Depth of Field Rendering Algorithms for Virtual Reality

Robert Konrad
Electrical and Computer Engineering
Stanford University
Email: rkkonrad@stanford.edu

I. Introduction

Depth of field, both a focus and depth cue, is not currently implemented in wearable virtual reality displays today even though there is evidence that suggests it could significantly improve the immersive experience. In this project I have investigated different depth of field rendering algorithms that give accurate reproductions of depth of field (retinal blur) on scenes displayed in virtual reality.

Depth of field, also known as the effective focus range, is the distance between the nearest and farthest objects in a scene that appear to be in focus. Even though lenses are only able to focus at precisely on distance, the decrease in sharpness is gradual around around the focal plane. Within a certain range, the sensor (the retina in the case of the human visual system) is not able to register these small changes. Depth of field is a property of optics in the physical world and the human visual system has evolved to use this information to better understand scenes. Current virtual reality implementations do not implement depth of field rendering and have effectively removed an entire source of information. Because of this, entire scenes are in focus, which of course is perceptually inaccurate. An example of a scene rendered without and with a DOF can be found in Figure 1.

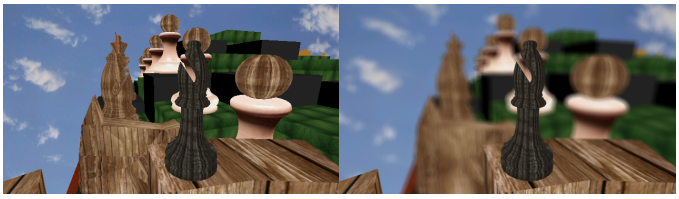


Fig. 1: Images depicting VR scene without and with DoF.

II. Related Work

A. DoF effects on visual system

Not only is the removal of depth of field perceptually inaccurate, but there has been research showing that we have come to rely on this information as a source of cues for both depth and focus.

In [1],it is shown that gaze contingent depth of field rendering can improve subjective perceived realism and depth, which aided users in judging the absolute depth of objects. The paper also presented data showing that a gaze contingent DOF aided in ordering objects and judging distances between objects. Therefore, adding DOF into VR systems would increase our depth perception.

Simulated DOF rendering, when used with stereoscopic displays as shown in [2], has been shown to decrease the fusion time of two images as well as increase fixation stability. The paper developed a method of presenting gaze-contingent blur in order to study the above effects. In [3], it has been shown that a perceptually correct depth of field simulated reduces the amount of visual fatigue experienced by users.

B. Accurate depth of field

Although there have not been any user studies performed comparing the effects of different depth of field simulations on the human visual system, I expect having accurate depth of field is key in achieving the results explained above. In 2008 Barsky compiled a survey of a collection of depth of field rendering techniques used in computer graphics [4].

In the survey, Barsky classifies techniques into two broad categories: objects space and image space methods. Object space methods are built directly into the rendering pipeline and operate on the 3D scene in order to generate depth of field effects. Image space methods, are a post-processing method which operate on images and their corresponding depth maps. The images and depth maps are obtained at the output of the normal rendering pipeline. Each pixel in the image is blurred using information about the camera model and depth map. In general, object space methods create more accurate results and are subject to fewer artifacts than their image space counterparts. However, there is a trade off. Image space methods are much faster. As explained in the next section, speed is critical for VR applications and we therefore are only interested in exploring image space techniques that are able to run in real time ($< 100\ ms$ per frame).

Image space methods must be carefully tuned in order to avoid the following artifacts in the resulting scene. Intensity leakage is the case where a blurred background blurs on top of an in-focus foreground. In a real image, this never occurs and must be handled appropriately by the post-processing technique. Depth discontinuity artifacts occur when the silhouette of a blurred foreground object is sharp, even if the background is in focus. These artifacts occur when the depth map changes abruptly. The final artifact can possibly occur even if a depth discontinuity is handled correctly. In real image, a blurred foreground will have soft edges and some portions of the background will be visible. Because image space methods operate on images generated from an effective pinhole camera,

the colors behind the foreground object are not known, and this artifact cannot be handled accurately with only one image. Ideally, all of these artifacts would be accounted for, but this requires the creation of complex and often time consuming methods.

C. Latency

As shown above, adding an accurate depth of field effect back into stereoscopic displays, like current virtual reality displays, is shown to have benefits for depth perception, fusion time, and visual fatigue. However, simulating an accurate depth of field requires at least one additional rendering pass in the rendering pipeline. This additional time is critical. Virtual reality systems today are very sensitive to latency of the display. If the latency to display is too high, a lag develops between head motion and what we see. This conflict between the vestibular system (inner ear) and the visual system has been shown to develop strong sickness in many cases. Also, as shown in [5], a low latency system provides a more immersive experience for the users. In the paper, users were presented two rooms, one non-threatening and one designed to evoke fear/stress, at different latencies. They showed that users under the lower latency system had a higher self-reported sense of presence and a higher change in heart rate between the two rooms than those exposed to high latency conditions. Therefore, although we are attempting to create a perceptually accurate blur we are constrained by the latency induced by the additional rendering.

III. Depth of field algorithms

A depth of field is created when light traveling along different paths through an aperture of a lens is integrated on the ”sensor” (retina in the case of the human visual system). All depth of field simulations somehow approximate this fact. Solutions range from stochastic sampling with ray tracing to basic filtering based on circles of confusion [6].

Because of the limitations on latency, I focused on image space DOF techniques. I implemented and compared the accuracy and latency of three different image space methods. I bench marked the three methods against a reference depth of field result generated by an accurate, but slow, object space method. The methods are described in the following sections.

A. Reference Depth of Field

In order to create a reference image, I used an object space technique that is a form of ray tracing, and implemented it in OpenGL. A scene is rendered from various discrete points over the aperture, simulating light traveling along different paths. The perspective matrix is adjusted at each sample location. I used 55 sample locations, which can be viewed in Figure 2. These 55 images were then averaged together, forming the reference depth of field result. This object space method is slow, because the scene must be rendered 55 times for just one output frame, but does not suffer from the artifacts named above.

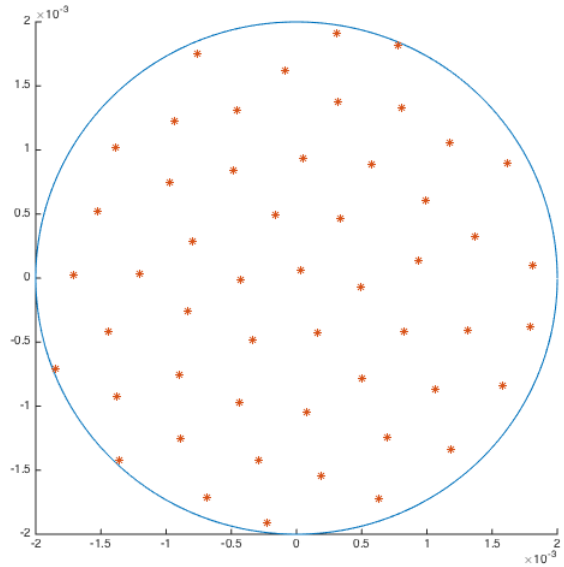
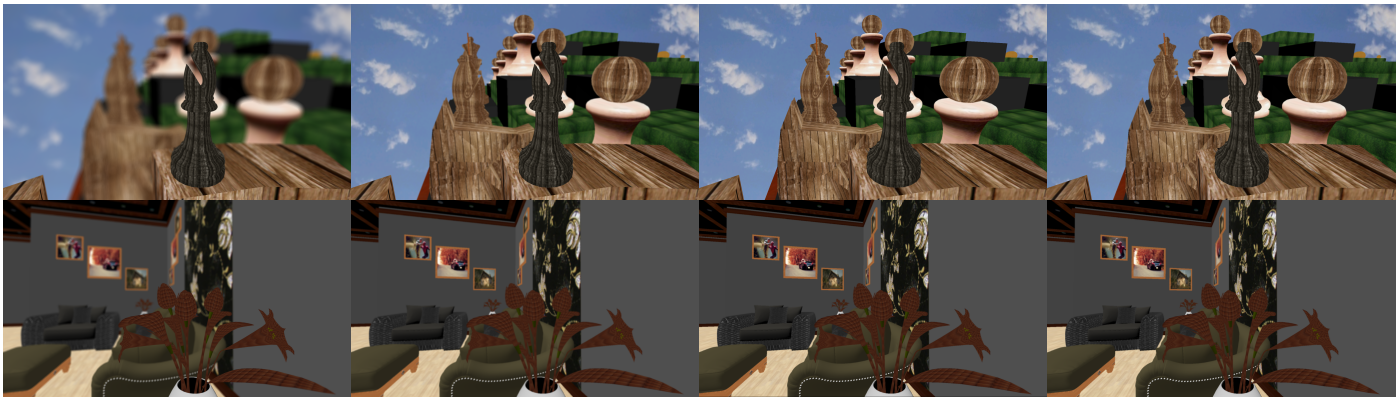


Fig. 2: Positions of 55 sample locations used to sample aperture

B. Adaptive Circle of Confusion Based Filter

The adaptive circle of confusion based filter closely follows the filter described in [7]. This adaptive filter chooses weights dynamically based on the circle of confusion of each pixel and is able to account for intensity leakage. The weights at each pixel are chosen based on 3 factors. The first factor dictates whether a sample pixel effects the center pixel (of the filter). A sample pixel P makes no contribution to a center pixel C if its circle of confusion does not overlap with C. The circle of confusion (originally defined on the focal plane) must be scaled into pixel space. The second factor is an approximation to the light intensity function which is too complex to implement practically in a pixel shader. It is approximated as the reciprocal of the square of the radius, assuming a uniform intensity across the circle of confusion. The third factor directly addresses intensity leakage which reduces the effect that a sampled pixel P has on center pixel C, if P is ”behind” the focal plane. If this is the case then P is scaled by a factor proportional to the radius of the circle of confusion of the center pixel. Effectively, if C is close to the focal plane, its circle of confusion will be very small hence attenuating P greatly. If C is away from the focal plane the attenuation factor will be small. If P is closer to the camera than the focal plane then its value is unchanged.

The paper proposed approximating this adaptive filter as a separable filter, using two 1-D filters, in order to reduce latency. However, this filter is blatantly non-separable. Because graphics compute power has increased greatly since 2007, I decided not to use the approximation in order to gain better accuracy. Compute power has increased so much in fact that the non-separable version rendering a 1080p screen was able to outperform the separable version, running on 2007 hardware, rendering a $512 \times 512\ px$ image.



(a) Reference (b) Adaptive 2D CoC (c) Bilateral Filtering (d) Adaptive Recursive Filtering

Fig. 3: Images showing the the outputs of the different algorithms for the two implemented scenes. Both of the scenes meant to have the foreground in focus: the dark chess piece in the top row and the wooden plant in the bottom row.

	Adaptive CoC Filter	Bilateral Filter	Adaptive Rec. Filter
PSNR (dB)	**30.9953**	29.8669	30.0275
HDR-VDP-2	**59.8957**	56.9642	58.4918
Latency per pixel (ms)	0.559	**0.065121**	0.111666

TABLE I: Comparison of Depth of Field Rendering Algorithms.

C. Bilateral Filter

Many post-processing techniques based on Gaussian filtering suffer from intensity leakage, because edges are not preserved in Gaussian filtering and blurred background pixels bleed into in-focus foreground pixels. I took advantage of the edge preserving capability of the bilateral filter proposed by [8]. In order to prevent intensity leakage, which occurs along depth edges, I replaced the range filter with a depth filter which is used if a pixel is closer than the focal plane. This accounts for intensity leakage.

D. Adaptive Recursive Filtering

The final image space method that I used is an adaptive recursive filtering technique described in [9]. The filter is interesting because it does not approach the problem with the standard a standard 2D filter. Instead it recursively filters an image first from left-to-right, then from right-to-left, then top-to-bottom, and finally bottom-to-top. This process is performed three times to produce the desired result.

First, the circle of confusion of each pixel is computed. Then the scene is segmented into 3 regions: foreground out-of-focus (FOR), in-focus (IR), and background out-of-focus. Depending on the regions of two adjacent pixels, different weights are assigned to the pair. Four different cases exist: both pixels are in IR,FOR,or BOR; one is in IR and the other in FOR; one is int IR and the other in BOR; and one is in FOR and the other in BOR. I used the same weighting functions defined in the paper. The weights only have to be computed once for the left to right and right to left recursion, and once for the top to bottom and bottom to top recursion, because the weights are only dependent on the depth of the pixels which do not change in the recursive process.

IV. Results

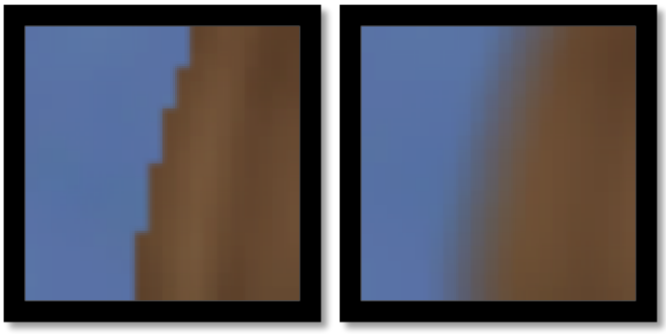
As mentioned earlier, I gather the images and depth maps, along with the images sampled over the aperture to generate the reference image, in C++ using the OpenGL library. Using these images and depth maps, I implemented each of the image space techniques in Matlab and measured their ”distance” to the reference image using two metrics, PSNR between the luma components of the image in Ycbcr color space (because we are most sensitive to the luminosity of images) and the HDR-VDP-2 metric defined in [10]. The HDR-VDP-2 metric is based on a calibrated visual model that can reliably predict visibility and quality differences between image pairs. I computed these results for two different scenes, with two different focal planes for each scene: near and far. Overall PSNR and HDR-VDP-2 values were generated for all 4 scenarios. The metrics along with computation time (per pixel) can be seen in table I. The computation times are not meant to be representative of how quickly these methods will run on a GPU, but rather to see the relative performance of the algorithms. As a benchmark, the adaptive filter based on circles of confusion algorithm was implemented in OpenGL and takes 7.67 ms to blur an entire frame. Per pixel performance is measured because the depth of field algorithms will be implemented on the GPU and operate on each pixel individually.

As seen in the table, the Adaptive 2D filter based on circles of confusion performs best both in terms of PSNR and HDR-VDP-2, but is also the slowest. The bilateral filtering algorithm performs the worst, but also happens to be the fastest method. The adaptive recursive filter performs slightly worse than the Adaptive CoC filter, but shows a significant decrease in computation time.

V. Discussion

The errors in the bilateral filtering algorithm can be partially explained by preserving all edge discontinuities, even ones that should be blurred. For example, in Figure 4 the edge is part of an out of focus background, but its edge is perfectly preserved. This edge should be blurred, and is handled correctly by the Adaptive CoC filter. After looking through the literature, this error can apparently be corrected by creating an adaptive bilateral filter that takes into the account the distance of a pixel from the focal plane. This would reduce the effect of the depth filter for edges that are far from the focal plane.

For any of these depth of field algorithms to have an impact on the state of VR today, eye tracking must be made available. The focal plane must be set dynamically as the user looks around the screen. There is work being done on eye tracking for headmounted VR displays, but there are still very few options. For the demo, I showed a quick and easy hack to show users the effect of depth of field rendering. I averaged over the middle $30 \times 30\ px$ region of each eye using gaussian weightings to choose the focal plane. If the users were able to look towards the middle of the screen, they could move their heads around the scenes to control the depth of field rendering. This of course is not a good replacement to eye tracking, as people tend to fixate on objects as they move around the scene, but users can have a better idea on the effect of depth of field rendering



(a) Bilateral Algorithm (b) Adaptive COC Algorithm

Fig. 4: Image showing artifact created by the Bilateral Filtering Algorithm.

VI. Conclusion

I explored different depth of field rendering algorithms for the purpose of VR applications. It has been shown that adding depth of field rendering to stereoscopic displays improves depth perception, reduces fusion time of images, and reduces visual fatigue. At the same time, the overall latency of the overall system must be kept to a minimum to minimize sickness and maximize immersion. and observed their accuracy when compared to a reference image as well as their computation time. I compared three different depth of field algorithms based on 2 different distance metrics to a reference image as well as the per pixel computation time. An interesting future work could investigate which of the artifacts created via image space methods has the largest impact on our visual system. Perhaps certain artifacts are insignificant in our perception of depth or in fusing images, and can be allowed to appear in the result resulting in a less complex and, perhaps, faster algorithm.

References

[1] M. Mauderer, S. Conte, M. a. Nacenta, and D. Vishwanath, "Depth perception with gaze-contingent depth of field," *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, pp. 217–226, 2014.

[2] G. Maiello, M. Chessa, F. Solari, and P. J. Bex, "Simulated disparity and peripheral blur interact during binocular fusion Guido Maiello," vol. 14, pp. 1–14, 2014.

[3] D. M. Hoffman, A. R. Girshick, and M. S. Banks, "Vergence accommodation conflicts hinder visual performance and cause visual fatigue," *Journal of Vision*, vol. 8, pp. 1–30, 2008.

[4] B. a. Barsky and T. J. Kosloff, "Algorithms for Rendering Depth of Field Effects in Computer Graphics," *World Scientific and Engineering Academy and Society (WSEAS)*, pp. 999–1010, 2008.

[5] M. Meehan, B. Insko, M. Whitton, and F. P. Brooks, "Physiological measures of presence in stressful virtual environments," *ACM Transactions on Graphics*, vol. 21, no. 3, 2002.

[6] M. Potmesil and I. Chakravarty, "A lens and aperture camera model for synthetic image generation," *ACM SIGGRAPH Computer Graphics*, vol. 15, no. 3, pp. 297–305, 1981.

[7] T. Zhou, J. X. Chen, and M. Pullen, "Accurate Depth of Field Simulation in Real Time," *Time*, vol. 26, no. 1, pp. 15–23, 2007.

[8] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Computer Vision, 1998. Sixth International Conference on*, pp. 839–846, Jan 1998.

[9] S. Xu, X. Mei, W. Dong, X. Sun, X. Shen, and X. Zhang, "Depth of Field Rendering via Adaptive Recursive Filtering," 2014.

[10] R. Mantiuk, K. J. Kim, A. G. Rempel, and W. Heidrich, "Hdr-Vdp-2," *ACM SIGGRAPH 2011 papers on - SIGGRAPH '11*, vol. 1, no. 212, p. 1, 2011.