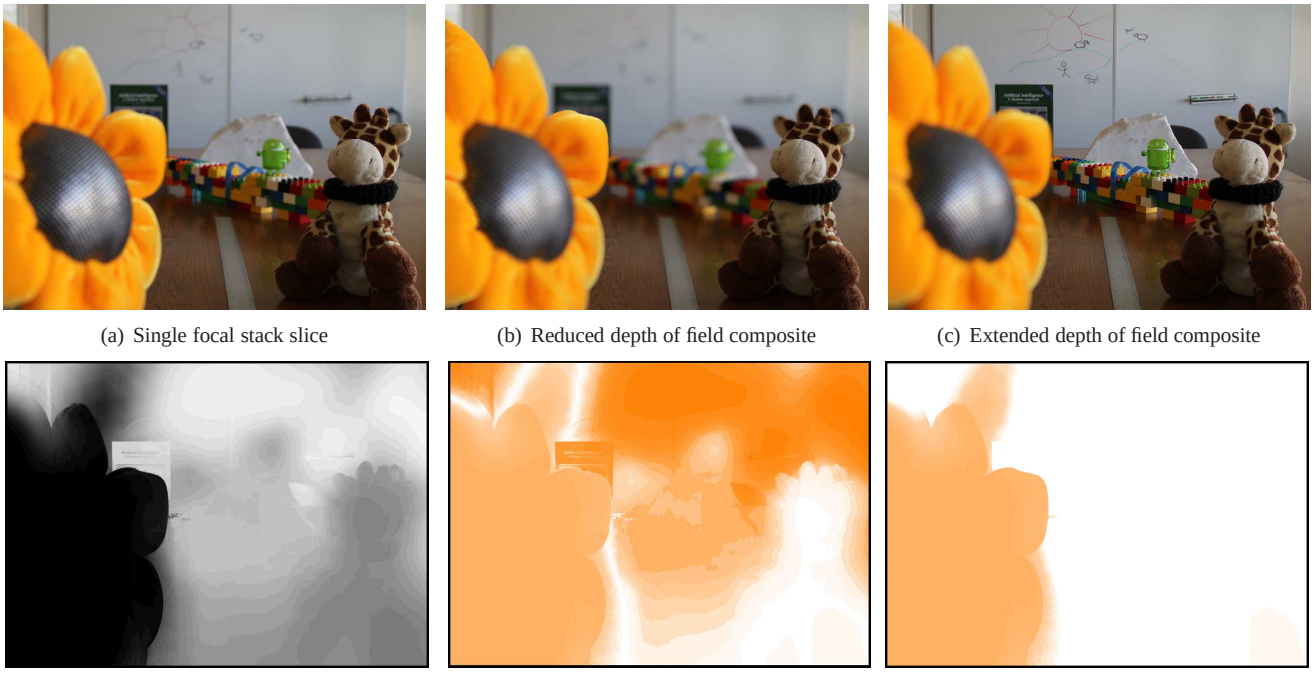
Focal Stack Compositing for Depth of Field Control

David E. Jacobs Jongmin Baek Marc Levoy
Stanford University*



(a) Single focal stack slice (b) Reduced depth of field composite (c) Extended depth of field composite

(d) Scene depth map (dark means close) (e) Defocus maps used to generate the images in (b) and (c), respectively (orange means blurry)

Figure 1: *Manipulating depth of field using a focal stack. (a) A single slice from a focal stack of 32 photographs, captured with a Canon 7D and a 28mm lens at $f/4.5$. The slice shown is focused 64cm away. (b) A simulated $f/2.0$ composite, focused at the same depth. To simulate the additional blur, objects closer to the camera are rendered from a slice focused afar, and objects far from the camera are rendered from a slice focused near. (c) An extended depth of field composite that blurs the foreground flower and is sharp for all depths beyond it. (d) A depth map for the scene, representing depth as image intensity (dark means close.) (e) A pair of defocus maps that encapsulate the requested amount of per-pixel defocus blur used to generate the composites above. Its magnitude is encoded with saturation.*

Abstract

Many cameras provide insufficient control over depth of field. Some have a fixed aperture; others have a variable aperture that is either too small or too large to produce the desired amount of blur. To overcome this limitation, one can capture a focal stack, which is a collection of images each focused at a different depth, then combine these slices to form a single composite that exhibits the desired depth of field. In this paper, we present a theory of focal stack compositing, and algorithms for computing images with extended depth of field, shallower depth of field than the lens aperture naturally provides, or even freeform (non-physical) depth of field. We show that while these composites are subject to halo artifacts, there is a principled methodology for avoiding these artifacts—by feathering a slice selection map according to certain rules before computing the composite image.

CR Categories: I.4.3 [Image Processing and Computer Vision]: Enhancement—Geometric correction; I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms

Keywords: Focal stack, compositing, depth of field, halo correction, geometric optics

*e-mail: {dejacobs, jbaek, levoy}@cs.stanford.edu

1 Introduction

Depth of field is one of the principal artistic tools available to photographers. Decisions about which scene elements are imaged sharply and which are out of focus direct a viewer's attention and affect the mood of the photograph. For traditional cameras, such decisions are made by controlling the lens' aperture and focus distance. Unfortunately, many consumer cameras—including mobile phone cameras and compact point-and-shoot cameras—have limited or no control over the aperture because of constraints imposed by portability and expense. However, nearly all cameras have focus controls and are capable of capturing a stack of images focused at different distances. This set of images is called a *focal stack*. As we will demostrate in this paper, these images can be combined to simulate depth of field effects beyond the range normally allowable by the camera's optics, including depth of field reduction, extension, and even freeform non-physical effects. Figure 1 shows examples of two of these manipulations.

In focal stack compositing, each pixel of the output is a weighted sum of corresponding pixels in the input images—often referred to as focal stack "slices." The choice of pixel weights determines the depth of field of the composite. Given a focal stack and user-specified novel camera parameters, appropriate blending weights can be computed via a compositing pipeline—ours is illustrated in Figure 2. The first step in compositing is to generate or otherwise

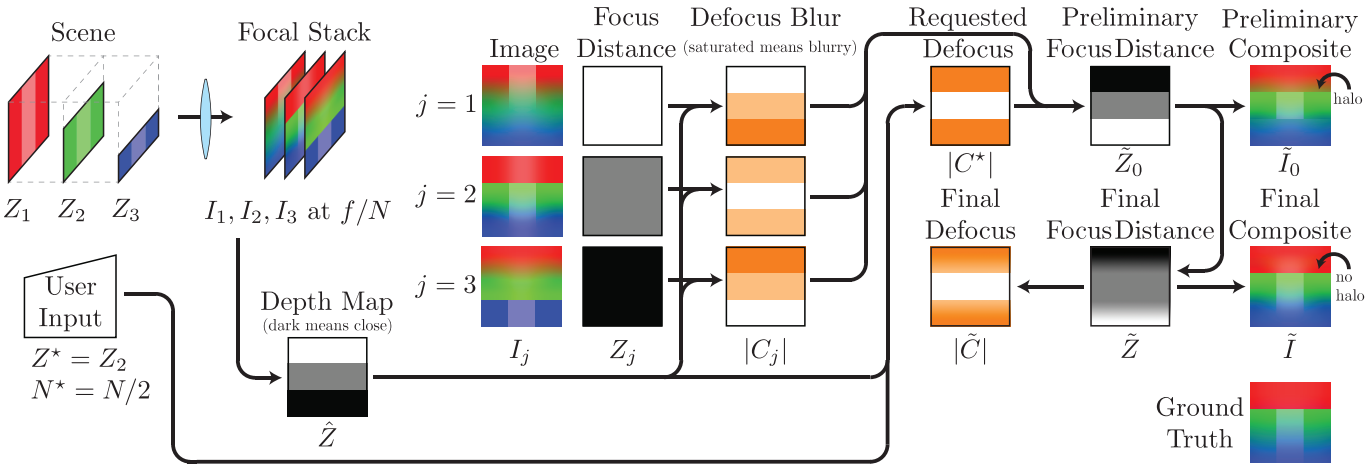


Figure 2: *Our compositing pipeline. Given the scene pictured in the upper left, we capture a stack of images $\{I_1, I_2, I_3\}$ focused at depths $\{Z_1, Z_2, Z_3\}$ with f-number N. This set of images is called a focal stack. We feed these images into a depth extraction algorithm (ours is described in Section 4) to generate an estimate for the distance $\hat{Z}$ between the lens and the object imaged by each pixel. For scene depth and focus distance maps (all images labeled Z) in the diagram above, we use intensity to represent depth; white means background, and black means foreground. Given the scene depth map $\hat{Z}$, we calculate per-pixel the signed defocus blur C_j corresponding to each slice I_j, using Equation* (1)*. Above, we visualize the degree of defocus blur (in images labeled $|C|$) using saturation, where orange means blurry and white means sharp. Equation* (1) *also allows us to compute a requested defocus map $C^\star$ given a user-specified focus distance $Z^\star$ and f-number $N^\star$. Some photographic examples of $C^\star$ are shown in Figure 1. In the example above, the user is requesting a reduced depth of field composite focused at $Z^\star = Z_2$ with f-number $N^\star = N/2$. We then compute a preliminary focus distance map $\tilde{Z}_0$ that specifies the depth at which each pixel should be focused in order to achieve the requested defocus $C^\star$. For example, in order to maximally defocus the distant red object at Z_1 visible in the top third of the composite, the preliminary focus distance calls for those pixels to be drawn from I_3, which is focused close to the camera. Indexing into our focal stack as described creates a preliminary composite $\tilde{I}_0$ that is inexpensive to compute, but contains halo artifacts visible near depth edges. To prevent such artifacts, we apply geometric constraints (discussed in Section 3.4) on $\tilde{Z}_0$ to create a smoother focus distance map $\tilde{Z}$. The resulting composite $\tilde{I}$ is locally artifact-free, but its corresponding defocus map $\tilde{C}$ does not match $C^\star$ perfectly. Finally, in the bottom right we show a ground truth image for a camera with the requested parameters $Z^\star, N^\star$.*

acquire a proxy for the scene geometry—in our case, we use a depth map. Some knowledge of the scene geometry is necessary in order to estimate the per-pixel defocus blur present in each slice of the focal stack. Additionally, scene geometry is required to calculate the per-pixel defocus appropriate for the synthetic image taken with a user's requested hypothetical camera. A basic focal stack composite, then, is given by selecting or interpolating between the slices that match the requested defocus as closely as possible at each pixel.

Certainly, one may produce similar effects without a focal stack—using only a single photograph. First, one can reduce the depth of field by segmenting the image into layers and convolving each layer with a blur kernel of the appropriate size. In practice, however, synthetic blur fails to capture subtle details that are naturally present in photographic (physically produced) blur. In particular, saturated image regions cannot be blurred synthetically because their true brightness is unknown. Similarly, scene inter-reflections and translucencies can cause a single pixel to have multiple depths; therefore, no single convolutional kernel will be correct. Photographic blur, by contrast, guarantees a consistent defocus blur regardless of depth map accuracy. In addition, physical optical effects like contrast inversion [Goodman 1996] cannot be correctly modeled by synthetic blur, but are present in photographic blur. Second, one can extend depth of field without a focal stack via deconvolution, but this process is ill-posed without significantly modifying camera optics or assuming strong priors about the scene.

Finally, the requirement to capture a focal stack is not as onerous as it would seem. Cameras that employ contrast-based autofocusing [Bell 1992] already capture most, if not all, of the required imagery, as they sweep the lens through a range. Contrast-based autofocusing is employed by nearly all cameras with electronic viewfinders. The only additional cost is the bandwidth required to save the autofocus ramp frames to disk. Additionally, the depth map required for our compositing algorithm is easily computed as a byproduct of capturing a focal stack.

We present a theory, framework and pipeline for focal stack compositing that produce composites matching a requested depth of field. This pipeline is shown in Figure 2, and is described throughout Section 3. We will analyze the geometry of such composites, discuss how halo artifacts (especially at occlusion edges) can arise, and show how the halo artifacts can be mathematically avoided by minimal alteration of the desired depth of field. We will then demonstrate the versatility of this framework in applications for reducing depth of field, extending depth of field, and creating free-form non-physical composites that are halo-free.

2 Prior Work

Depth of field is a useful visual cue for conveying the scene geometry and directing the viewer's attention. As such, it has been well-studied in the rendering literature. When raytracing a synthetic scene, one can obtain the desired depth of field by simulating the appropriate lens optics and aperture [Cook et al. 1984; Kolb et al. 1995] or by employing other image-space postprocessing [Barsky and Pasztor 2004; Kosloff and Barsky 2007] that nevertheless relies on access to the scene model. In traditional photography, however, the photographer determines the depth of field via his choice of the relevant camera parameters. While modifying the camera can partially increase the range of possible depth of field [Mohan et al. 2009] or the bokeh shape [Lanman et al. 2008], the depth of field is essentially fixed at the capture time, barring post-processing.

Overcoming this limitation requires correctly estimating the amount of blur present at each pixel, and then simulating the desired blur (if different), which may be greater or smaller than the pre-existing blur at the pixel location. For instance, defocus magnification [Bae and Durand 2007] and variable-aperture photography [Hasinoff and Kutulakos 2007] increase the per-pixel blur using image-space convolution, thereby simulating a narrower depth of field. Reducing the per-pixel blur, on the other hand, requires deblurring the image, and can be ill-posed for traditional lens bokehs [Levin et al. 2007].

There exists a large body of work in computational optics that combats the numerical instability of deblurring a traditional photograph by capturing a coded 2D image. Many of them employ spatiotemporal coding of the aperture, in order to increase the invertibility of the defocus blur [Levin et al. 2007; Zhou and Nayar 2009], and a large subset thereof is concerned with equalizing the defocus blur across depth, thereby avoiding errors introduced from inaccurate depth estimation [Dowski and Cathey 1995; Nagahara et al. 2008; Levin et al. 2009]. While the field of deconvolution has advanced significantly, deconvolved images tend to have a characteristically flat texture and ringing artifacts.

One alternative to capturing a coded 2D image is acquiring a redundant representation of the scene composed of many photographs. Light fields [Levoy and Hanrahan 1996; Ng 2005] and focal stacks [Streibl 1985] are composed of multiple images that are either seen through different portions of the aperture, or focused at varying depths, respectively. Light fields can be rendered into an image by synthetic aperture focusing [Isaksen et al. 2000; Vaish et al. 2005]. Prior works in focal stack compositing [Agarwala et al. 2004; Hasinoff et al. 2008] simulate a hypothetical camera's depth of field by extracting from each slice the regions matching the proper level of blur for a given aperture size and focus distance. However, while focal stack compositing is a rather well-known technique demonstrated to be light efficient, it is yet to be analyzed with respect to the geometric implications of a particular composite. Specifically, the proper spatial relationships between composited pixels necessary for artifact prevention are not yet well-studied. As a result, composites produced by these techniques frequently suffer from visual artifacts.

3 Theory

We now build a theory of focal stack compositing as a tool for manipulating depth of field, following the pipeline depicted in Figure 2. Our theory differs from prior work in two key ways: 1) it is fully general and allows non-physical, artistically driven depth-of-field effects, and 2) it explicitly models the interplay between the lens optics and scene geometry in order to remediate visual artifacts at depth edges.

We begin by assuming a thin-lens model and the paraxial approximation. For now, let us also assume the existence of a depth map $\hat{Z}(\vec{p})$, where $\vec{p} = (p_x, p_y)$ is a pixel's location on the sensor relative to the optical axis. $\hat{Z}(\vec{p})$ is defined as the axial distance between the lens and the object hit by the chief ray passing through $\vec{p}$ when the sensor is focused at infinity. We discuss depth map extraction in Section 4.

Although the pipeline shown in Figure 2 is written in terms of object-space depths Z, it is algebraically simpler to express our theory in terms of the conjugate sensor-space distances S. This simplification is a consequence of the 3D perspective transform applied by the lens as the scene is imaged. Figure 3 illustrates this transform. Many geometric relationships in object space become arithmetic in sensor space, and thus are less unwieldy to discuss.

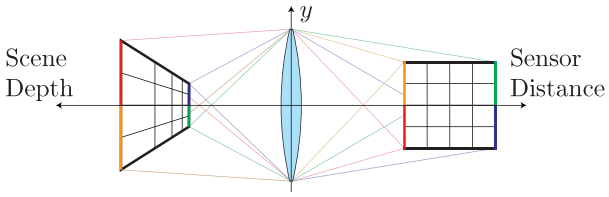


Figure 3: *A lens performs a perspective transformation. A linear change in sensor position (on the right) corresponds to a non-linear change in object distance (on the left.) Coupled with this change in depth is lateral magnification (up and down in the figure.) Although it is conceptually easier to reason about geometric relationships in object space, the non-linearity it introduces makes the algebra unwieldy. Therefore, we primarily work in sensor space for the remainder of the paper.*

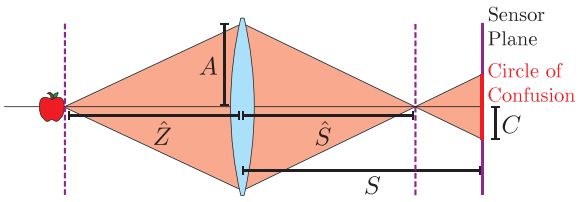


Figure 4: *Defocus blur. The rays emitted by the object at depth $\hat{Z}$ converge at a distance $\hat{S}$ behind the lens. A sensor placed at a distance S, instead of the correct distance $\hat{S}$, will not sharply image the object. The rays do not converge on the sensor but rather create a blur spot of radius C. The aperture radius A determines the rate at which C grows as the sensor moves away from $\hat{S}$.*

The Gaussian lens formula tells us the relationship between a scene point and its conjugate. If we apply this to the scene's depth map, $\hat{Z}(\vec{p})$, we define a map $\hat{S}(\vec{p}) = (1/f - 1/\hat{Z}(\vec{p}))^{-1}$, where f is the focal length of the lens. $\hat{S}(\vec{p})$ is constructed such that the pixel at $\vec{p}$ will be in sharp focus if the sensor is placed at a distance $\hat{S}(\vec{p})$.

Working in this conjugate sensor space, we define our framework as follows: given a set of images $\{I_j\}_j$ taken at sensor positions $\{S_j\}_j$ with f-number N, we want to calculate a composite $\tilde{I}$ that approximates a hypothetical camera with a sensor placed at $S^\star$ and f-number $N^\star$. Later we will relax this constraint and allow a hypothetical camera that is non-physical.

3.1 Defocus Blur

Defocus blur is a consequence of geometric optics. As shown in Figure 4, if the sensor is placed at a distance S from the lens, a blur spot of radius C forms on the sensor. This blur spot is known as the circle of confusion, and its shape is referred to as the lens' bokeh. Using similar triangles, one can show that $C = A(1-S/\hat{S})$. Rewriting the the aperture radius A in terms of the focal length f and the f-number N, we obtain,

$$C = \frac{f}{2N}(1 - S/\hat{S}). \tag{1}$$

Note that C, as defined in Equation (1), is a signed quantity. If $C > 0$, then the camera is focused behind the object. If $C < 0$, then the camera is focused in front of the object, and the bokeh will be inverted. For most lenses, the bokeh is approximately symmetric, so it would be difficult for a human to distinguish between defocus blurs of C and $-C$. Despite this perceptual equivalence,

we choose the above definition of C, rather than its absolute value, because it maintains a monotonic relationship between scene depth and defocus blur when compositing.

Defocus Maps

Depth of field is typically defined as the range of depths within which objects are imaged sharply. Implicit in this definition is, first, that sharpness is dependent solely upon the depth, and second, the range is a single contiguous interval, outside of which objects will be blurry. However, often the desired blurriness is dependent also on objects' locations in the frame (e.g. tilt-shift photography, or other spatially varying depth-of-field effects). Such non-standard criteria can be captured by a more general representation, namely a map of desired circle-of-confusion radii across the sensor. We call this function $C(\vec{p})$ a *defocus map*.

A defocus map encapsulates the goal of focal stack compositing. For example, if we wish to simulate a camera with the sensor placed at a distance $S^\star$ with f-number $N^\star$, the desired defocus map can be calculated from $\hat{S}(\vec{p})$ as

$$C^\star(\vec{p}) = \frac{f}{2N^\star}(1 - S^\star/\hat{S}(\vec{p})). \tag{2}$$

An all-focused image is trivially specified by $C^\star(\vec{p}) = 0$. One can specify arbitrary spatially-varying focus effects by manually painting $C^\star$, using our stroke-based interface presented in Section 5.3.

3.2 Sensor Distance Maps

Given $C^\star$ as a goal, our task now is to find a composite that corresponds to a defocus map as close to $C^\star$ as possible. We will represent our solution as a function that defines the desired sensor position for each pixel. We call such a function a *sensor distance map*. This is a convenient choice as a proxy for focal stack indices because it has physical meaning and is independent of the depth resolution of our focal stack. It also lends itself well to an alternative interpretation of focal stack compositing as the construction of a sensor surface that is conjugate to a (potentially non-planar) surface of focus in object space.

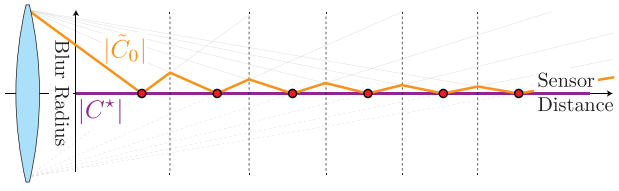
If our only concern is matching $C^\star$ as closely as possible, finding the optimal sensor distance map is straightforward. For any given (unsigned) defocus blur radius, two sensor positions will achieve the desired blur—one focused in front of the scene object and one focused behind. Because we defined the defocus blur radius $C^\star$ to be a signed quantity, however, there is no ambiguity. Accordingly, we can find the sensor distance map $\tilde{S}_0(\vec{p})$ for a preliminary composite by inverting the relationship given by Equation (1):

$$\tilde{S}_0(\vec{p}) = \hat{S}(\vec{p})\left(1 - \frac{2NC^\star(\vec{p})}{f}\right). \tag{3}$$

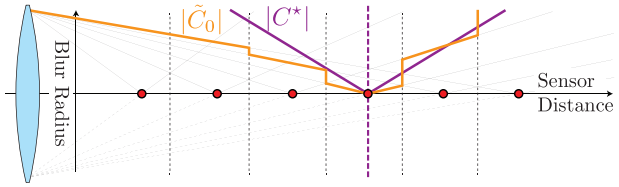
An all-focus image is trivially specified by $\tilde{S}_0(\vec{p}) = \hat{S}(\vec{p})$. We call $\tilde{S}_0(\vec{p})$ a *preliminary* sensor distance map because, as we will show in Section 3.4, it may not produce a visually pleasing composite.

3.3 Compositing

In order to build a *preliminary* composite $\tilde{I}_0$, we must determine which pixels from the focal stack best approximate the desired sensor distance $\tilde{S}_0$. A simple choice would be to quantize $\tilde{S}_0$ to the nearest sensor positions available in the focal stack and assign pixels the colors from those slices. The resulting defocus blur $\tilde{C}_0$ for such a composite will approximate $C^\star$. Figure 5 shows a comparison between $\tilde{C}_0$ and $C^\star$ for two depth-of-field manipulations.



(a) All-focus composite



(b) Composite simulating a wide aperture

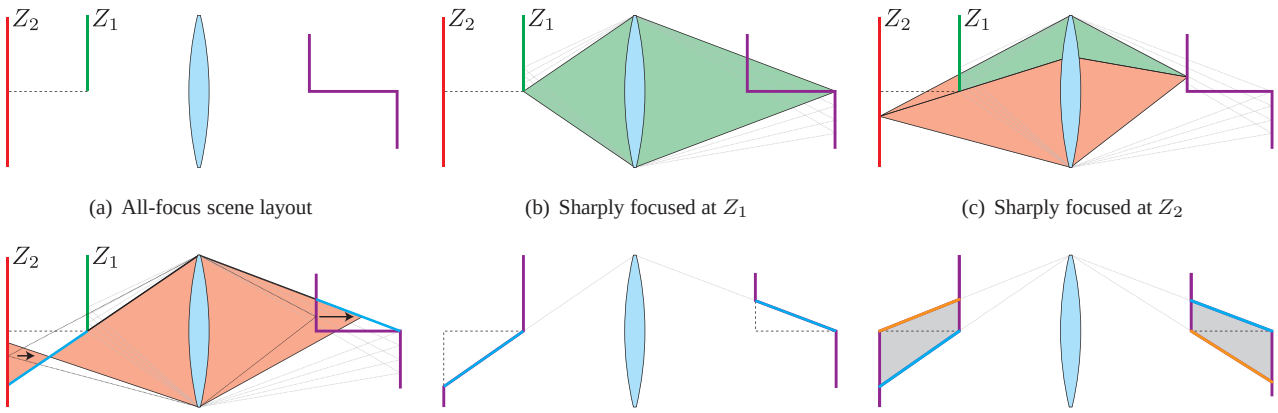
Figure 5: *Depth-defocus relationships for focal stack composites. (a) An all-focus composite is characterized by $|C^\star| = 0$ (shown in purple.) If our stack has slices at sensor distances corresponding to the red circles, then the composite will assign each pixel the color of the nearest stack slice in sensor distance (as segmented by the dotted vertical lines), thus creating the depth-defocus relationship given by $|\tilde{C}_0|$ (shown in orange.) $|\tilde{C}_0|$ is farthest from $|C^\star|$ midway between stack slices, so as one might suspect, adding more stack slices will improve the all-focus composite. (b) A simulated wide aperture composite is characterized by a $|C^\star|$ that grows quickly as it moves away from its conjugate plane of focus (dashed purple line.) This can be approximated by "flipping" sensor positions about the conjugate plane of focus, such that an object nearby is assigned the color of a slice focused far away and vice versa.*

However, quantizing $\tilde{S}_0$ as described above can create discontinuities in the defocus map $\tilde{C}_0$ as seen in Figure 5(b). These discontinuities manifest themselves as false edges in a composite when transitioning between stack slices. We can smooth these transitions by linearly interpolating between the two closest stack slices as an approximation for $\tilde{S}_0$ instead of quantizing. This provides a good approximation in most cases. Interpolation should not be used when $C^\star$ calls for a pixel to be sharper than both of its nearest stack slices (i.e. the scene object's focused sensor distance $\hat{S}(\vec{p})$ is between the two nearest stack slice positions.) In this circumstance, blending the two slices will only increase the defocus blur at $\vec{p}$, so it is best to just choose the closer single slice—this case is shown in Figure 5(a).

3.4 Eliminating Color Halos

Assigning each pixel a sensor distance independently of its neighbors will create a composite whose per-pixel blur $\tilde{C}_0$ matches $C^\star$ as closely as possible. However, our goal is not just to obtain blur that matches $C^\star$, but to do so without producing visual artifacts. Halo artifacts, which we define as color bleeding across depth discontinuities, are common in preliminary composites and are visually objectionable, as demonstrated in Figure 6. Therefore, we will compute a *final* sensor distance map $\tilde{S}(\vec{p})$ that generates a halo-free composite whose per-pixel blur is close to $C^\star$.

Halos, we claim, are the manifestation of the "double-counting" of rays, i.e. more than one pixel in the composite integrating a given ray. Consider a ray from an object sharply imaged in one pixel. If captured again by another pixel, it will necessarily appear as a defocused contribution of the same object. Figure 7 illustrates this geometry. The result is the characteristic color bleeding of halos.



(a) All-focus scene layout (b) Sharply focused at Z_1 (c) Sharply focused at Z_2

(d) Adjusting focus to prevent halo (e) A halo-free surface of focus (f) Alternative halo-free surfaces of focus

Figure 7: *Halo geometry. (a) An example scene with a green and red object at depths Z_1 and Z_2, respectively. For an all-focus composite, the requested surface of focus coincides with Z_1, then jumps to Z_2 at the occlusion boundary. This surface of focus corresponds to the sensor surface shown in purple on the right side of the lens. (b) All light emitted from a point on the foreground object at Z_1 (green shaded area) converges properly at the sensor. (c) Near the occlusion boundary, only a fraction of the light emitted from the background object at Z_2 (red shaded area) reaches the sensor. The light that is blocked is replaced with a defocused contribution from the foreground object (green shaded area.) This contribution appears visually as a green haze over the red background object—similar in appearance to that seen in $\tilde{I}_0$ in Figure 2. This haze, next to the otherwise sharp silhouette of the foreground object, is a halo artifact. (d) The closest halo-free alternative is to focus on the line passing through the edge of the foreground object and the corner of the lens aperture. Any bundle of rays leaving a point on this line will not be occluded by the foreground object. The blue portion of this line gives a halo-free transition in the surface of focus between Z_1 and Z_2. The corresponding sensor surface transition is drawn in blue to the right of the lens. (e) A halo-free surface of focus and its corresponding sensor surface. (f) Alternative halo-free surfaces and their conjugates can be found by choosing to defocus the foreground instead of the background (the orange transition connecting the two focus distances) or some combination of both (the grey shaded regions). The best transition choice is application-specific.*

(a) Haloed composite (b) Ground truth photograph

Figure 6: *Halo artifacts. (a) A preliminary composite with a halo artifact (inset). (b) A long-exposure ground truth $f/22$ image.*

Intuitively, observing a single ray twice in a composite should be avoided, because it is physically impossible. Real photographs of opaque objects never contain halos: once a ray has been captured by a pixel, it cannot be detected by another, even with an exotic, non-planar sensor surface (which we simulate with our composites.) Focal stack composites are not constrained in this manner, because pixels at different sensor distances are not necessarily captured simultaneously, leaving open the possibility of double-counting rays.

Geometrically, the double-counting of rays by two distinct pixels is equivalent to the two pixels being collinear with a point on the aperture. In order to detect this condition, we need to examine each pair of pixels, extend a line through them, and test whether this line intersects the aperture. If it does, then that pair of pixels will constitute a halo. Algebraically, this test is equivalent to asking whether the gradient of $\tilde{S}$ is bounded by some maximum rate of change. For example, for a pixel $\vec{p}$ located on the optical axis, $\tilde{S}(\vec{p})$ should have its gradient bound as follows,

$$\|\nabla \tilde{S}(\vec{p})\| \leq \frac{\tilde{S}(\vec{p})}{A}. \tag{4}$$

where A is the aperture radius. Under the paraxial approximation, Equation (4) applies to all other pixels $\vec{p}$. Therefore, acceptable sensor surfaces are those that satisfy Equation (4).

Now that we know how to mathematically characterize halo-inducing sensor configurations, we may construct a corrected sensor distance map $\tilde{S}$ that avoids them, by algorithmically enforcing the constraints. Note that naïvely checking the slope of $\tilde{S}$ between every pair of pixels will yield an algorithm whose runtime is quadratic in the number of pixels. Instead, we observe that for each $\vec{p}$, it is sufficient to check the slope between $\vec{p}$ and its closest neighbor $\vec{q}$ whose sensor distance is s, for each possible value of s. This holds because the constraints arising from checking all other pixels are necessarily weaker than those we check. This optimization reduces the time complexity of the algorithm to be linear in the number of pixels, at the cost of introducing a linear dependence on the depth resolution. However, the set of values occurring in the sensor distance map is typically small—on the order of the number of slices in the focal stack. Algorithm 1 summarizes the implementation of this optimization.

Algorithm 1 iterates over the set of sensor distance values. Each iteration, corresponding to a particular sensor distance s, enforces all pairwise constraints that involve any pixel whose sensor distance is s (the set of such pixels is denoted by Q_0.) More precisely, we identify pixels that interact with Q_0, ordered by increasing distance from Q_0, by iteratively dilating Q_0. We then adjust their sensor distances if their interaction with Q_0 violates Equation (4).

Algorithm 1 Constructing $\tilde{S}$.

$\tilde{S} \leftarrow \tilde{S}_0$.
DilationLimit $\leftarrow$ length of image diagonal
for all sensor distances s **do**
  Let Q_0 be the set of all pixels $\vec{q}$ such that $\tilde{S}(\vec{q}) = s$.
  for $r = 1$ **to** DilationLimit **do**
    Let $Q_r = \text{dilate}(Q_{r-1}, 1 \text{ pixel})$.
    Let ∂Q be the set of newly included pixels in Q_r.
    Let $(S_{min}, S_{max}) = (s - \frac{s}{A}r, s + \frac{s}{A}r)$.
    for all pixels $\vec{p}$ in ∂Q **do**
      Clamp $\tilde{S}(\vec{p})$ to be in $[S_{min}, S_{max}]$.
    end for
  end for
end for

The algorithm presented above generates a family of halo-free sensor distance maps. Figure 8 shows composites from one such family. The specific sensor distance map produced depends on the order in which the sensor distances are considered in the outer loop of the algorithm. Because each iteration resolves any conflict involving pixels at a specific sensor distance s, those pixels at distance s will be unaffected by future iterations. As such, more important sensor distances should be prioritized if possible. It is difficult to determine the best order a priori. In fact, it will vary based on the user's intentions. Our implementation uses a foreground-favored ordering by default, and would produce the composite shown in Figure 8(a).

The theory presented above relies heavily on knowing where the edge of the aperture is, and hence on the thin-lens model and the paraxial approximation. Real photographic lenses, on the other hand, are complex systems of multiple elements, and as such, may deviate from our assumptions. If we had knowledge of the exact optical parameters for a lens, we could perform a similar analysis to more accurately model the spatial extent of halos. In practice, without such knowledge, we conservatively over-estimate the size of halo effects to be twice the amount the theory would imply.

3.5 Reducing Blur with a Variable Aperture

The halo elimination algorithm just described removes color bleeding by sacrificing some accuracy in matching $C^\star$. For extended depth of field composites, this manifests as blurriness near depth edges. If the camera is equipped with a controllable aperture, we can further improve on our composite by operating on a *focus-aperture block*, rather than a focal stack. A focus-aperture block is a 2D family of photographs with varying focus as well as varying aperture radius.

Note that being able to capture narrow-aperture photographs does not necessarily obviate the work needed to generate an all-focus image, for two reasons: 1) the narrowest aperture may not be small enough, and 2) images taken with a small aperture are noisy, assuming a constant exposure duration. Wherever the depth map is flat, a properly focused wide-aperture photograph should be just as sharp as its narrow-aperture counterpart, and less noisy. However, near depth discontinuities, we can trade off noise against blurriness by selecting the appropriate f-number for each pixel.

Recall that the halo-free constraint is given by Equation (4). From this equation, we note that we may tune either $\tilde{S}(\vec{p})$ or A (or both) to satisfy it. Figure 9 visualizes how tuning the aperture can help satisfy the equation. Therefore, in order to create a maximally focused, low-noise, and halo-free composite, we should reduce the aperture near occlusion boundaries.

(a) Foreground-favored composite (b) Background-favored composite

Figure 8: *Alternative halo-free composites. Recall from Figure 7(f) that there exists a family of halo-free surfaces of focus that well approximate a given preliminary surface of focus. The specific halo-free surface of focus generated is determined by the order in which sensor distances are processed in the outer loop of Algorithm 1. (a) A composite that prioritizes foreground objects, produced by processing sensor distances in decreasing order. This corresponds to the blue depth transition shown in Figure 7(f). (b) A composite that prioritizes background objects, produced by processing sensor distances in increasing order. This corresponds to the orange depth transition shown in Figure 7(f).*

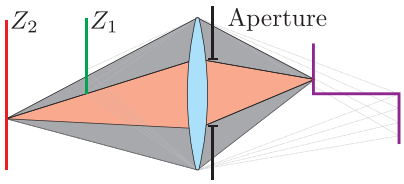
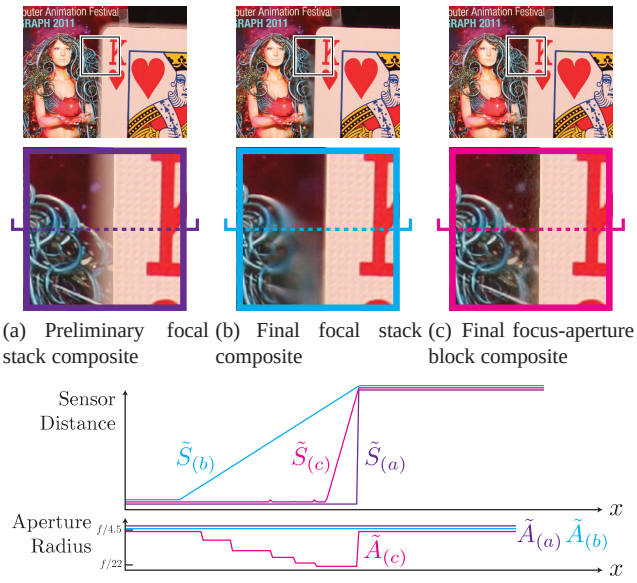


Figure 9: *Adjusting aperture to prevent halo. If we use the full aperture of the lens, the sensor will integrate all the light shown in the shaded regions, including a halo-causing contribution from the foreground object. While this can be addressed by adjusting focus as in Figure 7(d), stopping down the aperture as shown above reduces the light reaching the sensor to just the red shaded region, effectively blocking the contribution from the foreground object. In general, we can eliminate halos without introducing any blur, by reducing aperture near occlusion boundaries.*

To handle focus-aperture blocks, we must slightly modify Algorithm 1. Specifically, we are to find not only a halo-free sensor distance map $\tilde{S}(\vec{p})$, but also a spatially varying aperture radius map $\tilde{A}(\vec{p})$ that accompanies it. We solve this problem using a two-pass approach. In the first pass, we initialize $\tilde{A}(\vec{p})$ to be the largest aperture radius available, and execute Algorithm 1. However, instead of clamping the sensor distance whenever a halo is encountered, we narrow the aperture at the affected pixel location by the appropriate amount to satisfy Equation (4). It may be that the narrowest available aperture is still too large, in which case we settle for this value. In the second pass, we execute the algorithm in its original form, clamping the sensor distance according to the local constraint based on the spatially varying aperture map computed in the previous pass. Figure 10 visualizes the effect of the augmented algorithm on $\tilde{S}$, $\tilde{A}$ and shows the resulting composites.

4 Methodology

To test the validity of our theory of focal stack compositing, we captured several datasets with representatives from two common classes of camera: a Canon 7D, representing DSLRs, and a Nokia N900, representing mobile phone cameras. They were programmed with the Canon EDSDK and Frankencamera API [Adams et al. 2010], respectively, to capture a focal stack of 32 slices



(a) Preliminary focal stack composite (b) Final focal stack composite (c) Final focus-aperture block composite

(d) Sensor distance and aperture radius along a scanline

Figure 10: *Extended depth of field with a focus-aperture block. A focus-aperture block enables trading off blurriness for noise. (a) and (b) show the synthesized all-focus images from a focal stack. As expected, the latter is halo-free, but is blurry along the depth edge. (c) shows the result of applying halo correction on a focus-aperture block, and is free of blur at the expense of increased noise. (d) plots the sensor distance map along the scanlines visualized in the insets of (a),(b),(c), respectively labeled as $\tilde{S}_{(a)}, \tilde{S}_{(b)}, \tilde{S}_{(c)}$, as well as the aperture radius map $\tilde{A}_{(a)}, \tilde{A}_{(b)}, \tilde{A}_{(c)}$. Note that small aperture radii are used just before the depth edge, to minimize blur, but when possible, a larger aperture radius is used.*

(at 5184×3456 and 2592×1968 resolution, respectively), spread evenly in diopter space through the range of realizable focus distances (40cm-∞ for the 28mm f/4.5 lens used with the Canon 7D; 5cm-∞ for the 5.2mm f/2.8 lens affixed to the Nokia N900.) For some datasets we programmed the Canon 7D to capture a focus-aperture block with aperture values f/4.5 to f/22 at each of the focus positions, holding exposure duration constant. Capturing a full focal stack takes approximately 2 minutes for both platforms (or 10 minutes for a focus-aperture block). For the Canon 7D, most of the capture time is spent waiting for camera parameter changes to take effect; for the Nokia N900, writing to disk dominates the process. We revisit the issue of acquisition speed in Section 6.

Once a focal stack has been captured, we must ensure the slices therein represent aligned views of the scene. Although we assume a static scene, the field of view will not be constant; as a camera changes focus, its field of view expands or contracts slightly because image magnification changes with sensor distance, as shown in Figure 3. We measure the magnification for each camera-lens pair by capturing a focal stack of a highly textured test scene, and matching image features across every consecutive pair of slices in the stack. Once this calibration test is run, the ratios among the scale factor of all slices can be computed, and this set of scale factors can be applied on subsequent focal stacks in order to offset the change in field of view.

Once the slices are aligned, we extract a depth map from the focal stack by a simple contrast detection algorithm: we compute a contrast measure for each slice of the stack, and then select the depth that maximizes contrast at each pixel. Because this algorithm is not always robust, the depth maps for some datasets that appear later in this paper were manually corrected as necessary. Other, more sophisticated depth extraction techniques like depth from defocus [Subbarao and Surya 1994] could obviate this step. It should be noted, however, that in comparisons against prior work, we provide each algorithm the same depth map to ensure fair comparison.

5 Applications

Given the pipeline, illustrated in Figure 2, for processing a focal stack and defocus map into a composite, we now describe several applications and compare our results to [Hasinoff et al. 2008] where possible. The first two applications either extend or reduce the depth of field, while maintaining a physically plausible camera model. The last application demonstrates focus effects that cannot be realized by traditional optics. The choice of application determines how $C^\star$ should be created.

5.1 Reduced Depth of Field

Mobile phone cameras often have small, fixed apertures and, consequently, a wide and inflexible depth of field. Despite these limitations, we can create an SLR-like depth of field by choosing a defocus map that would accentuate out-of-focus blur. As an example, suppose that we have a focal stack captured at $f/2.8$, for a scene that contains a subject that is sharply imaged at the sensor distance S. Once the depth map is obtained, we can predict the defocus map for a hypothetical $f/0.7$ photograph focused on the subject, using Equation (2). Letting $C^\star$ equal this defocus map in our compositing pipeline, we obtain a composite corresponding to the hypothetical camera. Figure 11 shows this hypothetical case and another similar example.

The sensor distance map defined by such composites are generally "depth flipped" copies of the all-focus sensor distance $\hat{S}$ about S. Namely, prior to halo correction, we have,

$$\tilde{S}_0(\vec{p}) = S + k(S - \hat{S}(\vec{p})),$$

where k is a constant reflecting the aggressiveness of the depth of field reduction. In other words, to defocus objects that are behind the subject, the surface of focus should be in front of the subject, and vice versa. In order to make best use of the data available in focal stack, k should be close to 1. This roughly corresponds to a two f-stop increase in aperture radius. In practice, more ambitious composites ($k \gg 1$) will degrade in quality because the defocus blur $\tilde{C}_0$ will have discontinuities at depth layer boundaries—discussed earlier in Section 3.3—too large to be masked by interpolation.

5.2 Extended Depth of Field

The process of creating an extended depth of field image within our framework is analogous to that in the previous application. The desired defocus map is again dictated by Equation (2), but now $N^\star$ is larger than what is available in the input dataset. In the case of an all-focus image, $N^\star = \infty$, and the desired defocus map is zero everywhere: $C^\star(\vec{p}) = 0$. Consequently, $\tilde{S}_0(\vec{p}) = \hat{S}(\vec{p})$. Figure 12 shows all-focus images generated from focal stacks.

5.3 Freeform Depth of Field

One advantage of our pipeline is its capability to accept an arbitrary defocus map $C^\star$ as an input, rather than a target focus depth and f-number, and create a composite with a freeform depth of field. As such, a user-provided defocus map, even one that is physically impossible with traditional optics, can act as an input. That

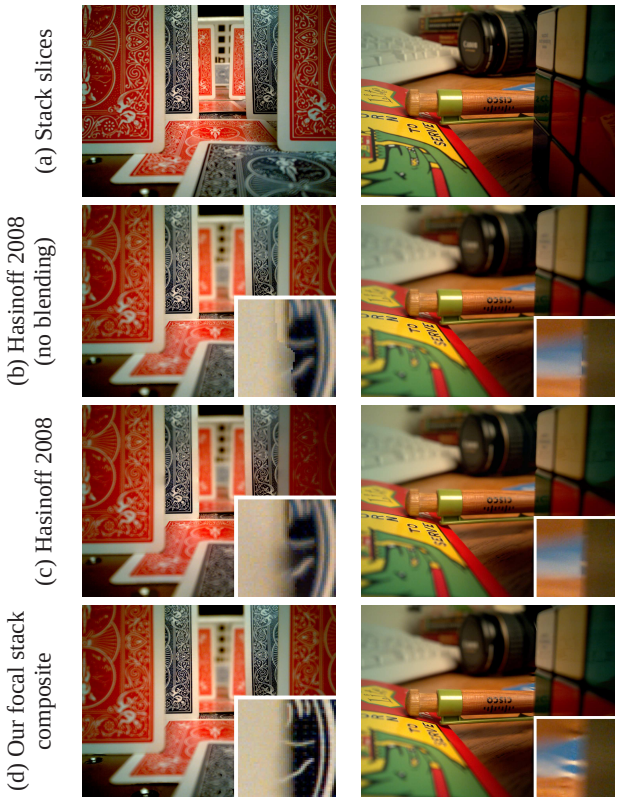


Figure 11: *Reduced depth of field. Both scenes were captured with a Nokia N900 mobile phone with a fixed $f/2.8$ lens. All composites simulate an $f/0.7$ lens focused at the same distance. (a) Examples of individual focal stack slices. (b,c) Composites generated by Hasinoff et al.'s algorithm without and with gradient domain blending, respectively. Note that the blending softens the harsh depth transition, but also has the unintended consequence of altering the global tone. (d) Composites generated by our proposed algorithm.*

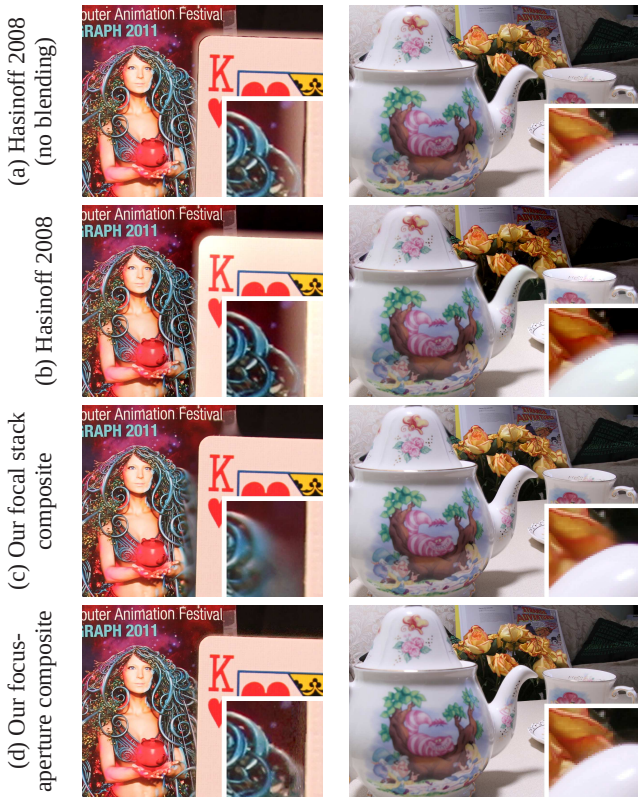


Figure 12: *Extended depth of field. Both scenes were captured with a Canon 7D with apertures $f/4.5$ to $f/22$. (a,b) Composites generated by Hasinoff et al.'s algorithm without and with gradient domain blending, respectively, using only the $f/4.5$ images. Note that the blending reduces the visibility of the halo, but requires changing the global tone of the photograph to do so. (c,d) Composites generated by our proposed algorithm using only the $f/4.5$ images and using the entire focus-aperture block, respectively.*

said, the realism of a composite is tightly coupled with a consistent relationship between $C^\star$ and $\hat{S}$. Accordingly, creating a defocus map by hand is a daunting proposition without the right tools. In this section we will describe a stroke-based interface that allows a user to specify a physically realizable depth of field and localized non-physical deviations from it.

Specifying Freeform Depth of Field

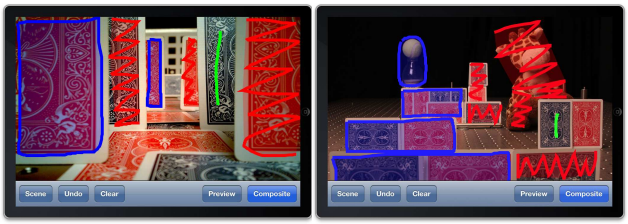
We provide a two-phase interface for specifying a freeform depth of field. In the first phase, the user specifies the range of depths that should be in sharp focus, and obtains a pilot composite that exhibits the depth of field similar to that of a conventional camera. In the second phase, the user iteratively applies localized edits to regions that should be either blurrier or sharper than they appear in the pilot image. When finished, the user obtains a final composite that may not correspond to any camera with traditional optics.

In order to specify the range of depths in the first phase, the user strokes objects in a pilot image, which is initialized to be a single stack slice. The depths of the objects stroked dictate the range that should be sharply imaged. Although it is possible to express the same preference using simple sliders for the equivalent focus distance and f-number, we find that our approach inspired by direct manipulation [Shneiderman 1983] is easier to work with. We can quickly compute a composite satisfying the desired depth of field by removing slices focused outside the desired range and constructing an all-focus image with the remainder. The result is presented to the user, who may select a new pilot image or elect to specify further edits in the second phase. We remark that the interaction in the first phase effectively emulates the A-DEP mode on old Canon cameras—which selects the camera's aperture and focus to match a photographer's desired depth of field.

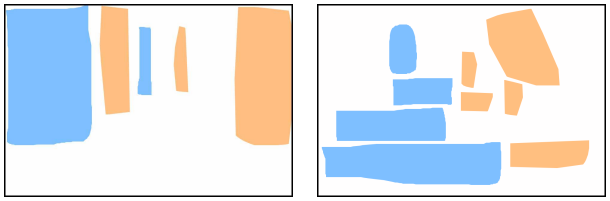
In the second phase, the user indicates image regions that should have their defocus modified, with stroke-based area gestures drawn on the pilot image. *Focus areas* are indicated by enclosing a portion of the pilot image in a self-intersecting loop, defining a region that should be sharper; *defocus areas* are indicated by scratching out a portion of the pilot image, defining a region that should be blurrier. Figure 13(b) show examples of these area strokes. After each markup, the user may request a composite satisfying the specified constraints. If not satisfied, the user can continue to supply edits on the returned composite.

In order to satisfy the user's request, we must first convert the user's sparse stroke-based input into a dense defocus map. To do so, we start from the defocus map of the current pilot image $C_0^\star(\vec{p})$, which is already densely defined. We define a markup map $D(\vec{p})$ whose sign corresponds to the user's intention at a given pixel location: $D(\vec{p}) < 0$ denotes sharpening; $D(\vec{p}) > 0$ denotes blur-

(a) Pilot images specified by the user



(b) Area strokes

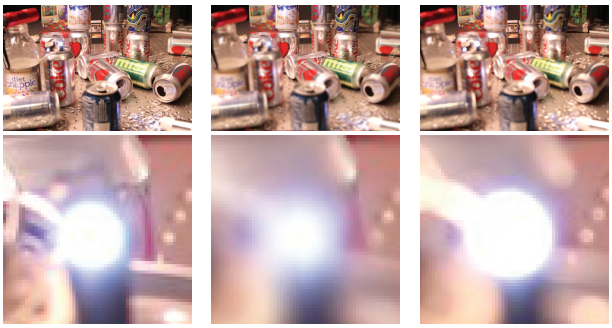


(c) User markup maps D (orange indicates blur; blue, sharpen)

(d) Freeform depth of field composites

Figure 13: *Freeform depth of field. (a) Pilot images specified by the user via simple strokes. The left image is focused on the row of upright blue playing cards. The right image is focused on the giraffe. (b) Area strokes denoting desired deviations from the pilot image. Defocus areas are marked in red; focus areas are marked in blue. The green stroke was used to select the pilot image. (c) User markup maps specifying the requested deviations from the defocus blur of the pilot images. We encode the requested deviation magnitude with saturation and its direction (blur or sharpen) with hue: orange indicates blur; blue indicates sharpen. White signifies regions left unspecified. (d) The final freeform depth of field composites produced by our pipeline. The left composite focuses sharply on the front and back card on the left, and the middle card on the right, while defocusing the others. The right composite focuses sharply on all the red cards and defocuses all the blue cards.*

ring; $D(\vec{p}) = 0$ is reserved for unspecified regions. Figure 13(c) shows some markup map examples. We then apply a cross-bilateral filter [Eisemann and Durand 2004] to D, respecting depth and color edges in the pilot image and weighting specified pixels more strongly than unmarked regions. We further apply a median filter to smooth the output. Once D is fully filtered, we create our desired defocus map $C^\star(\vec{p})$ from $C_0^\star(\vec{p})$ by pulling its value closer to or away from 0, depending on the sign of $D(\vec{p})$. The magnitude of $D(\vec{p})$ informs how strong its impact is. Finally, $C^\star$ can then be inserted into our pipeline to generate a halo-free composite.



(a) Focused slice (b) Synthetic blur of (a) (c) Defocused slice

Figure 14: *Saturated image regions like the specular reflections on the cans and the LED keychain (inset) visible in (a) have unknown true brightness. If a composite calls for blurring these regions, a synthetic blur like that presented in (b) will underestimate the correct brightness (shown in (c)) for a defocused image.*

Implementation

To demonstrate our user interface, we built a prototype application for iOS. Our implementation uses a client-server architecture, utilizing the phone for viewing and marking up composites, while the server performs the compositing operations. We used an Apple iPhone 4 connected over wi-fi to a 3GHz desktop computer with 4GB of RAM. Our focal stacks had 32 slices, each downsampled to 960×640 pixels to match the iPhone's screen resolution. The total latency between issuing a composite request and viewing the result is about 10 seconds—split roughly into 3 seconds computing $C^\star$, 5 seconds compositing, and 2 seconds of network overhead. We find it easiest to specify a composite iteratively, marking up an image, viewing the result, then adding further constraints. For these intermediate results we can drop the halo-free compositing constraint to cut the round trip time by 30%. When satisfied with the preliminary composite, the user can request a final, artifact-free result to be rendered. Figure 13 shows some composites and their corresponding user markups. Please refer to the supplementary video for a demonstration of our prototype.

6 Discussion and Future Work

In this paper, we presented a theory of focal stack compositing for depth of field control. We showed that focal stack composites can successfully emulate the images that would be produced by cameras with hypothetical focus and aperture values. As we demonstrated, there are precise geometric and arithmetic constraints that must be satisfied in order to create a halo-free focal stack composite. We applied our theory to three domains—reduced depth of field, extended depth of field, and freeform depth of field—using focal stacks captured with a variety of platforms. Below we discuss a number of theoretical and practical considerations for future work in focal stack compositing.

Theoretical Considerations

The framework for halo elimination we present is also applicable to other depth of field reduction techniques, such as [Hasinoff and Kutulakos 2007], that use synthetic blur in place of photographic blur. Figure 14 shows a comparison between using synthetic and photographic blur in simulating a wide-aperture photograph based on a halo-free sensor distance map. Although the synthetically blurred image appears reasonable, it fails to accurately portray the defocused specular highlights.

One limitation to reducing depth of field is that the extent to which it can be reduced is dependent on the location of the subject within the focal stack. If a subject is in the extreme foreground of the focal stack, it will be impossible to defocus the background any more than it already is in the slice that sharply images the subject. There simply is no stack slice that has a larger photographic defocus blur for the background. Depth of field reduction is similarly limited for extreme background subjects as well. In these situations where the proper photographic blur is not available, we can turn to synthetic blurring to get a similar effect—synthetic blur limitations aside.

Lastly, the quality of the composites is limited by the accuracy of the supplied depth map. Although our halo-correction step guarantees smooth transitions between stack slices regardless of depth map accuracy, the desirability of transitioning to a particular slice at all depends on the correctness of the depth map.

Practical Considerations

The acquisition of a focal stack is currently too slow for dynamic, real-world scenes. Nevertheless, the optics and sensors in current smartphones are more than capable of capturing a focal stack at full resolution in a fraction of a second: light-weight lenses can sweep through their full range in under 100ms, and modern mobile phone sensors can stream 5 megapixel images at 30fps. Currently our methodology is limited by the sensor-to-disk bandwidth; the total exposure time for capturing the focal stack, minus this overhead, is in the order of a single second. We expect that in the near future, a smartphone will be able to hold an entire focal stack in memory, alleviating this problem. Thus, we believe that future cameras, especially mobile phones, will be able to capture and store focal stacks in real-time without the user even being aware of the process.

Acknowledgments

The authors would like to thank Natasha Gelfand, Kari Pulli, and Sam Hasinoff for their helpful suggestions. David E. Jacobs is supported by a Hewlett Packard Fellowship. Jongmin Baek is supported by a Lucent Technologies Stanford Graduate Fellowship.

References

ADAMS, A., TALVALA, E.-V., PARK, S. H., JACOBS, D. E., ADJIN, B., GELFAND, N., DOLSON, J., VAQUERO, D., BAEK, J., TICO, M., LENSCH, H. P. A., MATUSIK, W., PULLI, K., HOROWITZ, M., AND LEVOY, M. 2010. The frankencamera: an experimental platform for computational photography. *ACM Trans. Graph. 29*, 4 (July), 29:1–29:12.

AGARWALA, A., DONTCHEVA, M., AGRAWALA, M., DRUCKER, S., COLBURN, A., CURLESS, B., SALESIN, D., AND COHEN, M. 2004. Interactive digital photomontage. *ACM Trans. Graph. 23*, 3 (Aug.), 294–302.

BAE, S., AND DURAND, F. 2007. Defocus magnification. *Computer Graphics Forum 26*, 3, 571–579.

BARSKY, B., AND PASZTOR, E. 2004. Rendering skewed plane of sharp focus and associated depth of field. In *ACM SIGGRAPH 2004 Sketch*, ACM.

BELL, C., 1992. Contrast-based autofocus mechanism. US Patent 5,170,202, 12. Assigned to Eastman Kodak Company.

COOK, R. L., PORTER, T., AND CARPENTER, L. 1984. Distributed ray tracing. In *ACM SIGGRAPH 1984 Conference Proceedings*, ACM, 137–145.

DOWSKI, E. R., AND CATHEY, W. T. 1995. Extended depth of field through wave-front coding. *Applied Optics 34*, 11.

EISEMANN, E., AND DURAND, F. 2004. Flash photography enhancement via intrinsic relighting. *ACM Trans. Graph. 23*, 3 (Aug.), 673–678.

GOODMAN, J. W. 1996. *Introduction to Fourier Optics*, second ed. McGraw-Hill.

HASINOFF, S. W., AND KUTULAKOS, K. N. 2007. A layer-based restoration framework for variable-aperture photography. In *ICCV 2007*, IEEE, 1–8.

HASINOFF, S. W., DUR, K. N. K. F., AND FREEMAN, W. T. 2008. Light-efficient photography. In *ECCV 2008*, Springer, 45–59.

ISAKSEN, A., MCMILLAN, L., AND GORTLER, S. J. 2000. Dynamically reparameterized light fields. In *ACM SIGGRAPH 2000 Conference Proceedings*, ACM, 297–306.

KOLB, C., MITCHELL, D., AND HANRAHAN, P. 1995. A realistic camera model for computer graphics. In *ACM SIGGRAPH 1995 Conference Proceedings*, ACM, 317–324.

KOSLOFF, T., AND BARSKY, B. 2007. An algorithm for rendering generalized depth of field effects based on simulated heat diffusion. Tech. Rep. UCB/ECS-2007-19, University of California, Berkeley.

LANMAN, D., RASKAR, R., AND TAUBIN, G. 2008. Modeling and synthesis of aperture effects in cameras. In *Proc. Computational Aesthetics in Graphics, Visualization, and Imaging*, Eurographics, 81–88.

LEVIN, A., FERGUS, R., DURAND, F., AND FREEMAN, W. T. 2007. Image and depth from a conventional camera with a coded aperture. *ACM Trans. Graph. 26*, 3 (July), 70:1–70:10.

LEVIN, A., HASINOFF, S. W., GREEN, P., DURAND, F., AND FREEMAN, W. T. 2009. 4D frequency analysis of computational cameras for depth of field extension. *ACM Trans. Graph. 28*, 3 (July), 97:1–97:14.

LEVOY, M., AND HANRAHAN, P. 1996. Light field rendering. In *ACM SIGGRAPH 1996 Conference Proceedings*, ACM, 31–42.

MOHAN, A., LANMAN, D., HIURA, S., AND RASKAR, R. 2009. Image destabilization: Programmable defocus using lens and sensor motion. In *ICCP 2009*, IEEE, 1–8.

NAGAHARA, H., KUTHIRUMMAL, S., ZHOU, C., AND NAYAR, S. 2008. Flexible depth of field photography. In *ECCV*.

NG, R. 2005. Fourier slice photography. *ACM Trans. Graph. 24*, 3 (July), 735–744.

SHNEIDERMAN, B. 1983. Direct manipulation: A step beyond programming languages. *Computer 16*, 8 (Aug.), 57 –69.

STREIBL, N. 1985. Three-dimensional imaging by a microscope. *J. Opt. Soc. Am. A 2*, 2 (Feb), 121–127.

SUBBARAO, M., AND SURYA, G. 1994. Depth from defocus: a spatial domain approach. Springer, vol. 13, 271–294.

VAISH, V., GARG, G., TALVALA, E.-V., ANTUNEZ, E., WILBURN, B., HOROWITZ, M., AND LEVOY, M. 2005. Synthetic aperture focusing using a shear-warp factorization of the viewing transform. In *CVPR 2005*, IEEE, 129–136.

ZHOU, C., AND NAYAR, S. 2009. What are good apertures for defocus deblurring? In *ICCP 2009*, IEEE, 1–8.