Painted Aperture for Portraits

Edward Luong (edluong@stanford.edu)

1 Introduction

Photographers often wish to control the depth of field. In particular, for portrait photography, a shallow depth of field can have dramatic effects on the composition and the feel of the picture. For example, a shallow depth of field can help direct attention to the subject by blurring out objects too close or too far from the camera, such as distracting objects or faces. Also, focusing on an off-center object can create an interesting shot composition.

Depth of field results from geometric optics when using a non-pinhole camera. The amount of blur is a function of depth and aperture size. High-end cameras with large lenses provide photographers the ability to open and close the aperture which expands or shrinks the depth of field. Common consumer cameras and especially mobile phones have limited capabilities and rarely allow the user to control depth of field beyond having a Macro setting.

My project aims to bring this capability to low-end cameras such as ones found on mobile phones. Since we cannot change the optical system of these cameras, we achieve the depth of field effect synthetically. My system captures multiple images of the scene with different camera positions. The images are blended together to simulate defocus blur. All in all, the system is intuitive and easy to use. While the current pipeline runs mostly offline, performance bottlenecks are not fundamental and an online solution is likely plausible in the near future.

2 Implementation

Most of the implementation occurs offline with only the capturing stage occuring on the Nokia N900. To simulate a camera with a limited optical system, I set the focus to the "farthest" setting. The goal of the system is to generate a full resolution image with user-controlled synthetic defocus blur.

2.1 Capture

Each image can be treated as a point sample of the lightfield. In order to synthesize a new image using a larger aperture, we want to gather a well distributed point sampling of the lightfield passing through the larger aperture. Capturing a point sample requires a non-motion blurred image. A "uniform" sampling means each captured image should be spaced out across the lightfield space [Mitchell 1991].

The capturing step first captures a single high resolution (2592×1968) "detail" image, followed by numerous low resolution (viewfinder resolution, 640×480) "blur" images. Capturing high resolution images have higher latency because the camera pipeline must be flushed. This is only a limitation on the N900; however, using low resolution blur images have other advantages. Since the blur images are only used to add blur to the final image, capturing them at lower resolution should not affect quality. In fact, for the purpose of defocus blur, capturing 4 images at the low resolution provides more useful information than a insgle high resolution image. Moreover, lower resolution reduces the workload for the rest of the pipeline.

For capturing the blur images, I implemented two different capturing modes. The first mode, SPOTS, optimized for capture quality. The second mode, STREAM, optimizes for speed. While this step could benefit from plenty of tweaking, I found these to be sufficient at least for early tests of the system.

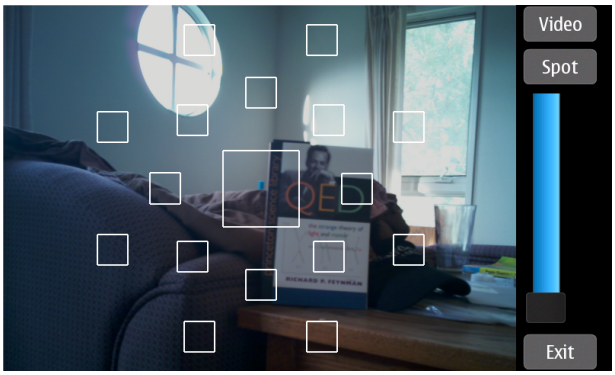


Figure 1: *The interface for* SPOTS. *The overlayed boxes (in white) trigger a capture when the user touches them.*

2.1.1 SPOTS

SPOTS aims to maximize the quality of the capture. In order to minimize motion blur, the user should know when roughly when the image will be captured. This allows the user to prepare and stabilize the shot. Uniform sampling is achieved by having the user move the camera and capturing images at the appropriate positions. We want to avoid both undersampling and oversampling the lightfield.

SPOTS addresses these concerns by a novel interface that takes advantage of the N900's touchscreen. The viewfinder has several marked regions ("spots") that trigger an image capture, shown in Figure 1. The user anchors his or her finger/stylus at a fixed point in space while touching the screen. The camera is then moved (while keeping the finger/stylus fixed) such that each spot is triggered.

Uniformity is accomplished by spacing the spots uniformly in concentric rings on the viewfinder. The number of rings can be increased or decreased to change the sampling density. Since the user can predict when each image will be captured, motion blur can hopefully be reduced.

This method suffers from major usability problems. Moving the camera while keeping the other hand steady requires quite a bit of dexterity. Also, depending on the number of spots, capturing can take a while. Over this time, it may be difficult for the subject to remain still and to control camera shake. Ideally, however, the rest of pipeline could be made robust to such changes.

2.1.2 STREAM

STREAM tries to fix these usability issues at the cost of quality. The camera streams in a frames as quickly as possible while the user is instructed to wave the camera in circles. This method can quickly capture 32 or 64 frames in just a few seconds. One can imagine other modifications to this system to minimize capturing motion blur frames, such as rejecting frames where there is not enough sharpness. I found that as long as the shutter time is short enough and the movement is slow enough, motion blur does not turn out to be a problem. Additionally, one can imagine an interface that

directs the user to areas that are undersampled (or away from areas that are oversampled). Alternatively, if camera position can be determined in real-time, the camera can also be configured to trigger a capture whenever the camera is in an appropriate position.

2.2 Registration

For registration, features are extracted from each image. In my implementation, I use SIFT features; however, any robust features would suffice. Since we are only translating the camera, the actual feature would only need to handle translation of the foreground object. SIFT features have the added benefit of being very robust to scale which handles cases where some features may be blurred. Regardless of the exact choice of features, feature extraction is important in reducing the problem to a sparse data set.

The detail image is downsampled to the blur resolution and features are extracted from all the images. The user selects a set of features $\mathcal{S}$, lying in the desired plane of focus. Feature selection only needs to be approximate, however, features off the plane of focus should be outliers.

Each feature in $\mathcal{S}$ from the downsampled detail image is matched to the "nearest" feature in each blur image. For each detail-blur image pair, RANSAC is used on the matches to find a homography between the images. RANSAC makes the pipeline more robust to user selection errors and feature matching errors.

2.3 Reconstruction

A proper reconstruction of the scene viewed from a camera with a synthetic aperture would reconstruct the signal from from the point samples and the reconstructed signal would be resampled at the appropriate aperture positions [Chen and Williams 1993; Vaish et al. 2004; Vaish et al. 2005]. In theory, this should allow the system to work with any point sampling (not necessarily uniform) of the aperture.

If we assume that the images represent a uniform sampling of the lightfield, we can instead just transform the images using the computed homography and blend (average) all the images together. The computed homography will align objects in the plane of focus. Defocus blur will result from parallax–objects at different depths will translate by different amounts.

2.4 Final Composition

The previous stages operate at a lower resolution. In this final stage, we output a final image at the detail resolution that contains the synthetic defocus blur. The blended blur images are upsampled to the higher resolution. This removes high frequency detail which we recover from the original detail image. A matte is computed by determining which pixels have not displaced (and hence are in focus). The final image is composed by selecting pixels from either the detail image or the upsampled blur image according to the mask.

The matte can be computed in several ways. For instance, one can use optical flow and use the length of the flow vectors to determine how far a pixel moved. I used a simplistic approach. For each pixel location, I compare the corresponding color values from the detail and the blurred images. I consider the pixel as unmoved if the difference in each color channel is within some small threshhold. I often search in a small 3×3 window to deal with problems from the loss of high frequency information in the blurred images. Further, I apply a Gaussian blur to the matte to smooth the matte.

It is important to note that a binary matte, such as one generated using a graph cut [Boykov and Jolly 2001; Rother et al. 2004] algorithm, will probably suffice. Since we are composing the detail image onto a very similar image, artifacts from a hard mask are not as obvious. However, I believe these matte selection algorithms may not yield the best results since the matte is not exactly "foreground" extraction.

3 Results and Limitations

3.1 Image Quality

Currently, the matte computed during the final composition step is not the best quality and tend to add too much detail in regions that have been blurred out. Figure 2 shows that too much of the table is sampled from the detail image and some of the blurred edges are too sharp. Further tweaking of the matte parameters or a different approach to this final composition can likely fix these problems. For the remainder of the results, I analyze the lower resolution blur (which contains many artifacts as well).

3.2 Ghosting

The images produced tend to have the look of a limited depth of field; however, they are far from perfect. Ghosting artifacts tend to be the most offensive artifacts in the final image. These artifacts are most apparent when using a sparse sampling. For instance, when using just 8 images, Figure 3(a), the defocus blur looks like shifted copies of the background. As the number of images captured increases, the blur becomes averaged and looks smoother, Figure 3(b). Interestingly enough, I found that ghosting artifacts tend to be removed when using around 64 blur images, which matches the literature for achieving high-quality depth of field in computer graphics [Fatahalian et al. 2009].

3.3 Scenes

The system is sensitive to the extracted features. If no features can be found and matched, the system will be unable to align the captured images. Most scenes tend to have plenty of SIFT features; however, these features may not lie on the plane of focus. The features should also be coplanar on the desired plane of focus. Either the object should indeed be planar or the camera must be positioned sufficiently far from the object. Moreover, moving objects pose a problem for this system since it tries to align on a set of features that remain fixed on the desired plane of focus. Features on the silhouette are more likely to remain fixed and would be preferred if the subject is, for example, the face of a person.

The scene must also be composed in a particular way to achieve a noticeable amount of defocus blur. Defocus blur results from parallax. If the user wants certain objects to be out of focus, they must be sufficiently far from the plane of focus. This problem exists when limiting depth of field with high end cameras; however, in this system, the problem may be exacerbated if the user must already be far from the subject for the coplanar assumption to hold.

Given these restrictions, my system tends to produce moderate looking blur with great simplicity. Moreover, this approach works relatively well for different depth planes, allowing for a coarse approximation to lightfield rendering [Levoy and Hanrahan 1996]. Figure 5 shows a very discretized focus stack. Even though it is discontinuous, it still provides the user with some amount of post-capture refocusing. Moreover, novel types of blur that are not possible optically can be achieved. Figure 5(c) shows an image with focal planes at the front (numpad) and back (mug). This is accomplished by having half of the images align on the front, and the other half aligning on the front.

(a) Final composite image. The matte includes too many regions from the detail image. The resulting image tends to look splotchy.

(b) Upsampled blur image. There is a loss of detail on Jesus' face and the teapot.

Figure 2: *A comparison of the final composite and the upsampled blur image*

3.4 Performance

The current performance bottleneck is registration. In fact, once the data has been reduced to a sparse set of features, the remaining computation is fairly light and quite feasible to run as a post-process on the phone. Computing the homography via RANSAC is very fast, and its performance can be improved by limiting the set of samples used. RANSAC also benefits from having parameters that can serve as a performance-quality knob (for example, the number of iterations or the inlier threshhold). One can imagine a system where previews of the captured lightfield can be displayed directly on the phone after all the frames are captured. A full, higher quality reconstruction can be performed offline.

The bottleneck in registration would benefit immensely from hardware acceleration. The ability to reduce a dense data set to a sparse one enables the use of much faster algorithms (optical flow versus finding a homography) and could have a large impact on numerous applications in computational photography. If features could be extracted in realtime, an interface that tracks the camera position and directs the user would be more computationally feasible.

4 Discussion

For future work, I would like to have more control on the aperture. For example, the user may only wnat to introduce a small amount of defocus blur. Currently, the system will use all samples; however, one can image a system that only uses images with a limited range of translation. Additionally, if resampling was done correctly, one could imagine also generating a synthetic bokeh by allowing the user to draw the shape of the bokeh. Also, performance could be improved to actually fully run on the N900.

Acknowledgments

Thanks to Jongmin Baek for advice on the project.

References

BOYKOV, Y., AND JOLLY, M.-P. 2001. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *ICCV*, 105–112.

CHEN, S. E., AND WILLIAMS, L. 1993. View interpolation for image synthesis. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 279–288.

FATAHALIAN, K., LUONG, E., BOULOS, S., AKELEY, K., MARK, W. R., AND HANRAHAN, P. 2009. Data-parallel rasterization of micropolygons with defocus and motion blur. In *HPG '09: Proceedings of the Conference on High Performance Graphics 2009*, ACM, New York, NY, USA, 59–68.

LEVOY, M., AND HANRAHAN, P. 1996. Light field rendering. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 31–42.

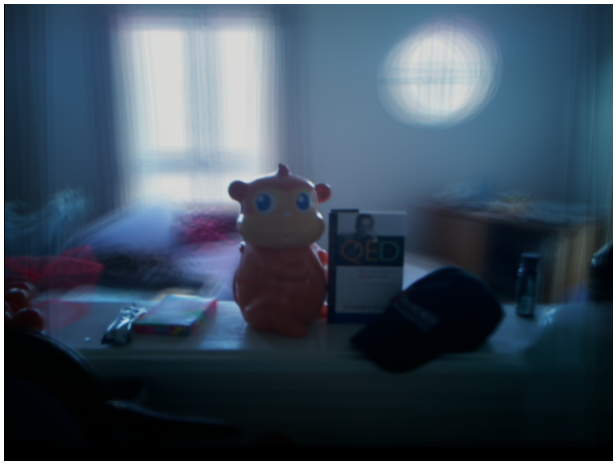
MITCHELL, D. 1991. Spectrally optimal sampling for distribution ray tracing. *Computer Graphics (Proceedings of SIGGRAPH '91) 25*, 4, 157–164.

ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. 2004. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph. 23*, 3, 309–314.

VAISH, V., WILBURN, B., JOSHI, N., AND LEVOY, M. 2004. Using plane + parallax for calibrating dense camera arrays. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on 1*, 2–9.

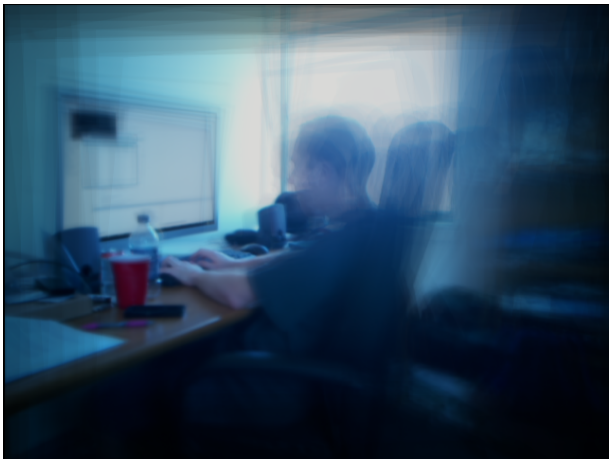
VAISH, V., GARG, G., TALVALA, E.-V., ANTUNEZ, E., WILBURN, B., HOROWITZ, M., AND LEVOY, M. 2005. Synthetic aperture focusing using a shear-warp factorization of the viewing transform. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, IEEE Computer Society, Washington, DC, USA, 129.

(a) Using 8 images for blur.



(b) Using 16 images for blur.

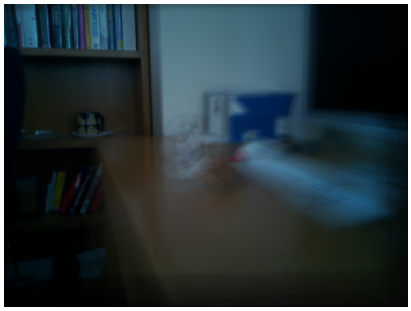
Figure 3: *The quality of the defocus blur depends on the number of images captured.*



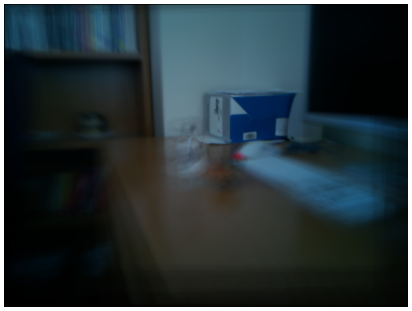
(a) Motion and insufficent features moves the infocus region to the person's hands instead of face.

(b) The background is too close to the foreground. A zoom blur effect is achieved instead of defocus blur.

Figure 4: *Other artifacts in the defocus blur.*



(a) Focusing on the mug on the shelf.



(b) Focusing on the blue box.

(c) Focusing on both the numpad and shelf.

Figure 5: *All of these images are rendered using the same set of captured images. The user can pick different depth planes after capturing.*