An Algorithm for Rendering Generalized Depth of Field Effects Based on Simulated Heat Diffusion

Todd J. Kosloff[1] and Brian A. Barsky[2]

[1] University of California, Berkeley
Computer Science Division
Berkeley, CA 94720-1776
USA
koslofto@cs.berkeley.edu
[2] University of California, Berkeley
Computer Science Division and School of Optometry
Berkeley, CA 94720-1776
USA
barsky@cs.berkeley.edu

Abstract. *Depth of field* refers to the swath through a 3D scene that is imaged in acceptable focus through an optics system, such as a camera lens. Control over depth of field is an important artistic tool that can be used to emphasize the subject of a photograph. In a real camera, the control over depth of field is limited by the nature of the image formation process and by physical constraints. The depth of field effect has been simulated in computer graphics, but with the same limited control as found in real camera lenses. In this paper, we use *diffusion* in a non-homogeneous medium to generalize depth of field in computer graphics by enabling the user to independently specify the degree of blur at each point in three-dimensional space. Generalized depth of field provides a novel tool to emphasize an area of interest within a 3D scene, to pick objects out of a crowd, and to render a busy, complex picture more understandable by focusing only on relevant details that may be scattered throughout the scene. Our algorithm operates by blurring a sequence of nonplanar layers that form the scene. Choosing a suitable blur algorithm for the layers is critical; thus, we develop appropriate blur semantics such that the blur algorithm will properly generalize depth of field. We found that diffusion in a non-homogeneous medium is the process that best suits these semantics.

1 Introduction

Control over what is in focus and what is not in focus in an image is an important artistic tool. The range of depth in a 3D scene that is imaged in sufficient focus through an optics system, such as a camera lens, is called *depth of field* [1][2][3]. This forms a swath through a 3D scene that is bounded by two planes that are both parallel to the film/image plane of the camera, except in the case of the view camera [4][5][6][7][8][9].

In a real camera, depth of field is controlled by three quantities: the distance at which the camera lens is focused, the f/stop of the lens, and the focal length of the lens.

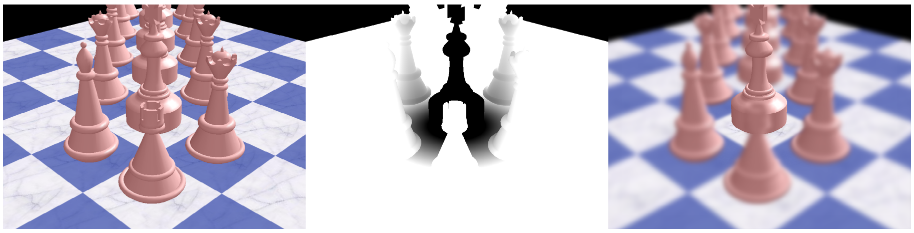

Fig. 1. Left: An unblurred chess scene rendered with a pinhole camera model. Middle: A cylindrical blur field. Right: Application of this blur field to the chess scene.

Professional photographers or cinematographers often control these adjustments to achieve desired effects in the image. For example, by restricting only part of a scene to be in focus, the viewer or the audience automatically attends primarily to that portion of the scene. Analogously, *pulling focus* in a movie directs the viewer to look at different places in the scene, following the point of focus as it moves continuously within the scene.

Creating computer generated imagery can be regarded as simulating the photographic process within a virtual environment. Rendering algorithms in computer graphics that lack depth of field are in fact modeling a pinhole camera model. Without depth of field, everything appears in completely sharp focus, leading to an unnatural, overly crisp appearance. Depth of field effects were first introduced into computer graphics as a means for increasing the realism of computer generated images. Just as in photography, depth of field is also used in computer graphics for controlling what portion of a scene is to be emphasized.

Real depth of field is discussed in Section 3.3. There are significant limitations on what can be achieved by adjusting the focus distance, f/stop, and focal length. It is evident that any given camera lens will have physical constraints that will limit the range of the adjustments. However, even if that were not the case, and we had at our disposal a magical lens that was unlimited in terms of these attributes, there are still many interesting and useful depth of field effects that could be imagined, but not realized by any combination of these adjustments, due to the nature of the image formation process.

When we allow our thinking to extend to what might be desirable, rather than what we have available, we can imagine useful depth of field effects that are not possible with conventional optics. Consider, for example, a scene consisting of a crowd of people. We want to draw attention to a person towards the front, and a second person towards the rear. We have to choose between focusing on the near person or the far person. If a sufficiently small aperture is available, we could perhaps get the entire crowd in focus, but this does not satisfy the goal of picking just the two people we have in mind. Instead, we would like to focus on the near person and the far person, while leaving the middle region out of focus. This cannot be done with a lens. Going even farther with this idea, consider a single row of people, arranged from left to right across the field of view. Since they are all at the same distance from the camera, they will all either be in focus, or all out of focus. There is no way to vary depth of field to focus in on any one of these people and not on the others.

In the case of computer graphics, we are still faced with precisely the same limitations as in real photography, since existing depth of field simulation algorithms were designed intentionally to precisely emulate real-world optics. We alleviate these limitations with the first implementation of an effect that we call *generalized depth of field*. In our system, focus is controlled not by the focus distance, f/stop, and focal length settings, but instead by a three-dimensional scalar blur field imposed on a scene. This approach enables the user to specify the amount of blur independently at every point in space.

Such a system overcomes many of the limitations on what can be achieved with focus. We can control the amount of depth of field independently from the amount of blur away from the in-focus region. Any number of different depths could be made to be in focus, while regions in between those depths will appear blurred. The amount of blur can be made to vary laterally, rather than only with depth. Even more exotic possibilities are just as easy with our system, such as contorting the in-focus region into the shape of a sphere, or any other geometric shape.

Our algorithm proceeds by first rendering the scene into a set of layers, which do not need to be planar. The 3D world coordinates of each pixel are stored in a position map associated with the layers. The position map is used to connect the user-specified 3D blur field to the layers. As is explained in Section 7.2 , blur values are also required that lie outside the object that is present within a layer. These blur values are extrapolated from the known values obtained through the aforementioned mapping. Having associated blur values with each pixel in the layers and having extrapolated those values, we use the blur values associated with each layer to blur the layers. Finally, the layers are composited from front to back, with alpha blending [10].

One difficulty is that naive spatially varying blur (a straightforward linear filter with a spatially varying kernel radius) produces artifacts in the form of holes appearing in the middle of objects. To solve this required a careful analysis of the true meaning of the blur field. This meaning, which we refer to as blur semantics, reflects our analysis of what we consider to be the essential characteristics of realistic depth of field that we wish to maintain, despite our goal of creating effects that cannot occur in the real world. The key insight is that the spatially variant blur operation that will satisfy our blur semantics is *diffusion in a non-homogeneous medium*.

2 Motivation for Generalized Depth of Field

2.1 Limitations of Real Depth of Field

When using depth of field as a tool for selecting only the relevant portions of a scene, there are limitations. If two people are next to each another, it is not possible to focus on just one or the other; both people are at the same distance from the camera, and hence both will be either in focus or out of focus. Alternatively, consider a crowd of many people. Perhaps we would like to use focus to highlight a few people scattered at a few locations throughout the crowd. There is clearly no combination of focus distance, f/stop, and focal length that can achieve this.

When simulating depth of field for computer generated images, the goal has generally been to faithfully replicate the behavior of real cameras, complete with these limitations. On the other hand, we observe that there is no reason to accept these limitations in computer graphics. Rather than control the focus distance, f/stop, and focal length, we allow depth of field to be controlled by a 3D blur field that specifies how much blur should be applied to every point in 3D space.

2.2 Partial Occlusion

Since a lens has a finite aperture, that which is imaged at any given point on the film/image plane is the aggregate of the light emerging from every point on the lens. Thus, the lens can be considered as viewing the scene from many different points of view, simultaneously. Specifically, light rays impinging at the center of the lens may be emanating from foreground objects that occlude background objects whereas other rays arriving at a point on the lens far from its center may bypass the occluder. Thus, the background is visible from some points on the lens, but not from others. Therefore, there are single points on the image where a given object is *partially occluded*. This can be seen when objects that are out-of-focus have soft, semi-transparent edges. Partial occlusion is a vital aspect in the appearance of depth of field.

A computer simulation of depth of field that lacks partial occlusion does not faithfully reproduce the appearance of real photographs. Consider, for example, a small object that is close to the camera with a large aperture. Partial occlusion would be very noticeable because it would apply to the entire object and not just to the edge. In such cases, the lack of partial occlusion is completely unacceptable.

3 Background

3.1 Basic Lens Parameters

The focal length of a lens, denoted by f, is the distance the lens must be from the film/image plane to focus incoming parallel rays to a point on that plane. The aperture is the physical opening of the lens, which determines the quantity of light that will enter the optical system. We denote the aperture diameter by a_{diam}. The f/stop (or F number) is the ratio of the focal length to the diameter of the aperture of the lens, $\frac{f}{a_{diam}}$. Note that the aperture diameter is not completely specified by the f/stop. Indeed, when changing the focal length on a zoom lens, the f/stop varies in many amateur lenses, whereas it is the aperture that varies in in more professional zoom lenses.

3.2 Circle of Confusion

Blur arises when a point in the 3D scene is not imaged as a point, but instead spreads to form a disk, which is called the *circle of confusion*. The amount of blur for a given depth can be described by the diameter of the circle of confusion for that depth.

Consider a point that we wish to be in focus, located at a distance d_{focus} in front of the lens. Using the thin lens approximation [11], the distance d'_{focus} behind the lens where the film/image plane would have to be located can be derived [12]:

$$d'_{focus} = \frac{f * d_{focus}}{d_{focus} - f} \tag{1}$$

where f is focal length.

A point that is not at the distance d_{focus} in front of the lens will thus not be imaged as a point at the distance d'_{focus} behind the lens where the film/image plane is, but instead would form a point at the distance d_{image} behind the lens. Thus, on the film/image plane, it would be imaged as a circle of confusion having a diameter denoted by c_{diam}. Using similar triangles, this can be calculated [12]:

$$c_{diam} = a_{diam} * \frac{d_{image} - d'_{focus}}{d_{image}} \tag{2}$$

where a_{diam} is the aperture diameter. From this, we see that the circle of confusion diameter is intrinsically independent of focal length, although focal length would enter into the equation if it were recast in terms of f/stop rather than aperture diameter.

3.3 Real Depth of Field

In an optical system such as a camera lens, there is a plane in the 3D scene, located at the focus distance, that is rendered at optimal sharpness. There is a swath of volume through the scene that is rendered in reasonable sharpness, within a permissible circle of confusion, to be exact. This region of acceptable focus is delineated by near and far planes. However, this region is not centered at the focus distance; rather, the near plane is closer to the plane of perfect focus than is the far plane.

The particular distance that is imaged in perfect focus can be controlled by moving the lens towards or away from the film/image plane. Changing the focus distance will have a concomitant effect on the amount of depth of field, that is, the size of the swath. Specifically, for a given f/stop and focal length, focusing at a distance that is close to the camera provides only a narrow range of depths being in focus, and the amount of depth of field increases in a nonlinear fashion as the focus distance is increased, and conversely.

The amount of depth of field is also affected by The size of the aperture. The infinitesimal aperture of a pinhole camera has infinite depth of field, and this decreases as the aperture increases, and conversely.

Less widely discussed is the behavior of blur outside the region of acceptable focus. Objects that are at an increasing distance from the focus distance become increasingly blurred at a rate related to the aperture size. For very large aperture size, not only is there a very narrow range of depths that are imaged in focus, but the out-of-focus areas have an extreme amount of blur. This arises frequently in the case of macro-photography where the subject is very close to the camera. Although these relationships are well understood by experienced photographers, they are not obvious to novices. Furthermore, no amount of experience in photography can enable an escape from the limitations imposed by the inherent image formation process.

4 Related Work

The first simulation of depth of field in computer graphics was developed by Potmesil and Chakravarty who used a postprocessing approach where a single sharp image is blurred using a spatially variant linear filter [13]. Postprocessing a single image can lead to occlusion artifacts. One such artifact occurs when background pixels are spread onto foreground pixels. Shinya solved this with a ray distribution buffer, or RDB. This approach, rather than blindly averaging adjacent pixels, stores pixels in a z-buffered RDB as they are being averaged. The z-buffer in the RDB ensures that near pixels will occlude far pixels during the averaging process. Rokita [14] achieved depth of field at rates suitable for virtual reality applications by repeated convolution with 3×3 filters and also provided a survey of depth of field techniques.

Cook introduced distributed raytracing [15], the first multisampling approach to depth of field. Several rays are traced for each pixel to be rendered. Each ray originates from a different point on the lens, and the rays are oriented such that they intersect at the plane of sharp focus. Kolb et al. performed distributed ray tracing through detailed lens models corresponding to actual lenses [16]. The resulting images exhibit the appropriate distortion and blur inherent in these lenses.

Haeberli and Akeley introduced the accumulation buffer [17], which is present on modern graphics hardware. The accumulation buffer allows several renders of a single scene to be accumulated. Depth of field is attained by accumulating aperture samples.

Scofield presents a fast and simple depth of field algorithm that, like ours, postprocesses layers, then composites the layers from back to front [18]. His method, like ours, uses layers to solve the partial occlusion problem. Scofield's technique, unlike ours, assumes that each layer lies on a plane parallel to the film/image plane. Blur is uniform within a given layer so the blurring can be performed efficently using an FFT.

Kosara, Miksch, and Hauser were the first to suggest that by departing from the physical laws governing depth of field, blur can become a much more useful tool for indicating the relevant parts of a scene [19]. They call their system *semantic depth of field*, indicating that blur is a meaningful design element, rather than merely an artifact of the way lenses work. Their system operates in a similar manner to Scofield's. That is, objects are rendered into buffers, then the buffers are blurred according to the relevance of the objects, and finally the buffers are composited. This approach, while fast and simple, operates at object-level granularity. There is no way to blur only half an object, nor can blur vary smoothly across the surface of an object. Thus, it is impossible to have a window of focus smoothly move across a scene. On the other hand, we support a fully generalized blur field, where every point in 3D space can be blurred or not blurred as desired.

Depth of field is relevant in all optical systems, not only for camera lenses. The human eye also has a finite aperture (which is the pupil) and it can focus at different depths by means of accommodation of the internal crystalline lens. The optical characteristics of human eyes are slightly different for each person, resulting in different point spread functions from person to person. Barsky introduced vision realistic rendering to the computer graphics community [20], which uses actual data scanned from human eyes measured a wavefront aberrometer. The aberrometer samples a wavefront, which is then interpolated using the Zernike polynomial basis. The wavefront is converted into a set of point spread functions, which are then used to blur different parts of an image, using a postprocessing method. Further details about that postprocessing method are discussed in detail by Barsky et al. [21][22].

Isaksen et al. developed a new parameterization for light fields that enables controllable depth of field [23]. It is especially relevant here that their parameterization includes a focal surface that is not planar. Therefore, they can control focus in ways beyond what a real camera can produce. They do not completely generalize depth of field, however, focal surfaces are merely a subset of what can be achieved with our arbitrary blur fields.

Krivanek developed a very fast method for rendering point cloud models with depth of field via splatting [24]. This elegant method simply replaces each splat footprint with a Gaussian whose standard deviation increases with circle of confusion. Rendering these large splats does not lead to a slowdown, as it is possible to render large splats using a simplified version of the model comprising fewer points, without substantial loss in quality.

Perona and Malik used anisotropic diffusion to blur images in a nonuniform fashion [25][26]. Their purpose in doing so is for performing edge detection. We also use diffusion for blurring images in a nonuniform fashion, though for a completely different reason.

Bertalmio et al., [27] used diffusion to alleviate intensity leakage. They did not, however, address disocclusion of the background.

Whereas cameras whose film/image plane is parallel to the lens plane follow the conventional rules for predicting circle of confusion, view cameras do not follow these rules [5][6][7][8][9][28]. A view camera can be adjusted via tilts and swings to have its film/image and lens planes at arbitrary angles relative to one another. This results both in a change in the perspective projection and a change in depth of field. A view camera enables the plane of sharp focus to be oriented in a general manner such that it does not have to be parallel to the film/image plane. Thus, the plane that is in sharp focus might be an entire kitchen floor, even though the floor spans a great range of depths. Barsky and Pasztor developed methods for simulating the view camera camera model for computer generated images [4]. One approach used distributed ray tracing and the other used the accumulation buffer. Generalized depth of field can easily simulate a view camera, using an appropiate blur field as input.

For a more thorough survey of existing depth of field techniques, we refer the reader to a pair of surveys by Barsky et al. where the techniques have been separated into object space [12] and image space [29] techniques.

None of these depth of field algorithms provide a technique that can be modified to achieve our goal of arbitrary blur fields, as existing methods are too closely tied to the conventional image formation process to be coerced into producing the generalized blur effects we desire.

5 Blur Field Semantics

The selection of a 2D blur operation to apply to the layers is based not only on the need to apply arbitrary blur fields, but also on the need to smoothly generalize realistic depth of field.

The relevant property of depth of field relates to partial occlusion along the edges of an out-of-focus object. Since a camera lens has a finite aperture, the edges of a blurred object become semi-transparent. We require this effect. However, we must decide how partial occlusion should behave in the presence of arbitrary blur fields. There is no physically-realizable camera or camera model to guide us here.

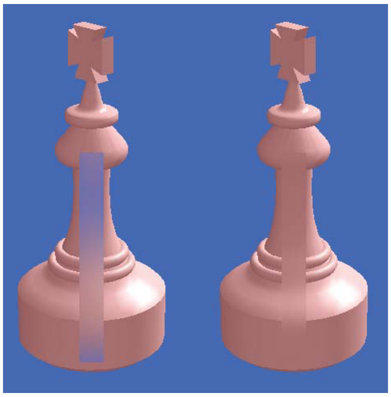

Fig. 2. Left: Naive linear filtering. Notice how the blurred region is transparent. Right: Diffusion. Notice how the blurred region remains opaque, as it should.

First, we explored a naive spatially variant linear filter on each layer. This results in images that are more blurred or less blurred in accordance with the blur field, and indeed we obtain partial transparency along the edges. However, transparency also occurred in isolated blurred regions interior to the foreground object. This does not correspond to anything present in realistic depth of field, and this is a highly objectionable artifact. Figure 2 (left) shows an example of such a hole.

We now examine the cause of this artifact and a solution to it. We represent transparency using RGBA pixel values [10], where transparent pixels have values of alpha less than 1. The initial unblurred image has some region in which the object lies. Inside this region, alpha is always 1 (in the case of opaque objects). Outside the object, the pixels have alpha of 0, indicating that nothing is there. During filtering, blurred pixels near the edge are averages of both the interior and the exterior of the object, leading to alpha values between 0 and 1. This results in a reasonable approximation to the partial occlusion found in depth of field. However, when the blur value inside an object is large, and the blur value towards the edges of the object are small, then the interior of the object may be averaged with pixels outside the object, despite the fact that the edge pixels themselves may remain perfectly sharp. We would avoid this artifact if the low-blur pixels on the edge acted as barriers through which colors cannot leak. This notion leads us to the key property that our 2D blur algorithm must possess: the averaging process must be aware of the blur values along the *entire path* that a color must traverse in order to reach a destination.

This property is inherently upheld by heat diffusion. Consider heat transfer on a sheet of metal. Making the analogy between heat and brightness in an image, imagine the idea of "painting a grayscale image" on the metal by heating some areas more than others. Over time, the heat diffuses, blurring the image. If the metal is more conductive in some places and less conductive in others, then at any given time, some portions of the image will be more blurred than others. Most importantly, if there is a barrier of low conductivity between two regions of high conductivity, then no heat will flow from one region of high conductivity to the other. Therefore we take inspiration from heat diffusion in building our 2D blur operation. Figure 2 (right) shows how regions of low conductivity prevent a hole from appearing inside an object.

6 Algorithm Overview

In this section we describe the overall structure of our method. First, the scene is rendered as a collection of layers. Associated with each layer is a position map. This position map is required because the pixels within any given layer need not all occupy the same depth. Thus, our layers are more like 3D objects and less like the flat images that the term "layer" may suggest. A position map is similar to a depth map, except that it stores the full three-dimensional world coordinates of the object seen under each pixel. The position map can be visualized as a high dynamic range image, where red, green, and blue correspond to x, y, and z coordinates. Figure 3 shows a typical position map.

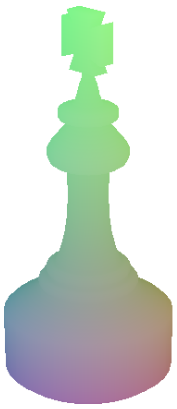

Fig. 3. A typical position map. This is a linearly scaled and clamped high dynamic range image. The red, green, and blue color channels represent x, y, and z world coordinates, respectively.

We will be applying a spatially varying blur operator to each layer, which is an inherently two-dimensional process. However, the blur field exists in three dimensions. We need a two-dimensional blur field for the layer blurring step. These two-dimensional blur fields are nonplanar slices of the three-dimensional blur field. The mapping between the two-dimensional blur field slice and the full three-dimensional blur field is performed by the aforementioned position map.

As will be explained in the next section, blur values are actually needed for some pixels outside the object itself. These pixels are those that lie within a certain distance of the object. We find these pixels by growing the region representing the object within the layer. The position map inherently cannot contain any information for pixels outside the object; thus, these blur values cannot be read from the three-dimensional blur field. Therefore, reasonable blur values must be extrapolated from the known data.

Next, our carefully chosen, two-dimensional spatially varying blur operation is applied to each layer. Finally, the layers are composited from back to front, with alpha blending.

7 Algorithm Details

We now describe the region growing, blur field extrapolation, and diffusion steps in detail.

7.1 Region Growing

Often, only a small portion of a layer is covered. Thus, it would be a waste of computational resources to apply expensive blurring operations to the entire image when most of it is empty. On the other hand, a blurred object expands in size; thus, it is not sufficient only to blur the pixels lying inside the unblurred object. We therefore designate a superset of the object as the region to be blurred. This region is initialized to the object itself. The region is then grown one ring at a time by finding all the pixels lying outside the current region, but in contact with it. Rings are added until the region is sufficiently large to encompass any potential growth caused by the blurring operation. Figure 4 shows an object and the corresponding grown region. It is quite often the case that this expanded region is significantly smaller than the total size of the layer, resulting in a dramatic speedup compared to blurring the entire layer.

7.2 Blur Field Extrapolation

We now have blur values for each pixel within the object within the layer. However, our two-dimensional blur operation requires blur values outside the object, because a blurred image is larger than the original image. The region of the blurred image outside the original object is very important, since this is where the fuzzy edges characteristic of out-of-focus objects are. We cannot look up these blur values directly from the three-dimensional blur field, since there are no appropriate world coordinates present in the position map. Therefore, we must fabricate exterior blur values. This blur extrapolation step must ensure a smooth blur field that seamlessly merges with the interior of the object. Exterior blur values are calculated as a weighted average of the edge blur values, where the weights decrease with increasing distance from the edge point of interest. Note that the weights are normalized. That is, points farther from the edge are not less blurred; these points are merely influenced by the various edge points to a greater or lesser extent. Figure 5 illustrates what a typical extrapolated blur field looks like.

Consider the extrapolated blur value $B_{2D}(x,y)$ at a pixel (x,y) lying outside the object (but inside the expanded region). Let n be the number of pixels on the edge of the object, and g be a weighting function causing the relative contribution of each edge

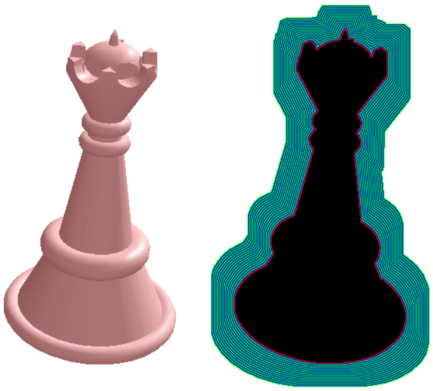

Fig. 4. A chess piece and the surrounding region within which blur will be applied

pixel to decrease with distance. Let (x_i, y_i) denote the i^{th} edge pixel. Then $B_{2D}(x,y)$ can be computed as:

$$B_{2D}(x,y) = \frac{\sum_{i=1}^{n} B_{2D}(x_i,y_i) * g(x_i,y_i,x,y)}{\sum_{i=1}^{n} g(x_i,y_i,x,y)}. \tag{3}$$

where $g = \frac{1}{d(x_i,y_i,x,y)^2}$, and $d(x_i,y_i,x,y)$ is the Euclidean distance between pixels (x_i,y_i) and (x, y); that is, $d = \sqrt{(x_i - x)^2 + (y_i - y)^2}$.

7.3 Blurring a 2D Layer

Two-dimensional spatially varying blur is performed by simulating heat diffusion over a medium with non-homogeneous thermal conductivity.

Whereas we might have started with the partial differential equation describing heat flow [30], then discretized it in space and numerically integrated it through time, our intent is to blur images, not to accurately predict the behavior of heat. Therefore, we use a simple implementation consisting of iterated neighborhood averaging, and we will not discuss the underlying differential equations any further.

A number of iterations are carried out. For each iteration, each pixel is averaged with its four neighbors, where the blur value for each pixel determines the weights in the average.

This neighborhood is iterated so that the object can be effectively blurred by kernels of any size. Repeated local averaging, rather than a single linear filter with a wider kernel, is necessary to prevent colors from leaking across regions of low blur.

Our blur operator works as follows: To calculate the blurred value of pixel (x,y) after one iteration, we average pixel (x,y) with its four neighbors, which we refer to as north, south, east, and west, and denote by C_n, C_s, C_e and C_w, respectively. The pixel at location (x,y) is referred to as "center", and is denoted by C_c. C_i denotes the image after i iterations. First, define W_n, W_s, W_e, and W_w to be the blur field elements, entries from B_{2D}, corresponding to the pixels north, south, east, and west of the pixel in question

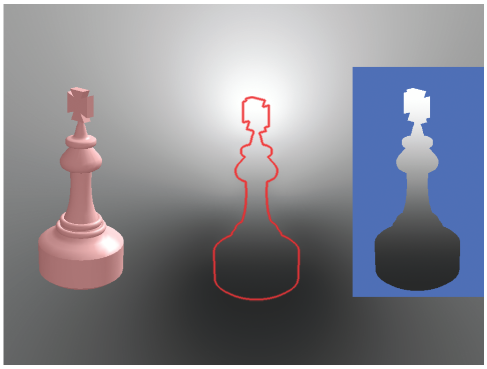

Fig. 5. Left: A layer. Right: The blur field slice for the layer. Middle: The blur values outside the object have been extrapolated.

Now we calculate C_{i+1}, the color of pixel (x,y) after the $(i+1)$'th iteration, using the following formulation:

$$C_c = C_i(x,y)$$

$$C_n = C_i(x,y-i)$$

$$C_s = C_i(x,y+1)$$

$$C_e = C_i(x+1,y)$$

$$C_w = C_i(x-1,y)$$

$$C_{i+1} = W_c * C_c + \frac{1}{4} * W_n * C_n + \frac{1}{4} * W_s * C_s + \frac{1}{4} * W_e * C_e + \frac{1}{4} * W_w * C_w$$

W_c is the weight for the center pixel. It is chosen such that all the weights will sum to 1.0, ensuring no unwanted brightening or dimming even when adjacent blur values are vastly different. That is, $W_c = 1.0 - \frac{W_n - W_s - W_e - W_w}{4}$.

8 Results: An Example

We have applied generalized depth of field to a chessboard scene. A fully in-focus version of this scene was rendered using a pinhole camera model. The scene was rendered into layers and corresponding position maps. The board itself is one layer, and each chess piece forms another layer. Figure 1 shows the original scene on the left, a blur field in the middle, and, on the right, the blurred scene corresponding to that blur field. In this figure, the blur field is a vertical cylinder of focus centered on one chess piece. Figure 6 shows a highly populated chessboard with a "plus"-shaped volume in focus. In Figure 7, we see a visualization of the chess scene intersecting with a blur field. This

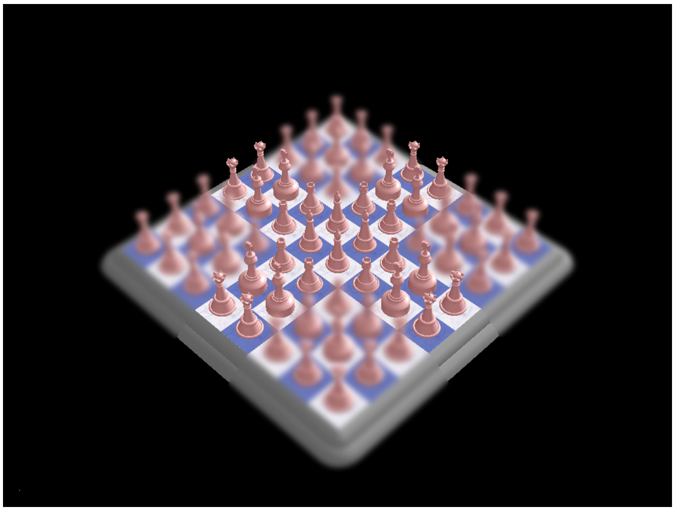

Fig. 6. The volume that is in focus is shaped like a "plus" sign

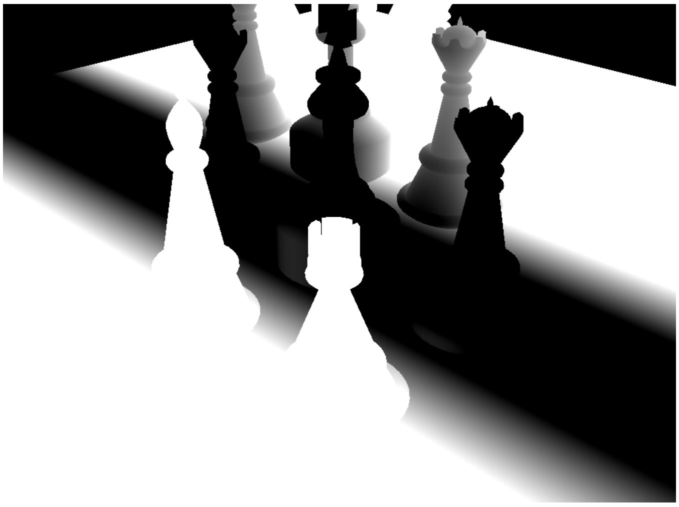

Fig. 7. This is a blur field that has a swath of acceptable focus which is planar, vertical, and aligned exactly with one row of the chessboard

blur field has a volume of focus oriented along a column of the chessboard. Figure 8 shows the blurred scene corresponding to the aforementioned blur field.

By varying the blur field over time, generalized depth of field can be used to create interesting variations on the traditional "pulling focus" For example, the cylindrical focal volume can smoothly glide from one chess piece to another. Alternatively, the

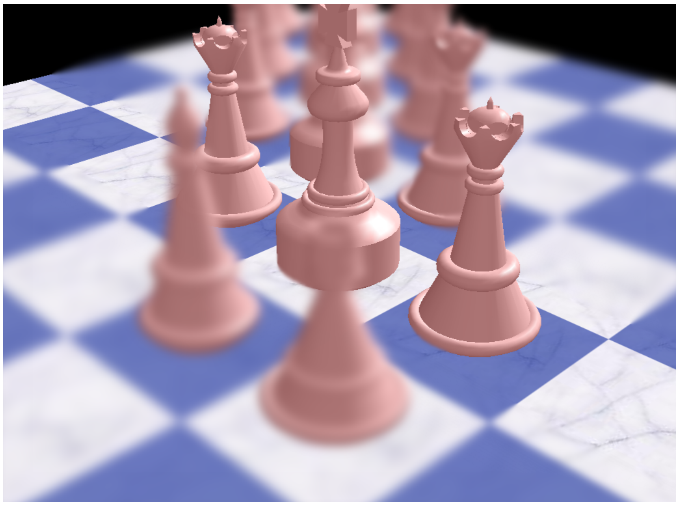

Fig. 8. The chess scene blurred with the swath of acceptable focus that was shown in Figure 7

scene can gradually come into focus as a sphere of focus emerges from the center of the scene and gradually expands in size.

9 Future Work

Currently, the scene must be split into a series of layers in order for our algorithm to achieve partial occlusion. Although any correct handling of partial occlusion would require some representation of depth complexity, our present approach does not work well for scenes that cannot easily be split into layers. Our layers are nonplanar, alleviating an important restriction in previous post-processing approaches to depth of field. The chessboard, for example, is one layer, while in previous methods it would have had to be split into many layers, because it spans a deep range of depths.

However, our method cannot presently handle a complex layer that occludes itself. Depth complexity is handled solely through different layers; hence, there cannot be any depth complexity within a single layer. This problem, however, is not unique to our formulation, but is an issue common to many post-processing approaches to depth of field.

A reasonable way to solve this would be to operate on a layered depth image (LDI) [31]. An LDI records the colors and depths of every object that is hit along each ray emanating from the center of projection. LDIs do not explicitly group the scene into objects or layers, but they do retain the colors of objects that are occluded. Thus, they have the benefits of layers without the disadvantages, and hence would make an excellent representation on which a future version of our algorithm could operate.

Generalized depth of field would be a useful tool in photography. We believe that by processing and combining a series of exposures taken with a camera over a variety of focus settings, and with the help of computer vision techniques to estimate depth, generalized depth of field should be achievable for real world scenes.

10 Conclusion

In this paper, we have generalized depth of field in computer graphics by enabling the user to independently specify the amount of blur at each point in three-dimensional space. Generalized depth of field provides a novel tool to emphasize an area of interest within a 3D scene, to pick objects out of a crowd, and to render a busy, complex picture more understandable by focusing only on relevant details that may be scattered throughout the scene.

It is important to note that our method not only enables new effects, but it can also simulate conventional depth of field as a special case. This can be done easily, by constructing a blur field based on the circle of confusion of a conventional lens. A realistic blur field corresponding to a conventional camera would vary only along the depth axis, remaining constant within planes parallel to the image plane. The variation along the depth axis is related to the circle of confusion. Specifically, the blur value at a given depth is specified by equation 2 in section 3.2. Alternatively, a blur field corresponding to a view camera with tilted plane of sharp focus could be simulated by letting the circle of confusion vary along a tilted axis, rather than the depth axis.

Our approach has advantages over existing approaches for simulating realistic depth of field. Our approach is a post processing technique. Although other post processing methods (e.g. [13]) do not correctly handle occlusion, our approach does correctly handle occlusion, like the more expensive multisampling approaches. Unlike the other postprocessing methods, our approach is not based on the assumption that objects can be approximated as planar layers that are aligned parallel to the image plane, nor does our method suffer from discretization issues. Conversely, multisampling methods handle occlusion properly and do not suffer from discretization issues, but their performance degrades in proportion to the time it takes to render the scene. Thus, a scene that comprises many small polygons with complex shaders, which would already be slow to render due to its complexity, would need to be rendered many times in a multisampling method. On the other hand, postprocessing methods (such as ours) operate on an image based representation that is output by the renderer. Thus, our method requires no more time to process a scene of a million polygons than for a hundred polygons.

Acknowledgements

The chessboard model used to demonstrate our technique was created by Brian Sung Chul Choi and Jean Yi-Chin Lai.

References

1. Erickson, B., Romano, F.: Professional Digital Photography. Prentice-Hall, Englewood Cliffs (1999)
2. London, B., Upton, J., Kobre, K., Brill, B.: Photography, 7th edn. Prentice-Hall, Englewood Cliffs (2002)

3. Stroebel, L., Compton, J., Current, I.: Basic Photographic Materials and Processes, 2nd edn. Focal Press (2000)
4. Barsky, B.A., Pasztor, E.: Rendering skewed plane of sharp focus and associated depth of field. In: SIGGRAPH 2004 Tech Sketch, ACM Press, New York (2004)
5. Merklinger, H.M.: Focusing the View Camera (1993) ISBN 0-9695025-2-4
6. Merklinger, H.M.: View camera focus and depth of field, parts i and ii. View Camer magazine, July/August 1996, pp. 55-57 and September/October, pp. 56–58 (1996)
7. Simmons, S.: Using the View Camera, Amphoto Revised edn., Watson-Guptill Publications (1992)
8. Stone, J.: User's Guide to the View Camera, 3rd edn. Prentice-Hall, Englewood Cliffs (2004)
9. Stroebel, L.: View Camera Technique, 7th edn. Focal Press (1999)
10. Porter, T., Duff, T.: Compositing digital images. In: ACM SIGGRAPH 1984 Conference Proceedings, pp. 253–259 (1984)
11. Jenkins, F.A., White, H.E.: Fundamentals of Optics. McGraw-Hill, New York (1976)
12. Barsky, B.A., Horn, D.R., Klein, S.A., Pang, J.A., Yu, M.: Camera models and optical systems used in computer graphics: Part i, object based techniques. In: Kumar, V., Gavrilova, M., Tan, C.J.K., L'Ecuyer, P. (eds.) ICCSA 2003. LNCS, vol. 2667, pp. 246–255. Springer, Heidelberg (2003)
13. Potmesil, M., Chakravarty, I.: Synthetic image generation with a lens and aperture camera model. ACM Transactions on Graphics 1(2), 85–108 (1982)
14. Rokita, P.: Generating depth of-field effects in virtual reality applications. IEEE Computer Graphics and Applications 16(2), 18–21 (1996)
15. Cook, R.L., Porter, T., Carpenter, L.: Distributed ray tracing. In: ACM SIGGRAPH 1984 Conference Proceedings, pp. 137–145. ACM Press, New York (1984)
16. Kolb, C., Mitchell, D., Hanrahan, P.: A realistic camera model for computer graphics. In: ACM SIGGRAPH 1995 Conference Proceedings, pp. 317–324. ACM Press, New York (1995)
17. Haeberli, P., Akeley, K.: The accumulation buffer: hardware support for high-quality rendering. In: ACM SIGGRAPH 1990 Conference Proceedings, pp. 309–318. ACM Press, New York (1990)
18. Scofield, C.: $2\frac{1}{2} - d$ depth of field simulation for computer animation. In: Graphics Gems III, Morgan Kaufmann, San Francisco (1994)
19. Kosara, R., Miksch, S., Hauser, H.: Semantic depth of field. In: Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01), p. 97 (2001)
20. Barsky, B.A.: Vision-realistic rendering: simulation of the scanned foveal image from wavefront data of human subjects. In: Proceedings of the 1st Symposium on Applied perception in graphics and visualization, pp. 73–81. ACM Press, New York (2004)
21. Barsky, B.A., Tobias, M.J., Horn, D.R., Chu, D.P.: Investigating occlusion and discretization problems in image space blurring techniques. In: First International Conference on Vision, Video, and Graphics, pp. 97–102 (2003)
22. Barsky, B.A., Tobias, M.J., Chu, D.P., Horn, D.R.: Elimination of artifacts due to occlusion and discretization problems in image space blurring techniques. Graphical Models 67(6), 584–599 (2005)
23. Isaksen, A., McMillan, L., Gortler, S.J.: Dynamically reparameterized light fields. In: ACM SIGGRAPH 2000 Conference Proceedings, ACM Press, New York (2000)
24. Krivanek, J., Zara, J., Bouatouch, K.: Fast depth of field rendering with surface splatting. In: Computer Graphics International 2003 (2003)
25. Perona, P., Malik, J.: Scale space and edge detection using anisotropic diffusion. IEEE Trans. Pattern Anal. Mach. Intell. 12(7), 629–639 (1994)
26. Perona, P., Malik, J.: A network for multiscale image segmentation. In: IEEE Int. Symp. on Circuits and Systems, pp. 2565–2568. IEEE Computer Society Press, Los Alamitos (1988)

27. Bertalmio, M., Fort, P., Sanchez-Crespo, D.: Real-time, accurate depth of field using anisotropic diffusion and programmable graphics cards. In: IEEE Second International Symposium on 3DPVT, IEEE Computer Society Press, Los Alamitos (2004)
28. Shaman, H.: The view camera: operations and techniques. American Photographic Book Publishing Co. (1978)
29. Barsky, B.A., Horn, D.R., Klein, S.A., Pang, J.A., Yu, M.: Camera models and optical systems used in computer graphics: Part ii, image based techniques. In: Kumar, V., Gavrilova, M., Tan, C.J.K., L'Ecuyer, P. (eds.) ICCSA 2003. LNCS, vol. 2668, pp. 256–265. Springer, Heidelberg (2003)
30. Carslaw, H.S., Jaeger, J.C.: Conduction of Heat in Solids, 2nd edn. Oxford University Press, Oxford (1959)
31. Shade, J., Gortler, S.J., He, L.-w., Szeliski, R.: Layered depth images. In: Proceedings of ACM SIGGRAPH 1998, ACM Press, New York (1998)
32. Greenleaf, A.R.: Photographic Optics. The Macmillan Company, NYC (1950)
33. Ray, S.F.: Applied Photographic Optics, 3rd edn. Focal Press (2002)
34. Kingslake, R.: Optics in Photography. SPIE – The International Society for Optical Engineering (1992)
35. Williams, J.B.: Image Clarity: High-Resolution Photography. Focal Press (1990)
36. Goldberg, N.: Camera Technology: The Dark Side of the Lens. Academic Press, London (1992)
37. Shinya, M.: Post-filtering for depth of field simulation with ray distribution buffer. In: Proceedings of Graphics Interface '94, Canadian Information Processing Society, pp. 59–66 (1994)
38. Demers, J.: Depth of field: A survey of techniques. In: GPU Gems, pp. 375–390 (2004)
39. Merklinger, H.M.: The Ins and Outs of Focus: Internet Edition (1992), `http://www.trehnholm.org/hmmerk/download.html`
40. Barsky, B.A., Bargteil, A.W., Garcia, D.D., Klein, S.A.: Introducing vision-realistic rendering. In: Eurographics Rendering Workshop, posters, pp. 26–28 (2002)
41. Baker, A.A.: Applied Depth of Field. Focal Press (1985)