Michael Warren

CS 499 – Senior Project

Proposal

1. Scripture

“3 And God said, Let there be light: and there was light.

4 And God saw the light, that it was good: and God divided the light from the darkness.”

Genesis 1:3-4

2. Abstract

The focus of this project is Cluster Computing. The application of this focus is a cluster that produces an image using ray tracing.

3. Background

*Definitions*

Eye-Based: Trace rays starting from the eye.

Source-based: Trace rays starting from the light source.

Beowulf cluster: computers connected by local area network

*Interest*

Graphics are pretty cool and the senior project is a good excuse to play around with ray tracing. Ray tracing is slow but is considered highly parallelizable. The goals parallel processing are easily measurable thus making simpler benchmarks for a senior project. Cluster computing has some real application for my work as well.

*Prior work by others*

* + Lots of the math in ray tracing has been worked out and implemented in libraries.
  + Beowulf clusters are common.
  + MPJ express is a message passing library for java
  + Several libraries for message passing have been written to help with parallel programming.

*Prior work by you*

* + Matrix multiply from operating systems class.
  + Research

4. Description

My senior project will be to create a simple ray tracing engine that can work in parallel or in sequence on one to many computers and one to many processors.

Success in my project is displayed by demonstrating the increase in speed of the ray tracing engine in a parallel environment vs a non-parallel environment.

*Tasks*

1. Preliminary Research and Proposal Preparation
   1. Use Google, Wikipedia, ACM and IEEE digital libraries to learn more about parallel processing and ray tracing.
   2. Using the given template write up the proposal
2. Research
   1. Focus more on papers and other sources that give more details about how to build a parallelized ray tracing engine.
3. Requirements
   1. Using knowledge from research and following the format given in the syllabus put together requirements.
4. Design
   1. Using knowledge from research and following strictly to requirements put together a feasible design.
   2. Techniques such as UML and flow charts will be considered as well as drafts of the UI.
   3. Minimal Effort will be put into UI since project focuses on parallel processing not user interface.
5. Math Unit Tests and code
   1. There is a lot of math involved in ray tracing. To insure that the math is working right unit test will be written to validate that the code is performing the right math.
6. Sequential Ray Tracer
   1. Write the ray tracer to work in sequence.
7. Parallel ray tracer code
   1. Extend the sequential ray tracer to run in parallel, and sequence.
   2. Same math code for sequential run but parallel overhead should not affect sequential run speeds.
8. Cluster Ray Tracer
   1. Extend the ray tracer to run on a cluster over the network.
   2. Insure ray tracer runs long enough to see a difference in running time.

5. Scope

This project is limited to parallel processing on computers on the same local area network and computers with multi core processors. This project is not to show the difference in different parallel programing techniques but to show the performance difference between running a ray tracing engine **in parallel, clustered, and in sequence.**

The ray tracing engine is simple and requires only spheres and boxes to be able to be drawn. Any additional complexity exhibited is to help with increasing the amount of computation required so that a measurable speed difference can be measured. The math behind ray tracing is not part of the project libraries and code with proper documentation of source are allowed for the math portion of the ray tracer.

6. Tasks and Schedule

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task | Start Date | End Date\* | hours to completion | Cumulative hours |
| 1. Preliminary Research and Proposal Preparation | 3/6/2014 | 3/13/2014 | 12 | 12 |
| 2. Research | 3/13/2014 | 3/31/2014 | 30 | 42 |
| 3. Requirements | 3/31/2014 | 4/4/2014 | 6 | 48 |
| 4. Design | 4/4/2014 | 4/11/2014 | 12 | 60 |
| 5. Math Unit Test and code | 4/11/2014 | 4/24/2014 | 18 | 78 |
| 6. Sequential Ray Tracer Tests and code | 4/24/2014 | 5/5/2014 | 18 | 96 |
| 7. Parallel Ray Tracer code\*\* | 5/5/2014 | 5/15/2014 | 18 | 114 |
| 8. Cluster Ray tracer Code\*\* | 5/15/2014 | 6/9/2014 | 42 | 156 |
| \*based on 12 hours of time each week. Also no tasks are planned to end or begin on a Sunday.  \*\* May happen at the same time or separate times. | | | | |

7. Deliverables

* Requirements Specification
* Design Document,
* Source Code

8. Applicability

This project may use aspects from the networking class and some parallel programming techniques learned the operating systems class. These aspects are applied to cluster computing. While some techniques are taught in these two classes cluster computing is not approached in ether course.

9. Required Resources with Costs

|  |  |  |
| --- | --- | --- |
| Item | Cost | Description |
| 2 or more Computers | 0. access to Linux lab is sufficient | To program on and to run the program in parallel on. |

10. References

Paul Rademacher. “Ray Tracing: Graphics for the Masses.” URL: https://www.cs.unc.edu/~rademach/xroads-RT/RTarticle.html

“Ray tracing (graphics).” URL: http://en.wikipedia.org/wiki/Ray\_tracing\_(graphics)

“Message Passing Interface.” http://en.wikipedia.org/wiki/Message\_Passing\_Interface

“Master/Slave (technology).” http://en.wikipedia.org/wiki/Master-slave\_(technology)

“What is a Beowulf.” http://yclept.ucdavis.edu/Beowulf/aboutbeowulf.html

“Introduction to Parallel Computing.” https://computing.llnl.gov/tutorials/parallel\_comp/

“Message Passing Parallel Programming.” https://sites.google.com/site/parallelalgorithmslab/mpi

MPJ express. http://mpj-express.org/

“Vector.java.” http://introcs.cs.princeton.edu/java/33design/Vector.java.html Libary for vector math in java.

“Ray Tracing.” www.cs.utah.edu/~shirley/books/fcg2/rt.pdf

“Implementing the Raytracing Algorithm.” http://www.scratchapixel.com/lessons/3d-basic-lessons/lesson-1-writing-a-simple-raytracer/implementing-the-raytracing-algorithm/