Perceptually Accurate Depth of Field Rendering

Robert Konrad
email rkkonrad@stanford.edu

April 29, 2015

1 Introduction

Focus cues, accommodation and retinal-image blur, are challenging to achieve with wearable virtual reality displays today. An accurate reproduction of these cues are critical in providing a visually immersive experience. Yet, many consumer displays on the market do not support either of these cues. In this project, I plan on exploring fast, efficient algorithms for rendering a retinal-image blur (depth of field) on scenes displayed in virtual reality experiences.

2 Depth of Field

Computer graphics methods for creating depth of field effects can be categorized into two bins[1]. The first, object space methods, operate on the 3D scene representation, and build depth of field effects directly into the rendering pipeline. The other, image space methods operate on images that were rendered with everything in perfect focus. With the aid of a depth map, the images can then be blurred.

In general, object space methods generate more realistic results but suffer from large computational overhead. Image space methods, trade off some of the accuracy for computational efficiency. Because I am focused on virtual reality applications, latency is a critical factor. Therefore I will focus on image space methods in hope of achieving accurate depth of field rendering in real time.

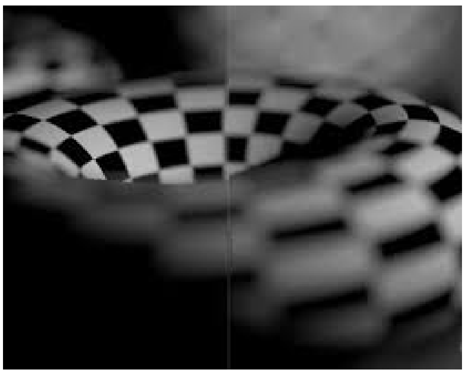

Figure 1: Simulated depth of field.

3 Approach

I plan on evaluating different algorithms based on their distance from a baseline image and their performance. Throughout the experiments, I will be assuming a thin lens model of the human eye. The baseline image will be generated using geometric optics, which Cook et.al. have shown can be directly simulated by distributed ray tracing [2]. This method will render a scene from many points (more than 100) over the area of the human pupil, and then average the images to create the depth of field effect.

I will then go on to investigate various image space methods on generating depth of field effects. Because the images displayed will be computer generated, the methods will have access to a perfect depth map of the scene. All of the following algorithms first be implemented in Matlab to compare their accuracy to the baseline image. Their performance will be measured as well. The most accurate algorithm meeting performance requirements (60fps) will then be implemented on the GPU using OpenGL and demoed at the demo session on an Oculus Rift.

3.1 Linear Filtering

The first, obvious method, will be to use Gaussian filtering to add depth of field to the image. I will use a spatially variant linear filter that has a depth-dependent point spread function similar to the method implemented in Potmesil et.al.[3]. The filter will operate directly in the spatial domain and will therefore have a cost proportional to the area of the PSFs, but because the Gaussian filter is separable, computation cost will be low.

3.2 Non-Linear Filtering

The second approach will involve an adaptive bilateral filter where the weights are based on the depth values of neighboring pixels, as proposed in Wu et.al.[4]. In a real image, a blurred background will never blur on top of an in-focus foreground. Because a bilateral filter preserves edges, it should reduce intensity leakage and give better results than the Gaussian filtering approach, at the cost of added computation.

3.3 Approach Based on Simulated Heat Diffusion

A third approach is based on the the natural physical process of heat diffusion. When a thermally conductive object has an uneven distribution of heat, the distribution will progressively come more even, or blurred, over time. The differential equations of heat diffusion provide an alternative mathematical framework from which we can derive blurring algorithms, as shown in Kass et. al.[5]. I expect this method to give promising results, but with a significant hit to performance.

References

[1] Brian a. Barsky and Todd J. Kosloff. Algorithms for Rendering Depth of Field Effects in Computer Graphics. *World Scientific and Engineering Academy and Society (WSEAS)*, pages 999–1010, 2008.

[2] Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. *ACM SIGGRAPH Computer Graphics*, 18(3):137–145, 1984.

[3] Michael Potmesil and Indranil Chakravarty. A lens and aperture camera model for synthetic image generation. *ACM SIGGRAPH Computer Graphics*, 15(3):297–305, 1981.

[4] Shang Wu, Kai Yu, Bin Sheng, Feiyue Huang, Feng Gao, and Lizhuang Ma. Accurate depth-of-field rendering using adaptive bilateral depth filtering. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7633 LNCS:258–265, 2012.

[5] Michael Kass, Aaron Lefohn, and John Owens. Interactive Depth of Field Using Simulated Diffusion on a GPU. *Computing*, pages 1–8, 2006.