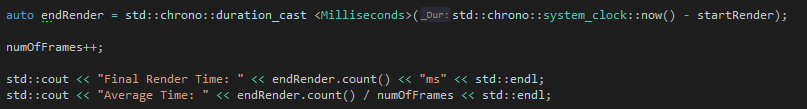
For the assignment, the task was to optimize an inefficient ray tracer framework for not only faster execution speed. In addition to this, the structure of the framework has been changed to increase ease of use and readability for users. This report illustrates the alterations to the code made to reach these objectives and displays tabulated data comparing the speeds of the original and improved ray tracer.

# **Optimizations**

## **Use of C++ Standard Library – Chrono**

To record the speeds between the original ray tracer framework, the optimized and threaded versions, the standard C++ library feature, chrono, has been utilized. This gets the current the system clock at the has been used to record the start and end time of the framework during certain sections of processing. Namely the “SmoothScaling” function in which the program outputs to console the time taken each sphere file to be created/updated and the total time for the whole process with an average time taken as well. The program also outputs to file the total time and average time in milliseconds and seconds.





## **Memory Management**

### **Heap Manipulation**

To better monitor the use of memory within the application, I have created my own memory manager which uses its own new and delete functions which malloc and free memory stored data into random access memory (RAM). A heap class which stores class names and size, and a heap factory which contains all the heaps (default and class specific) created.

## **Framework**

### **Ray Tracer Class**

### **Renderer Class**

### **Animation Classes**

## **Threading**

## **File Input & Output**

## **Data Structures**

# **Conclusion**

# **References**

Json for Modern C++

Nlohmann.github.io. (2019). *JSON for Modern C++: JSON for Modern C++*. [online] Available at: https://nlohmann.github.io/json/ [Accessed 6 Jan. 2019].