Leopold-Franzens-Universität Innsbruck

Institute of Computer Science
Interactive Graphics and Simulation Group

Raytracing Project

Advanced Computer Graphics
Documentation

Phillip Mildenberger
Stefan Spiss
Cem Okulmus

advised by
Savoye Yann Pierre, PhD

Innsbruck, November 19, 2014

1 Introduction

For this project we had to implement certain advanced features based on smallPT. Every student in one group had to implement at least one main feature. In our group we chose Motion Blur (by Cem Okulmus), Depth of Field (by Stefan Spiss) and Specular/Texture/Normal Mapping (by Philipp Mildenberger). Additionally in our project extra features were implemented, such as: Acceleration Structures (by Philipp Mildenberger), simple Camera Control (by Stefan Spiss).

2 Technical Overview

SmallPT was extended by .obj import and complex geometry now supporting triangle meshes. Additionally a bounding volume hierarchy (BVH) is used to speed up the calculation of complex scenes. Texture-/normal- and specular-mapping is implemented to generate more realistic images. Furthermore to simulate a camera, Depth of Field provides correct blurring of objects that are not in focus. It's now also easy possible to move the camera around, change it's focal distance, it's field of view and it's aperture size. To capture movements in still pictures the effect of Motion Blur is approached by simulating the aperture shutter speed.

3 Implementation Detail

In this section we will give a short explanation how every feature was implemented. All of these are extensions of the smallPT code.

3.1 Motion Blur

If we want to simulate motion using multiple ray-traced images, we will observe a strobing effect, since a camera always blurs a moving object. This is called Motion Blur. As was explained during the lecture, the basic idea behind this effect is that we want to distribute the rays not in space but in time. For this we usually also want to have some sort of animation, otherwise the effect wouldn't show. This method was first detailed by Cook et al. [3]. Since smallPT also uses a distribution over space, only an extension of this was needed. The rays are uniformally distributed over time, with equal weight for all time steps. So the four subpixel samples are assigned different time scenes and the average is returned. This simulates an object that's moving with no acceleration.

3.2 Depth of Field

In standard smallPT the pinhole camera model is used. So everything in the scene is in focus. Normal cameras always have a Depth of Field, which means that only objects in a specific distance are in focus. Everything else is blurred depending on the distance to the focal point. In the project implementation the Thin-Lens model is used. For this the ray direction is changed. Rays are shot not only from one camera position, but distributed over the aperture circle. As target for the rays, the focal plane is used instead of the image plane. See Figure 1

Furthermore for general information about Depth of Field following sources were used:

- A realistic camera model for computer graphics, [5]

- Camera models and optical systems used in computer graphics: part I, object-based techniques, [1]
- Algorithms for rendering depth of field effects in computer graphics, [2]

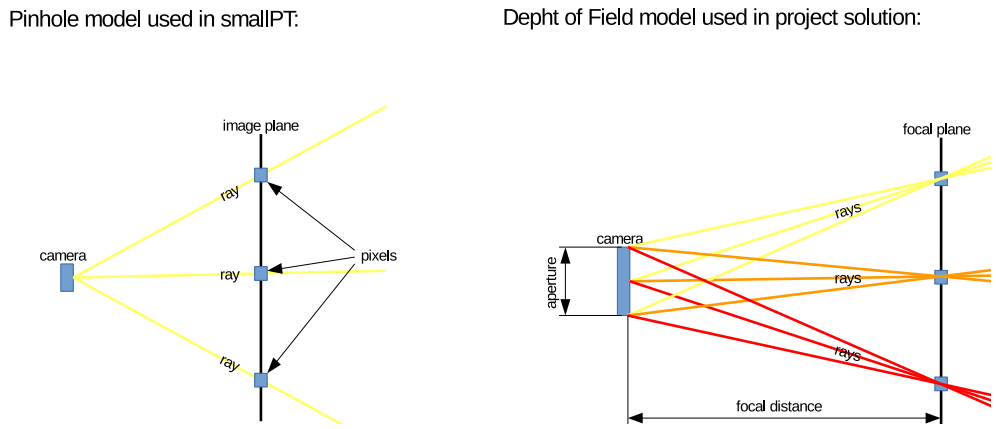

Figure 1: This figure shows the simplified camera models. On the left the model used in smallPT, on the right the model for the project solution can be seen. (**Recreation from:** online tutorial from Steve Harvey, [4])

3.3 Specular/Texture/Normal Mapping

UV mapping from obj-files is supported, including import of color, specular and normal data. For specular mapping a new material had to be created: Glossy surface. It's used to supply specular data with a texture. The normal mapping is changing the intersection normal for depending on the texture for the surface. For every intersection we return a special intersection structure, additionally containing the needed data for specular/normal/texture mapping.

3.4 Acceleration Structures

To accelerate the intersection of complex triangle-meshes in smallPT, we implemented different acceleration structures. The first is just simple approach: A bounding box is put around every object, and the intersection function first checks every bounding box and only continues if one was hit. The second is a Bounding Volume Hierarchy (BVH). An octree is used to build up the hierarchy recursively: Every object is inserted using its centroid. From these objects, using bottom-up traversal, the BVH is constructed.

For the implementation of Acceleration Structures mainly the website scratchapixel [6] was used.

3.5 Camera Control

In smallPT the camera settings are hard-coded (fixed position, direction, field of view). The project solution uses a additional camera class which implements all camera aspects like position, direction, field of view and focal distance. Now it is very simple to render a scene from different perspectives with different camera settings.

4 Results

The following images are rendered with our program showing different features of it.

- Motion Blur:

Figure 2: Picture rendered just with Motion Blur

- Depth of Field and changed camera settings:

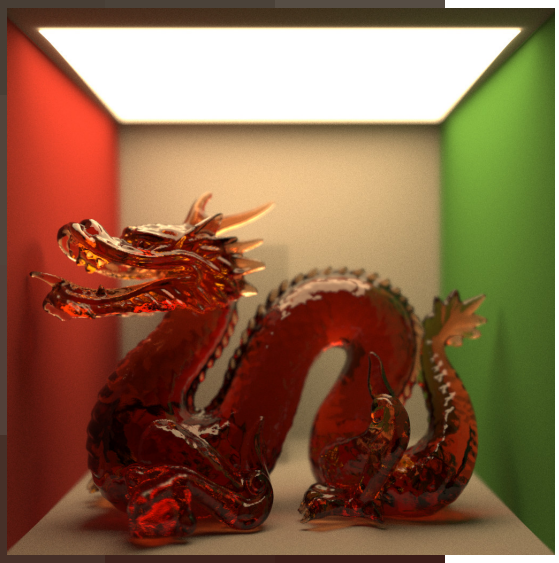

Figure 3: Picture rendered just with Depth of Field

Figure 4: Picture rendered with changed camera settings

- Specular/Texture/Normal Mapping:

References

[1] Brian A Barsky, Daniel R Horn, Stanley A Klein, Jeffrey A Pang, and Meng Yu. Camera models and optical systems used in computer graphics: part i, object-based techniques. In *Computational Science and Its Applications—ICCSA 2003*, pages 246–255. Springer, 2003.

[2] Brian A Barsky and Todd J Kosloff. Algorithms for rendering depth of field effects in computer graphics. In *Proceedings of the 12th WSEAS international conference on Computers*, volume 2008. World Scientific and Engineering Academy and Society (WSEAS), 2008.

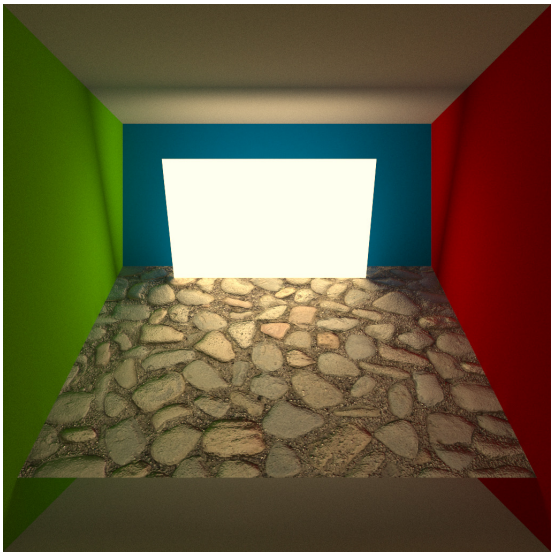

Figure 5: Picture rendered with Specular/Texture/Normal Mapping

[3] Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. *SIGGRAPH Comput. Graph.*, 18(3):137–145, January 1984.

[4] Steve Harvey. Ray tracer part six – depth of field. `http://steveharveynz.wordpress.com/2012/12/21/ray-tracer-part-5-depth-of-field/`, 2012. [Online; accessed 2014-11-19].

[5] Craig Kolb, Don Mitchell, and Pat Hanrahan. A realistic camera model for computer graphics. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 317–324. ACM, 1995.

[6] Scratchapixel. Lesson 12: Introduction to acceleration structures. `http://www.scratchapixel.com/old/lessons/3d-basic-lessons/lesson-12-introduction-to-acceleration-structures/`, 2012. [Online; accessed 2014-11-19].