Kirbie Dramdahl

CSCI4901: Senior Seminar

Summary of Previous Senior Seminar Paper

21 January 2013

For this assignment, I will be summarizing a paper written for the twenty-seventh computer science discipline senior seminar conference, which took place during the fall semester of 2011 on 3 December. The paper is written by Will W. Martin and is titled *Implementation of Kd-Trees on the GPU to Achieve Real Time Graphics Processing*.

This paper begins by describing the usage of ray tracing in generating computer graphics. This technique renders high-quality, realistic graphics, but also requires a significant amount of time (sometimes hours) to render a single frame, obviously making it impractical for on-line, real time applications. From here, the paper then describes how a type of data structure referred to as kd-trees can be used to organize pixel coordinates into axis-aligned bounding boxes that may be efficiently searched. The paper then demonstrates that, if this system is used in combination with running the ray tracing algorithm on the graphics processing unit (GPU) rather than the central processing unit (CPU), the ray tracing technique may be used to efficiency in on-line, real time applications.

The paper seems well-organized. The first section is logically enough titled “INTRODUCTION” and details what ray tracing is, how it works, and its advantages and disadvantages, as well as providing a brief summary of the rest of the paper. More specifically, the first paragraph describes what ray tracing is and what it is used for and the advantages of using it, the second paragraph provides a brief description of how ray tracing works, and the third paragraph describes the disadvantages of ray tracing alongside an explanation of how these disadvantages can be counteracted using kd-trees and the GPU. Finally, the third paragraph explains that kd-trees will be discussed in detail in the second section and the GPU in the third section. The second section is titled “THE KD-TREE DATA STRUCTURE” and is broken up into two subsections, the second of which is itself broken up into three smaller subsections. The first paragraph briefly describes some of the usages of kd-trees within graphics processing. After this, the first subsection begins, titled “What is a Kd-Tree?” The first paragraph of this section describes kd-trees and their structure, specifically under the context of ray tracing. The second paragraph briefly describes how the structure of the kd-tree may be used to divide the grid using split planes, which proves useful in determining where rays will cross the scene. Here the second subsection begins, titled “Kd-Tree Tools.” The first paragraph essentially summarizes the content of the following subsection of the second subsection. The first subsection of the second subsection is titled “Axis-aligned bounding box.” The first paragraph explains what axis-aligned bounding boxes are and how they can be used to reduce the area that must be checked, and the second paragraph describes how this tool can be applied specifically to ray tracing. The second subsection of the second subsection is titled “Surface Area Heuristic.” The first paragraph explains the usefulness of the Surface Area Heuristic (SAH) in distributing split planes more evenly and thereby improving efficiency, the second paragraph describes how the SAH assigns cost to kd-trees and thereby provides a measure by which to test their efficiency, and the next several paragraphs describe the process by which the equation for the SAH is obtained. The third subsection of the second subsection is titled “Empty space minimizing and split median.” The first paragraph describes empty space minimizing, and the second paragraph describes median split. Both are methods by which empty space is eliminated and the efficiency therefore improved. The third section is titled “GPU VS. CPU” and is broken up into three subsections. The first paragraph briefly compares the CPU and GPU. After this, the first subsection begins, titled “Benefits Provided by The Threading Abilities of the GPU.” This section contains one paragraph, which describes exactly what the subsection title says it will. Here the second subsection begins, titled “Running Kd-Tree construction on the GPU.” This section contains two paragraphs, the first of which describes in detail how the GPU constructs kd-trees, and the second of which describes how the SAH and split median work to create cubic areas, as these prove most efficient because they have the smallest surface area. Finally, the third subsection begins, titled “Results.” This section contains three paragraphs. The first paragraph compares the results of running the kd-tree construction and traversal on the CPU versus the GPU, the second paragraph describes the scaling capacity of the GPU algorithm, and the third paragraph compares the results of the GPU algorithm to two other algorithms. The fourth and final section is titled “CONCLUSIONS” and contains one paragraph once again summarizing the content of the paper, concluding that with the use of kd-trees and the GPU, ray tracing may be efficiently used for on-line, real time applications.

This paper references fourteen sources. The first source (R. Austinat, GPU ray tracing: New pictures reveal current possibilities) is referenced solely for the purpose of using an image (Figure 2). The second source (F. S. Hill Jr., *Computer Graphics Using Open* GL) again appears to be used solely as the basis for an illustration (Figure 1). The third source (J. Goldsmith and J. Salmon, Automatic creation of object hierarchies for ray tracing) is once again referenced as a basis for illustrations (Figures 5 and 6). The fourth source (B. Grass, X3D class examples of ray tracing) is also referenced for use of an image (Figure 2). The fifth source (W. Jarosz, H. W. Jensen, and C. Donner, Advanced global illumination using photon mapping) is referenced in the first paragraph of the second section in reference to applying kd-trees to photon mapping). The sixth source (J. D. MacDonald and K. S. Booth, Heuristics for ray tracing using space subdivision) provides an image (Figure 7) and much of the information on the Surface Area Heuristic, perhaps most importantly one of the equations used in its calculation. The seventh source (S. Popov, J. Gunther, H. P. Seidel, and P. Slusallek, Experiences with streaming construction of SAH kd-trees) appears to be used solely in reference to the application of kd-trees to ray tracing. The eighth source (A. Reshetov, A. Soupikov, and J. Hurley, Multi-level ray tracing algorithm) seems to serve the same purpose as the seventh. The ninth source (M. Shevtsov, A. Soupikov, and A. Kapustin, Highly parallel fast kd-tree construction for interactive ray tracing of dynamic scenes) was used in a manner similar to the seventh and eighth, but also was the source of one of the algorithms against which the GPU algorithm was tested. The tenth source (L. J. Shiue, I. Jones, and J. Peters, A realtime GPU subdivision kernel) appears to be referenced for is information on kd-tree tools. The eleventh source (I. Wald, S. Boulos, and P. Shirley, Ray tracing deformable scenes using dynamic bounding volume hierarchies) is the source of one of the algorithms against which the GPU algorithm was tested. The twelfth source (I. Wald and V. Havran, On building fast kd-trees for ray tracing, and on doing that in O(nlog(n))) seems to be used in a manner similar to the seventh, and also provides the same equation for calculating the SAH provided earlier by the sixth. The thirteen source (I. Wald, J. T. Purcell, J. Schmittler, C. Benthin, and P. Slusallek, Realtime ray tracing and its use for interactive global illumination) is used in the same manner as the seventh. The fourteenth source (K. Zhou, Q. Hou, R. Wang, and B. Guo, Real-time kd-tree construction on graphics hardware) appears to be the main source, as it is the source of several images (Figures 10 and 11 and Table 4) and the vast majority of the information on kd-trees, including the primary equation used in computing the SAH.

Sources five, seven, eight, nine, ten, eleven, twelve, thirteen, and fourteen are all peer-reviewed and (relatively) recent – relatively here meaning within the last ten years, which in terms of computer science is pushing it. Sources three and six are peer-reviewed, but certainly not recent. However, as both of these sources are used either for images or mathematics, the shelf life is perhaps a bit longer. This leaves sources one, two, and four. Sources one and four are very recent relative to the paper, but not peer-reviewed. Source two appears to be a book reference, but no date is provided. However, again, all three of these sources are referenced for images rather than information, which allows them some leniency. As nine of the fourteen sources used are both recent and peer-reviewed, and the main source is among these, the paper seems to satisfy the requirement that the key sources be recent and peer-reviewed research papers. However, some of the sources appear to be used solely as “stuffing,” that is they provide no new information, but merely restate what other sources have already contributed. This could be seen as a method of verifying the authenticity of information, but it comes off as somewhat excessive and unnecessary.

The author uses several techniques to convey the material to their audience, including image examples, diagrams, tree representations, graphs, and tables. Particularly helpful are the tree and graphical representations of kd-trees and their method of data organization (Figures 3, 4, 8, and 9), and the series of mathematical equations presented in 2.2.2 demonstrating how to obtain the SAH.

For the most part, yes, this is a well-written paper, and easy to understand. I feel that some clarification could be used in 2.2.3 and 3.2, but that also may just be me being dumb, which happens more often than I would care to admit.

As stated previously, I really enjoyed the mathematical walkthrough in 2.2.2, and Figures 3, 4, 8, and 9 were really effective in helping to clarify kd-trees. Tables are also always good and fun and effective. The organization also seems very logical. However, again, as mentioned in the previous paragraphs, there were a couple of sections that I got a bit lost in, and these I feel could be clarified or explained differently. Also, the reference stuffing sort of bothers me. Overall, however, this was an interesting, engaging, and informative paper.

I do have some questions. I actually read several of the papers from the senior seminar conference of fall semester 2011, and what struck me was how un-computer science-esque some of these seemed. This applies especially to the first two papers. Obviously, they contained elements of computer science, but it didn’t feel like the focus. And yet, their inclusion in the booklet would indicate they passed inspection. I guess my question here is, what exactly entails a passable (preferably better) computer science senior seminar paper? I suppose I was expecting something more technical.