Comparison of Acceleration Structures for Ray Tracing EM Propogation

Eli Ribble

Abstract— **I cover the current state-of-the-art in acceleration structures for Ray Tracing. I propose a method for testing the pros and cons of various structures as they apply to electro-magnetic wave propogation, specifically finding secondary intersections between an eye ray and the scene geometry. My proposal involves using Galileo, an open source ray tracing engine, to test different structures and time them to compare their strengths at finding secondary intersections. A task list, risks and mitigation plans are proposed and deliverables identified for the project.**

Keywords— **Ray Tracing, EM propogation, Thesis Proposal**

I. Introduction

I HAVE recently become involved in a research project that aims to find a way to simulate, in real time, electromagnetic (EM) wave propogation in urban areas. Aspects of this problem have been discussed before in various areas of ray-tracing research, but this will be the first project to attempt to create a viable real-time simulator. While EM waves are often simulated in ray-tracers, the wavelength is generally restricted to the visible spectrum. For this project we will be focused on wavelengths associated with data communication, which behaves very differently. As such we expect to need different acceleration structures than are often used in real-time ray-tracers today. Current acceleration structures are based on suitability to handle very large scene data, ability to spawn secondary and tertiary rays for reflection, refraction and other effects, and parallelization. A better acceleration structure for data communication would be optimized for finding secondary intersections such that distance traveled in a substance is easy to find. This makes it easier to determine signal attenuation through various materials, which plays a major role in our simulation.

My research will compare some well-known ray-tracing acceleration structures against each other for suitability for our task. I will do this by creating a ray-tracer that finds primary and secondary intersections between the cast rays and the scene data. It will time how long it takes to find all of the intersections for each of the acceleration structures. Comparing the times will give a good indication of which acceleration structure is most suited to our task.

II. History

Various acceleration structures have been important to ray-tracing since it's beginnings due to the high cost of calculating the intersection of every ray with every object in the scene. Different methods have been studied to speed up ray-tracing [2]. These methods can be generally broken down into Bounding Volume Hierarchies (BVH) and Space Partitioning (SP).

BVHs have been used for 25 years or more by various researchers [3] [4] [5] [6]. BVHs consist of a hierarchal system of volumes that divide the scene data. A single large volume will contain smaller child volumes, which eventually contain the actual scene data. Searching for the nearest hit involves a search very similar to a binary search tree, and has shown experimentally to have approximately O(R * log N) render time complexity, where R is the number of rays and N the number of objects in the scene. BVHs have also been shown to have superior performance for sparse scenes.

[1] first proposed using a k-D to accelerate associative retrieval. k-D trees are a type of space partition, and many variations exist in the field of ray-tracing [7] [8] [9] [10]. The basic idea is to partition all space such that it belongs to one and only one leaf of the tree. Leaves belong to parent nodes which are generally organized so that nereby locations in space are close together in the tree. As such, finding nearby volumes by navigating the tree is easy. Different variations of the Kd tree may partition space on axis-aligned planes where the planes bisect one primitive in the scene [11].

Due to the proliferation and power of these acceleration structures, real-time ray tracing has become possible using commodity hardware [12]. Furthermore, photon mapping shows that it is possible to simulate many different properties of EM waves in ray-tracing, including radiosity solutions, as a pre-processing step [13]. This indicates that a real-time ray-tracing solution to simulating EM propogation at communication frequencies is possible, it is only a matter of finding the best technique of the many to choose from.

III. Proposal

I propose to use the open-source ray-tracing environment Galileo to measure the suitability of various ray-tracing acceleration structures for finding secondary intersections with scene geometry. Galileo already supports some acceleration structures and has several scenes that have been checked in with the code. I will create a module for Galileo that will find the secondary intersection for all primary rays and time how long it takes at different resolutions and for several of the built-in scenes. I define secondary intersections to be the intersection of a primary ray after the first intersection with the ray continuing on in the same direction. Secondary intersections should almost always occur as the ray leaves the object it first intersects. In order to

Thanks to Lee Butler, Pete Shirley and Ken Stevens for making this happen.

understand which structure is best over a range of different types of scenes I will test each structure against scenes of differing complexity, from a few thousand primitives to hundreds of thousands.

A. Project Tasks

Table I shows the projected task list and due dates.

It is important to get Galileo running on two different computers for several reasons. First, it will allow me to check the robustness of any code I write. The two computers will use different CPU architectures and hardware configurations which will force my code to be portable. Second, it will allow me to test the various acceleration structures on different systems to ensure that any speedup is not an artifact of a particular system. Third it will force me to intelligently update the source code repository with any changes so that I can move code from one machine to another. This step should be relatively simple as I've already gotten Galileo compiled and running on one machine.

Task 2 is important because I will need to familiarize myself with the renderer's code in order to add modules to it. The project's webpage currently has no documentation, and the SVN shows very little documentation as well. By writing documentation for the project I will need to learn how the code works and create a useful byproduct for others. To facilitate writing the documentation and communication with those who have already worked on the code I will set up a wiki where information can be posted and discussed before adding it to the documentation for the project. This wiki will be set by January 1, 2008. As the information on the wiki crystallizes it will be added to the project page documentation. This phase will be completed by January 20, 2008. Before the phase finishes I will begin work on tasks 3 and 4.

Task 3 goes hand-in-hand with task 2. As I document the code for Galileo I will also document the sample scenes, which are included with the code in the SVN. I will choose 5 of the samples scenes to use for my tests. Tests will be chosen to show large variance in number of primitives and complexity of the scene.

I will begin designing the code for task 4 while learning about the code during task 2. It is expected that most of the design will be completed by the time task 3 is finished, which leaves all of February for coding, optimization, and debugging. The wiki will be an invaluable asset in designing the secondary intersection module as it will allow me to collaborate with experienced Galileo developers to create a module that works well with the system as a whole.

The first module I will create is a timing module that shows the time spent in each aspect of a scene render, such as pre-processing, ray generation, primary ray intersection, and secondary ray intersection testing. I'll make this module first as it should be the easier of the two.

The second module I will create will calculate the time it takes to determine the second intersection of a ray with the scene data. This module may be quite complex as the best method to find the intersection will depend on the acceleration structure being used. The module will not produce images, but rather produce nothing but timing data.

Task 5 should take very little time as I will be using the test scenes to debug my secondary intersection module. I have alloted 10 days to taking accurate measurements on the two computer systems I will be using, and troubleshooting any difficulties I have with the scenes.

If I'm able to get far ahead of this schedule, I will implement an acceleration structure of my choice and add it to Galileo's current acceleration structures. This will be an important step in showing that I understand why a given acceleration structure is superior to another for secondary ray intersection tests. By coding another acceleration structure of my choice I show that I can choose from the myriad structures that have been created to pick one that improves upon the structures already available. This step also has important implications for the EM transmission simulator as a whole because the structure that I code will likely become the acceleration structure of choice for the simulator.

The final task will be to write the thesis paper based on my results. It will be written in LaTeX and should show the course of my research during 2007-2008. I will also produce presentation materials so that I can present at the Technical Open House on March 27th. I plan to finish the writeup by March 20th so that I can have a week to practice for my presentation, as well as to give me a week of breathing room should something go terribly wrong.

B. Schedule Flow

Although I will be the primary person to work on this project, I will not be working alone. I will be working with Lee Butler, who is in charge of teh EM transmission simulator as a whole, as well as Pete Shirley, my advisor. Furthermore, Lee will be bringing in consultants from time to time who I may need to work with. Because I am on the payroll of the Army, I may be required to change my schedule to work on other aspects of the simulator project. I am expecting to spend 10 hours a week on this project as a whole, but because of the above those 10 hours may or may not be spent on the tasks I've outlined above. As such, the schedule outlined above is very flexible. The only non-flexible date is March 27th.

C. Components to be Purchased/Built

Galileo is open-source, and as such is free to download, modify and re-release. I already own the computer systems on which I plan to develop and test the software. Therefore there is no equipment to purchase.

There will be a few software components to build. The largest piece will be the secondary intersection and timing code. If there is time I will also build a new acceleration structure. This code will be re-released by me under the same MIT license, and as such will be integrated into the Galileo SVN.

TABLE I
PROJECT TASK LIST

Task	Due Date
1. Download and compile Galileo on two different computers	January 5, 2008
2. Update sourceforge page to describe the contents of the SVN	January 20, 2008
3. Determine primitives in sample scenes. Choose 5 scenes to be used	January 25, 3008
4. Create a module that finds secondary intersections and render time details	March 1, 2008
5. Test the existing acceleration structures against the sample scenes	March 10, 2008
6. Write up the results of the tests	March 20, 2008

D. Interface Issues

Because I am interfacing with existing software, it will be important to ensure that the software I build interfaces correctly with Galileo. The new secondary intersection system will be enabled via a commandline switch. This switch will disable image output (as secondary intersections do not contribute anything to an image, and a final image is not necessary to time the secondary intersections) and enable the timing system. The user will be warned when the secondary intersection system is engaged to prevent confusion as to why no image was produced.

The timing system will also be available as a stand-alone commandline switch. It will also disable image output. The output from the program will then be several lines of timing information that note how many milliseconds were involved in the scene pre-processing, how many in the primary intersection and how many in the secondary intersection (if enabled) as well as any other major processing steps. This data will be instrumental in determining where the timing strengths and weaknesses of different acceleration structures lie.

The final major interface issue has to do with the creation of documentation for Galileo. The wiki I set up will use MediaWiki, which includes powerful collaboration tools for improving the accuracy of the information before it is included in the source. Because wikis can be modified by anyone, information will naturally tend towards accuracy over time, provided enough people have access to the wiki. Data accuracy will be verified against the source code and by developers on the project. This will help to ensure that the data is accurate, consistent and useful to the final end user.

E. Testing and Integration

Testing is very important as I will be producing source code to incoporate in an existing system, and it is essential to avoid damaging the functionality of the system when adding new features. The project is already managed by an SVN, which will allow us to rollback code should something catestrophic happen. As updates are made I will test the new code on two different computer systems to ensure that it is portable. Updates to the code will be small to ensure that bugs are caught early and that rollbacks in the SVN are less painful.

Each of the two modules I will produce will be tested in the same fashion. A design document will be produced on the wiki that will discuss the planned features, which files are to be modified and how the functionality of the system should be effected. Before any coding occurs a testing document will also be created on the wiki that will specify all of the tests to be performed on the new functionality. Testing will be black-box and only test the externally visible portions of the module. When these documents are approved by two of the wiki users, development may begin. Branching the SVN won't be necessary for these modules, but commits should happen at least every day during development. When development is complete the tests will be performed on the module and the results for each version in the SVN will be posted with the test document on the wiki. For any failed tests a note will be made at which version the bug occurred and at which version it was resolved. Further development will not occur until all tests pass.

I do not anticipate that the tests will become part of the source code, as its beyond the scope of this project to create a entire testing framework for Galileo. This is why black-box testing is the best strategy for this project - it will allow us to test new features by manipulating the program rather than creating an internal testing framework.

Because modifications to Galileo will occur in the trunk of the SVN, the new code will be integrated into the system as development progresses. The only major integration issue will be adding the new documentation to the SVN. The documentation will be added to a new ./doc directory which will point users to the wiki for indepth information. Documentation will be plaintext and broken into various files such as README, INSTALLATION, FAQ, and RELEASE_NOTES. The commandline documentation will also be updated to reflect the new commandline options.

F. Risk Assessment

The greatest risk to the project at this point is whether or not the work can be completed by March 27 so that the results can be presented at the Technical Open House. Because of funding changes I have had to start this project with 4 months time total to do the project and a limit of 10 hours a week paid work. While I can work more hours and not get paid, I will be a full-time student while doing this project and so will have other responsibilities in other classes.

To deal with this risk I plan work unpaid hours if necessary to complete enough of the project to give a useful presentation. If that still isn't enough time to get enough work done, I will use some of the results of Vlastimil Havran's thesis work to discuss some of the expected results of my research [14]. Doing that work will require some time, so I will make the decision on whether or not I will have coherent results 2 weeks before the March 27 deadline (March 13). That will be enough time to have become familiar with what acceleration structures Galileo supports so that I can make a projected analysis on their suitability for secondary intersection tests. I'll then create a presentation around that analysis, in spite of the fact that I don't have experimental results.

The next risk to the project comes in the limitations of Galileo itself. I do not have time to make large modifications to the system, and there currently is no documentation to explain what the limitations are. Over the next month it is possible that I find that Galileo doesn't have the capabilities to support secondary intersection testing, accurate timing or that the code is too complex to modify in the time allotted.

The mitigation plan for this risk involves switching to another ray tracing engine. I have access to the source code for Manta, which I have worked with in the past and provides another way of making the same tests. I have also created a ray tracer for a previous class which I could use, though it would require a great deal more coding than using an existing engine, and be a great deal less useful to the EM propogation simulation. It is, however, a system I am much more familiar with and would be much easier to modify to add the capabilities I need.

The final risk comes from my funding source. As I've found in the past several moths, it's very possible for research funding to dry up at inconvenient times, or for funding stipulations to change. It is unlikely, but possible, that the Army pulls me from this project for one reason or another, and I no longer have access to the Galileo developers and the other researchers on this project.

The mitigation plan for this risk is relatively simple. I will work by myself, without pay, the same way many other students have done. I expect that the people I have come to know on this project will still be willing to help me, even if I don't have official status/funding, and so I should still be able to get the work done.

G. Deliverables

Over the course of this project I will produce experimental results that show the time taken to determine the secondary intersections for all of the primary rays for 5 different scenes. These results will be reproduced 10 times and averaged. The data will include information on preprocessing, building the data structure, creating the rays and finding the intersections. It will include data from three different data structures, including a kD tree and a BVH. This data will show which type of acceleration structure is most suited to finding secondary intersections, which will indicate how to best create an interactive ray-tracing solution to simulating EM propogation in urban landscapes.

Aside from the results I will also produce code that will be released under the MIT license and integrated into Galileo, the rendering system I will use to find the results. This code will include modules that will be activated via a commandline switch. One module will disable image output and show timing data for the render specified at the commandline. The other module will disable image output and calculate the secondary intersection for the primary rays. Both modules will interact with the current Galileo commands that specify scene data, image resolution and other aspects of the final render.

As part of the module code I will also produce documentation explaining to a user how to use the commandline switches. I will create documentation for developers explaining what was added to Galileo and how it effects the program. I will create a wiki that will document Galileo, how it works and how I designed and implemented the changes to it. This wiki will also document discussions between developers and how decisions were made.

Finally I will produce presentation materials and a final thesis document. The presentation materials will be used at the Technical Open House at the University of Utah on March 27th as part of my thesis course. The final thesis document will have similar content and show the results of my project.

IV. Summary

Simulating electromagnetic wave propogation at interactive rates poses few problems in light of the amount of research that has already been done. The major question that determines whether or not such a renderer is feasible relies on knowing which acceleration structure is best suited to handling EM at communication frequencies. Secondary ray intersections are a good way of assessing the strengths of different acceleration structures. By modifying Galileo, an open source ray tracer, we can time how long it takes to find the secondary intersection for each of Galileo's sample scenes. We can then time a secondary intersection render using each of the acceleration structures supported by Galileo, including space partitions and bounding volume hierarchies. This will give a strong indication about which of these structures will yield the best results in a realtime EM propogation simulator.

This project will produce the results of the above tests, as well as modules so that others can repeat the tests using any scene data and Galileo. I will also create documentation and a wiki to improve Galileo's usability for future developers. The final results of this project will be presented at the Technical Open House on March 27th and the code will be released to the public. The results will also be used by the Army to create a simulator that can show in realtime how to best position antennas to facilitate communication in an urban area.

V. Acknowledgements

I'd like to thank my wife for giving me up for a while to get lost in my projects. I'd like to thank Lee Butler for the idea for this project, Pete Shirley for his overall guidance and for finding me funding to do what I enjoy and Ken Stevens for constantly pushing me to get things done.

References

[1] Jon Louis Bentley "Multidimensional Binary Search Trees Used for Associative Searching," *Communications of the ACM 18*, 9, pp.509-517, Sept. 1975

[2] A. S. Glassner, ed., *An Introduction to Ray Tracing*, Academic Press, 1989

[3] T. Whitted, "An Improved Illumination Model for Shaded Display," *Communications of the ACM 23*, 6, pp. 343-349, June 1980

[4] S.M. Rubin and T. Whitted, "A 3-Dimensional Representation for Fast REndering of Complex Scenes," *Computer Graphics 14*, 3, pp. 110-116, July 1980

[5] H. Weghorst, G. Hooper, and D.P. Greenberg, "Improved Computational Methods for Ray Tracing," *ACM Trans. on Graphics 3*, 1, pp. 52-69, Jan 1984

[6] T.L. Kay and J.T. Kajiya, "Ray Tracing Complex Scenes," *Computer Graphics 20*, 4, pp. 269-278, August 1986

[7] J.G. Cleary, B. Wyvill, G.M. Birtwistle, and R. Vatti, "Multiprocessor Ray Tracing," *Research Report No. 83/128/7 Dept. of Computer Science University of Calgary* (1983)

[8] A. S. Glassner, "Space Subdivision for Fast Ray Tracing," *IEEE Computer Graphics and Applications 4*, 10, pp. 15-22, Oct. 1986

[9] A. Fujimoto and K. Iwata, "Accelerated Ray Tracing," *Proc. CG Tokyo '85*, pp. 41-65.

[10] A. Fujimoto, T. Tanaka, and K. Iwata, "ARTS: Accelerated Ray-Tracing System," *IEEE Computer Graphics and Applications 6*, 4, pp. 16-26, April 1986

[11] Maxim Shevtsov, Alexei Soupikov, Alexander Kapustin, "Highly Parallel Fast KD-tree Construction for Interactive Ray Tracing of Dynamic Scenes," *Computer Graphics Forum 26*, 3, pp. 395-404, Sept. 2007

[12] Ingo Wald, Solomon Boulos, Peter Shirley, "Ray Tracing Deformable Scenes Using Dynamic Bounding Volume Hierarchies," *ACM Transactions on Graphics 26*, 1, Article 6, Jan 2007

[13] Henrik Wann Jensen, "Global Illumination using Photon Maps" *Proceedings of the Seventh Eurographics Workshop on Rendering*, pp. 21-30, 1996

[14] Vlastimil Havran, "Heuristic Ray Shooting Algorithms" *Czech Technical University*, Ph.D Thesis