Light Field Occlusion Removal

Shannon Kao
Stanford University
kaos@stanford.edu

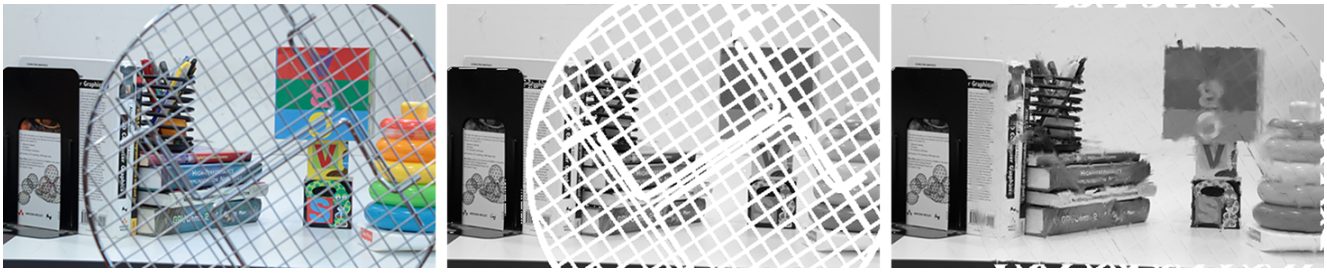

Figure 1: Occlusion removal pipeline. The input image (left) is part of a focal stack representing a light field. Each image is taken with the camera shifted horizontally a set amount. The occlusion is detected and masked (center), using the obstruction-free photography technique [8] which leverages motion fields to detect occlusions. Finally, the missing pixels are filled using patch-based texture synthesis (right) with patches from the median image.

Abstract

We present a novel method for detecting occlusions and in-painting unknown areas of a light field photograph, based on previous work in obstruction-free photography and light field completion. An initial guess at separating the occluder from the rest of the photograph is computed by aligning backgrounds of the images and using this information to generate an occlusion mask. The masked pixels are then synthesized using a patch-based texture synthesis algorithm, with the median image as the source of each patch.

1. Introduction

Occlusions are a common problem in photography. In some cases, scenes may have accidental occlusion, such dirty lenses or architectural artifacts; in others, occlusion may be unavoidable, as in systems with fixed cameras, such as surveillance networks.

Removing such objects, then, is the task of selecting the occluding region and filling in the scene behind it. While occluders necessarily represent gaps in information about the scene, there are ways to augment data collection in order to conduct the in-painting of occluded regions. In traditional cameras, photographs from multiple angles provide this information. One of the strengths of a light field photograph is that it allows us to see "around" occluders, leveraging the extra information from the light field to fill in unseen areas of the photograph. With the advent of light field cameras it has become possible to remove occluders with the data from a single light field image.

The term "occlusion" often extends to both physical occlusions, which are completely opaque, and reflections, which preserve some of the scene behind them. While this project focuses solely on physical occlusions, the framework could easily be extended into the realm of reflections.

The system presented here takes an input sequence of a light field, which can be 4D image, as generated by Lytro cameras, or a simple horizontal light field consisting of evenly shifted images. It then automatically registers the images and uses these aligned images to detect the occlusion. Finally, the system masks the occlusion and completes the unknown regions of the scene using existing data from the image, rendering a full light field with the occlusion completely removed, as shown in figure 1.

2. Related Work

Removal of occluding objects is a commonly-researched subject, and there are many methods addressing the problem in conventional cameras. The process of occlusion removal can be broken down into two key areas: occlusion detection and image completion.

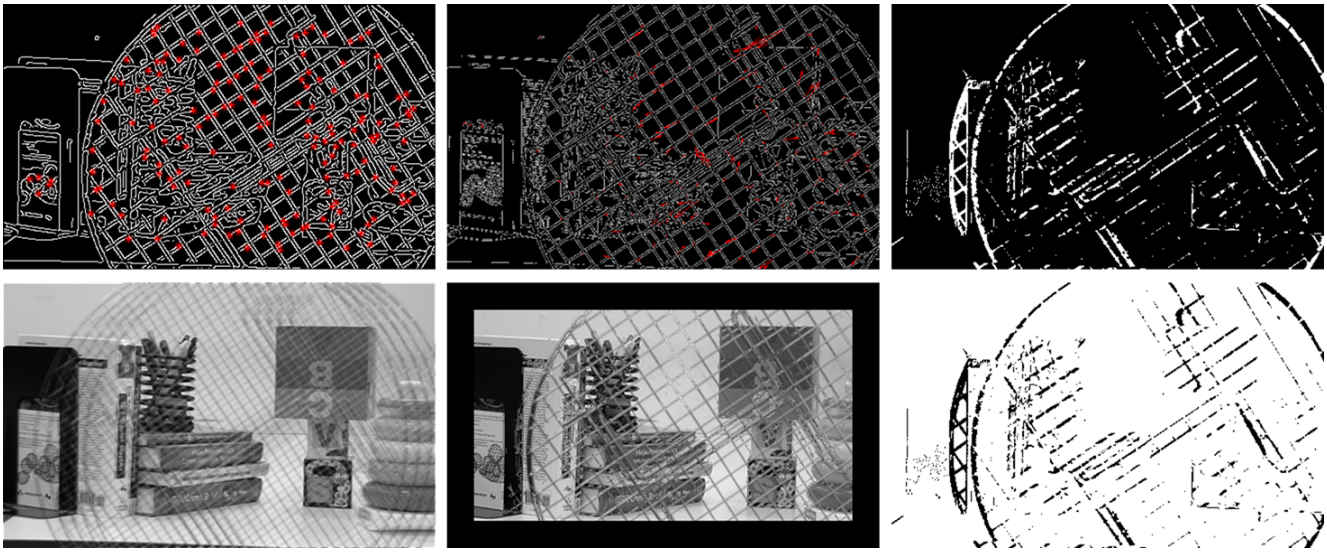

Figure 2: To detect the occlusion we run the Canny edge detector and extract corners (top left). These corners are used to generate a motion field (top center). From the motion field, we calculate a transformation to align the images and take an average (bottom left). Finally, we assign pixels to the occlusion mask (top right) or the background (bottom right) based on the difference in intensity between any pixel and the average.

2.1. Conventional occlusion removal

Favaro [5] presents an algorithm that requires multiple, highly-textured images in order to see beyond occlusions in a scene. Gu [6] provides a method that requires several calibration images to remove image artifacts due to dirty camera lenses. Yamashita [9] targets occluding fences, specifically, using information from a series of images at different focal depths to remove these artifacts.

However, capturing multiple photographs for occlusion removal is a nontrivial task, as it requires a certain level of precision and consistency between photographs. The light field camera, records this data in a single image, removing the need for multiple photographs. It would be ideal to leverage this additional data for more accurate, occlusion removal, from a single photograph.

2.2. Occlusion detection

Occlusion detection is often a manual step in the occlusion removal pipeline. Many systems allow users to manually specify which regions of the image are occlusions.

Alternatively, classical computer vision has many techniques to register images and detect occlusions, which are here defined as objects that have significant shifts in position across the focal stack when the background is fully registered. There are also techniques for removing regular occlusions, such as fences or evenly spaced grids [7]. These methods are all tailored for a specific type of occlusion, relying on this expected structure to perform the occlusion detection and removal.

Xue et al [8] introduced an ”Obstruction-Free Photography” technique that simply required a series of horizontally shifted images in order to detect the occluding object. This system builds on that algorithm, utilizing the initialization techniques they present. However, the Obstruction-Free Photography methodology requires that every pixel in the ”ground-truth” scene be visible somewhere in the captured image stack. By utilizing a more generic image completion technique, this system relaxes that constraint.

2.3. Image completion

Once the occluding region has been identified, a system must determine how to best fill in the missing information. The obvious approach to occlusion removal is to simply take the average of all pixels in the light field patch, but this approach does not preserve detail or utilize any of the depth or positional information encoded in a light field photograph.

A next step would be to simply to mask the occluded areas, and use texture synthesis [4] or image in-painting [1] to fill in the unknown regions. Unfortunately, these approaches require a certain density of data to be effective, and relevant information is often scattered across light fields. Simple integration operations fail for the same reasons [3].

Yatziv et al [10] implemented a median-image based synthesis technique that performs very well on large occlusions, and our occlusion removal system builds on this, incorporating previous work to automate the detection of

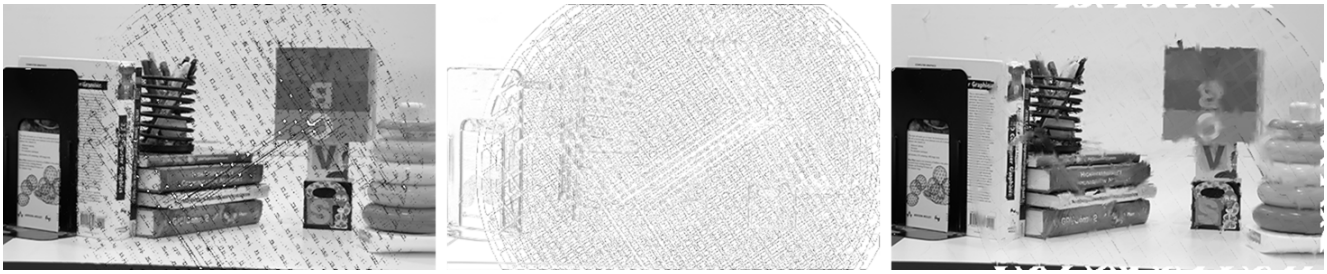

Figure 3: The median image is calculated by taking the median non-null pixel in each patch of the light field (left). The confidence value for each median pixel is also visualized (center). Using the median and confidence, we are able to synthesize the missing portions of the image (right).

occlusions rather than relying on the user to specify the occlusion mask.

3. Data

The system was tested on a sequence of photographs from a controlled setting, with a ground truth decomposition between occluder and background. The camera was shifted horizontally a fixed amount and scene was structured such that the occluder was between the background and the camera. The occluder was also required to be opaque.

4. Occlusion detection

The two key stages of occlusion removal are occlusion detection and image completion. Given a photograph with no metadata, the system uses reference layers to detect which pixels are part of the occlusion and remove them. This initial occlusion detection, which can be defined as a decomposition of the image into occlusion and background components, is based off of the initialization stages of the Obstruction-Free Photography [8] algorithm.

The occlusion detection first computes a motion field for each layer, then uses these motion fields to align each image in the focal stack. These aligned images are used to calculate the occlusion and background components. The full process is visualized in figure 2.

4.1. Motion estimation

We first pick a central layer of the focal stack to be the reference image. The motion-based occlusion detection system calculates a motion field between each layer and the reference image, then uses this motion field to align the images.

Rather than computing a motion field over individual pixels, the system uses an ”edge flow” algorithm, computing the flow between edges in each layer. We first use the Canny edge detector to extract the edges and isolate the corners within the image. The Lucas-Kanede method is used to approximate the motion between each individual layer and the reference layer.

4.2. Image alignment

Having obtained this sparse motion field between each layer of the focal stack and the central reference layer, we generate a transformation to align the images. This system makes an assumption that the transformation is a simple translation, and finds the best-fit translation to align the edge pixels of the frame with the reference, assuming, as the original paper does, that the background pixels are dominant.

4.3. Decomposition

With the aligned images it becomes straightforward to calculate the decomposition of a layer into the occlusion and background components. We first take the average of each layer in the focal stack. The backgrounds have been registered so the occluding object is the only thing shifting between frames. Then, for each frame, we compare the intensity of the average pixel with the intensity of the input pixel. If this difference in intensity is above a certain threshold, we conclude that this pixel was occluded in the original input, and assign it to the occlusion layer.

We also experimented with using a spatial coherence metric rather than an average. Instead of taking the mean of all values in the light field patch, we picked the pixel that was most similar to the other pixels in it’s immediate neighborhood. This, however, did not generate noticeably better results.

With this process, we’ve obtained an occlusion mask which identifies which pixels in the image are most likely part of the occluding object.

5. Image completion

The input to the image completion stage of the occlusion removal pipeline, then, is a focal stack with missing

pixels where the occlusion has been masked out. The system must then use the existing data to in-paint, or complete, these unknown portions of the scene. Two approaches were implemented, with varying degrees of success.

5.1. Median image-based completion

The median image-based completion approach, introduced by Yatziv and Levoy [10], involves first calculating a median image for the light field then using this median image to synthesize the unknown portions of the light field, as shown in figure 3.

5.1.1 Median image and confidence

The median image itself is simply the median pixel within each patch of the light field. There is a corresponding confidence value, $C_i \in [0, 1]$, for each pixel i in the median image, which is based the number of unknown pixels in the patch, as well as how similar the median pixel is to the other known pixels.

$$C_i = D_i * Pr_i \tag{1}$$

$$D_i = \frac{|UV|}{U + V - 1}; Pr_i = 1 - \frac{\sum_N (I(p_i) - I(p_M))^2}{|UV| * \frac{I_{max}^2}{2}} \tag{2}$$

D_i is the percentage of known pixels over the total number of pixels (including occluded pixels). Pr_i is the similarity of pixel p_M, from the median image, with each pixel p_i in the set of neighbors N. U and V are the pixels along the u and v directions in the light field, and $|UV|$ is the set of un-occluded pixels.

5.1.2 Synthesis

Using this median image, we perform patch-based texture synthesis using the median as the source for texture patches, rather than the individual focal stack layers. The patch-based texture synthesis is fairly standard, iterating over the neighborhood of each unknown pixel to find the most similar corresponding patch in the median image. The distance formula used here is simply the sum of squared differences over known pixels, ignoring values that have not yet been synthesized.

The confidence is also factored in at this stage–rather than performing the texture synthesis exclusively on unknown pixels, we also perform it on pixels with a low confidence value. The resulting synthesized pixel is added into the image using a weighted average based on the confidence value.

5.2. Focal stack propagation

We also experimented with a focal stack propagation technique [2], which was similar to the median image completion. Rather than using the median image, though, this technique first calculates an all-in-focus image, along with a depth field. It then uses the all-in-focus image to perform the synthesis. As each patch is synthesized, the system checks the depth field and applies a Gaussian blur kernel to the synthesized portions of the image in order to appropriately blend it with the existing portions of the scene.

6. Results

The results of occlusion detection and image completion on the input light field can be seen in figure 6. The optical-flow based occlusion detection was reasonably effective, as it clearly masked the right regions within each layer of the light field focal stack. However, in order to test the image completion algorithm, we manually masked the input images and performed texture synthesis on that data in order to more effectively visualize the results.

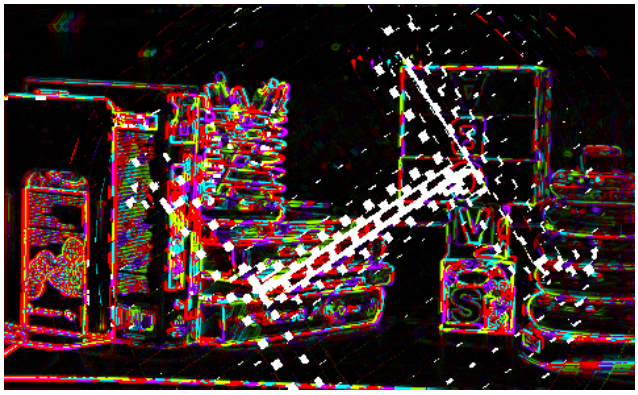

Figure 4: Depth, calculated from the input light field. The result is not particularly accurate in this test case, as the background of the scene is fairly uniform in terms of depth.

The focal stack propagation technique was not particularly effective because the depth recovered from the light field (figure 4) was very inconsistent. The median image light field completion method was significantly more accurate.

In general, the algorithm was able to recover a good deal of detail, as can be seen in the spiral pattern on the pencil holder and the letter 'A' in the stack of blocks, which is almost completely obscured in many of the input images.

Visually comparing the image to the results from Xue et al (figure 5), we can see that the optical flow and iterative optimization approach (left) achieves a cleaner final image, with more details preserved. However, our approach has

Figure 5: Comparison between the image recovered from Xue et al (left) and the image recovered from our system (right).

fewer artifacts remaining from the occluder, and in general allows for relaxed constraints on the input image sequence.

7. Future Work

While this pipeline produced very reasonable results for occlusion removal, there are several areas where improvement is necessary before this system can be of any practical use.

7.1. Parallelization

The most crucial next step is some form of parallelization or optimization. The texture synthesis stage of the current algorithm is extremely slow, taking up to five minutes to fully synthesize one image. The performance on a larger data set, such as a full Lytro light field image, would be almost too slow to utilize. Parallelizing the texture synthesis could greatly improve the overall performance, and optimization across the entire pipeline would certainly be beneficial.

7.2. Optimization

The Obstruction-Free Photography [8] technique employs an iterative optimization in cycling back and forth between the motion field and the decomposed background and occlusion layers. Applying such an iterative process to this pipeline, where the output of the median filter was used as input into another cycle of obstruction detection and removal, would probably benefit the accuracy of the final result. The current performance of the system is prohibitive, but with a faster system one could run more iterations. As the current system is fairly thorough, possibly only a few iterations would be required before a satisfactory final image is generated.

7.3. Alternative approaches

Finally, there are several interesting alternatives to occlusion detection that would be interesting to explore. Light fields offer positional and depth information, in addition to multiple views of the scene, and it would be interesting to try and factor this information into a more accurate or faster scene decomposition.

8. Conclusion

The light field occlusion removal system described in this paper represents a full pipeline, from occlusion detection to image completion. It uses motion fields to detect the occlusion, then applies texture synthesis techniques to in-paint the missing image regions. The use of texture synthesis techniques allows us to relax constraints on the input images, permitting fewer images and larger occlusions.

References

[1] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 417–424, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[2] T. Broad and M. Grierson. Light field completion using focal stack propagation. In *ACM SIGGRAPH 2016 Posters*, SIGGRAPH '16, pages 54:1–54:2, New York, NY, USA, 2016. ACM.

[3] A. Duci, A. J. Yezzi, S. K. Mitter, and S. Soatto. Region matching with missing parts. *Image Vision Comput.*, 24(3):271–277, Mar. 2006.

[4] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 341–346, New York, NY, USA, 2001. ACM.

[5] P. Favaro and S. Soatto. Seeing beyond occlusions (and other marvels of a finite lens aperture). In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–579–II–586 vol.2, June 2003.

[6] J. Gu, R. Ramamoorthi, P. Belhumeur, and S. Nayar. Removing image artifacts due to dirty camera lenses and thin occluders. In *ACM SIGGRAPH Asia 2009 Papers*, SIGGRAPH Asia '09, pages 144:1–144:10, New York, NY, USA, 2009. ACM.

[7] Y. Mu, W. Liu, and S. Yan. Video de-fencing. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(7):1111–1121, July 2014.

[8] T. Xue, M. Rubinstein, C. Liu, and W. T. Freeman. A computational approach for obstruction-free photography. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 34(4), 2015.

[9] A. Yamashita, A. Matsui, and T. Kaneko. Fence removal from multi-focus images. In *Proceedings of the 2010 20th International Conference on Pattern Recognition*, ICPR '10, pages 4532–4535, Washington, DC, USA, 2010. IEEE Computer Society.

[10] L. Yatziv, G. Sapiro, and M. Levoy. Lightfield completion. In *Image Processing, 2004. ICIP '04. 2004 International Conference on*, volume 3, pages 1787–1790 Vol. 3, Oct 2004.

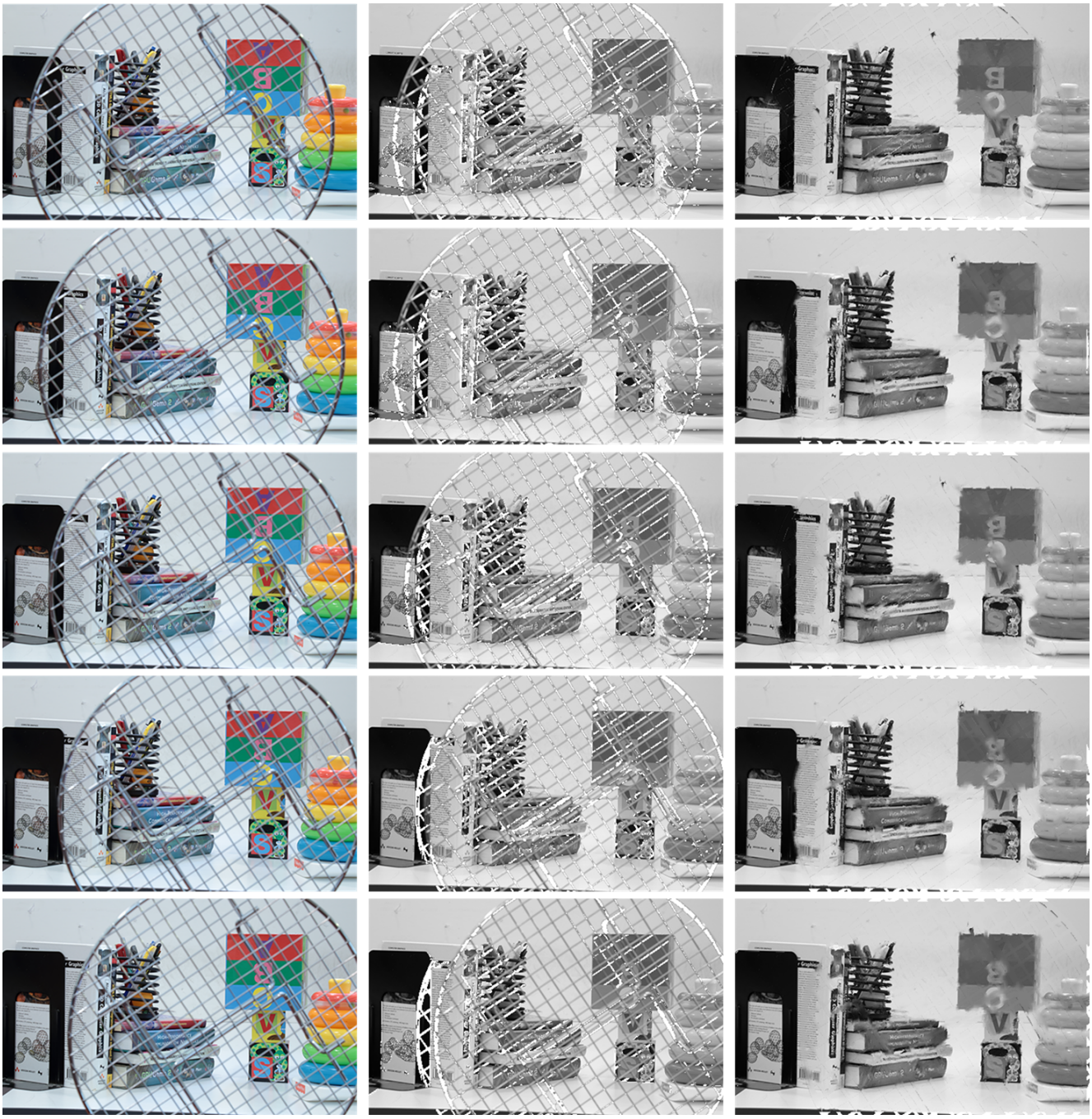

Figure 6: The input light field, consisting of five horizontally shifted images (left). The results of optical-flow based obstruction detection, with the obstruction masked out (center). The results of texture synthesis on the masked light field (right).