Analysis of two common hidden surface removal algorithms
Painter’s algorithm and Z-buffering
Daniel Nyberg
08/02/2011
1 Introduction

In today’s world, computer generated 3D is abundant. It is in your movies, your video games, your phone, your operating system (Windows Vista) on your computer. It is especially with the growth of powerful processors and graphics processors that an average homeowner has come to possess an extremely powerful piece of equipment, whose use is almost exclusively for playing video games. With this, the video game market has exploded, and has gone from a niche market in the late 70’s into the global juggernaut of an industry it is today. With 3D computer games comes the problem of rendering an entire virtual world, sometimes seemingly infinite, without slowing down due to the graphics processor’s limitations. This is where hidden surfaces enter the picture.

Hidden surfaces are abound in 3D computer graphics today. A hidden surface is where a surface is not or cannot be viewed by the virtual camera e.g. the player in a video game. This renders the hidden surface useless, and thus is an unnecessary strain on the graphics processor. Hidden surfaces are especially numerous in video games and computer generated 3D effects, where the game cannot afford to put a heavy load on the processor. The same scene seen from the virtual camera can have vastly larger amount of polygons in a graphical shader that doesn’t utilize hidden surface detection algorithms than a shader that does.

This area is especially interesting to me, as I have grown up with video games, and aspire to one day work for a video game company. Thus, 3D graphics and techniques pertaining to their usage are of special interest to me personally. I hope that with this thesis that I will gain more knowledge and insight into the world of video games and 3D graphics rendering in general.

2 Problem statement

To determine if a surface is hidden, there are several hidden surface detection algorithms employed in the field of 3D computer graphics today. This report will aim to show two of the most common algorithms and compare them to each other: Z-buffering and Painter’s Algorithms.

3 Approach

I intend to implement simple versions of the above mentioned hidden surface detection algorithms; after which I will compare. This comparison will be done by implementing the two algorithms and having them detect hidden surfaces on the same 3D image. Using a code profiler, the time taken to render the image, the amount of polygons removed from the hidden image, and the memory usage of the application is measured. This will give measurable data to help in comparing the two algorithms.

I also intend to interview an experienced member of the gaming industry to gain some insight into the methods used in the industry to avoid hidden surfaces.
4 References

- De Berg, Mark 1993 *Generalized Hidden Surface Removal*
- Erickson, Jeff 2000 *Finite-Resolution Hidden Surface Removal*
- Sharir, Micha 1992 *A Simple Output-Sensitive Hidden Surface Removal*
- Sutherland, Ivan E. & Sproull, Robert F. & Schumacker, Robert A. 1974 *A Characterization of Ten Hidden-Surface Algorithms*
- Taxén, Gustav – Ph.D. Human-Computer Interaction at the Royal Institute of Technology; Has worked for Avalanche Studios.

5 Time plan

- 11th February – Start reading through for sources.
- 14th February – Start writing introduction of thesis. Start implementing one of the algorithms.
- 15-25th February – Try to get an interview and analyze it. Try to finish the first algorithm.
- 9th March – Halfway meeting
- 10th March – Reevaluate and make a new time plan