Focused plenoptic camera and rendering

Todor Georgiev
Adobe Systems Inc.
345 Parke Avenue
San Jose, California 95110
E-mail: tgeorgie@adobe.com

Andrew Lumsdaine
Indiana University
Computer Science Department
215 Lindley Hall
Bloomington, Indiana 47405

Abstract. *Plenoptic cameras, constructed with internal microlens arrays, capture both spatial and angular information, i.e., the full 4-D radiance, of a scene. The design of traditional plenoptic cameras assumes that each microlens image is completely defocused with respect to the image created by the main camera lens. As a result, only a single pixel in the final image is rendered from each microlens image, resulting in disappointingly low resolution. A recently developed alternative approach based on the focused plenoptic camera uses the microlens array as an imaging system focused on the image plane of the main camera lens. The flexible spatioangular tradeoff that becomes available with this design enables rendering of final images with significantly higher resolution than those from traditional plenoptic cameras. We analyze the focused plenoptic camera in optical phase space and present basic, blended, and depth-based rendering algorithms for producing high-quality, high-resolution images. We also present our graphics-processing-unit-based implementations of these algorithms, which are able to render full screen refocused images in real time.* © 2010 SPIE and IS&T.
[DOI: 10.1117/1.3442712]

1 Introduction

Integral photography, introduced by Ives and Lippmann over 100 years ago[1,2] has more recently reemerged with the introduction of the plenoptic camera. Originally presented as a technique for capturing 3-D data and solving computer-vision problems,[3,4] the plenoptic camera was designed as a device for recording the distribution of light rays in space, i.e., the 4-D plenoptic function or radiance. The light field and lumigraph, introduced to the computer graphics community, respectively, in Refs. 5 and 6, established a framework for analyzing and processing these data. In 2005, Ng *et al.*[7] and Ng[8] improved the plenoptic camera and introduced new methods of digital processing, including refocusing.

Because it captured the full 4-D radiance, Ng's handheld plenoptic camera could produce effects well beyond the capabilities of traditional cameras. Image properties such as focus and depth of field could be adjusted after an image had been captured. Unfortunately, traditional plenoptic cameras suffer from a significant drawback; they render images at disappointingly low resolution. For example, images rendered from Ng's camera data have a final resolution of 300×300 pixels.

A different approach, called "full-resolution lightfield rendering,"[9] can produce final images at much higher resolution based on a modified plenoptic camera (the "focused plenoptic camera"[10]). This modified camera is structurally different from the earlier plenoptic camera with respect to microlens placement and microlens focus. These structural differences in turn result in different assumptions about the sampling of the 4-D radiance. The traditional plenoptic camera focuses the main lens on the microlenses and focuses the microlenses at infinity. The focused plenoptic camera instead focuses the main camera lens well in front of the microlenses and focuses the microlenses on the image formed inside the camera—i.e., each microlens forms a relay system with the main camera lens. This configuration produces a flexible trade-off in the sampling of spatial and angular dimensions and enables positional information in the radiance to be sampled more effectively. As a result, the focused plenoptic camera can produce images of much higher resolution than can traditional plenoptic cameras.

Other radiance-capturing cameras similar to the focused plenoptic camera include the following: the microlens approach of Lippmann,[2] the thin observation model by bound optics[11] (TOMBO), the handheld plenoptic camera,[7] Fife *et al.*'s multiaperture image sensor architecture,[12,13] and the Panoptes sensor.[14]

This paper presents the design and analysis of the focused plenoptic camera and associated image rendering algorithms. In particular, it describes

1. the background of the plenoptic camera 2.0, including basic radiance theory and modeling, the plenoptic camera 1.0, and other radiance-capturing cameras
2. a complete development of the focused plenoptic camera, including derivation of its sampling in optical phase space, basic rendering algorithms, and a detailed description of the hardware

Paper 10039SSRR received Mar. 5, 2010; revised manuscript received Apr. 9, 2010; accepted for publication Apr. 20, 2010; published online Jun. 11, 2010.

1017-9909/2010/19(2)/021106/11/$25.00 © 2010 SPIE and IS&T.

3. a development of high-quality rendering algorithms that provide high-resolution realistic camera effects, such as depth of field and refocusing
4. a presentation of a computational framework for working with focused plenoptic camera data, including implementations of rendering algorithms and their efficient realization on modern GPU (graphics processing unit) hardware.

2 Radiance Theory and Modeling

Following the development in Refs. 9 and 10, we denote the radiance at a given plane perpendicular to the optical axis by $r(q,p)$, where q and p represent position and direction in ray space, respectively. Compactly, a coordinate in ray space is represented by $x=(q,p)^T$.

Rays are transformed by the application of optical elements. An arbitrary ray transfer matrix $\mathbf{A}$ transforms each ray according to

$$x' = \mathbf{A}x. \tag{1}$$

Refraction by a lens and travel of rays in free space are, respectively, described by the matrix transforms $\mathbf{L}$ and $\mathbf{T}$:

$$\mathbf{L} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix}, \quad \mathbf{T} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}. \tag{2}$$

Optical transforms of rays induce corresponding transforms of functions (such as radiance) defined on ray space. Let $\mathbf{A}$ be an optical transform of Eq. (1), and consider the induced transformation of $r(x)$ to $r'(x)$. Since all optical transfer matrices satisfy det $\mathbf{A}=1$, and assuming conservation of energy, we have the radiance conservation property of all optical systems, i.e., we must have $r'(x')=r(x)$. Taken with $x'=\mathbf{A}x$, we must also have $r'(\mathbf{A}x)=r(x)$. Considering a ray $y=\mathbf{A}x$, we thus obtain $r'(y)=r(\mathbf{A}^{-1}y)$. Since y is an arbitrary ray, we obtain the radiance transformation formula:

$$r'(x) = r(\mathbf{A}^{-1}x). \tag{3}$$

Finally, an image is related to the radiance in the following way. The intensity of an image at a given spatial point, denoted $I(q)$ is the integral of the radiance over all of the rays incident at that point, i.e.,

$$I(q) = \int_p r(q,p)\mathrm{d}p. \tag{4}$$

3 Plenoptic Cameras

3.1 *Plenoptic Camera 1.0*

The traditional plenoptic camera is based on an array of microlenses at the image plane of the main camera lens, with the sensor placed one focal length behind the microlenses (see Fig. 1). The camera samples the radiance in front of the microlenses with a kernel, as shown in Fig. 2. Each microlens image is a vertical stack of samples in the (q,p) plane, capturing strictly the angular distribution of the radiance at the image plane.

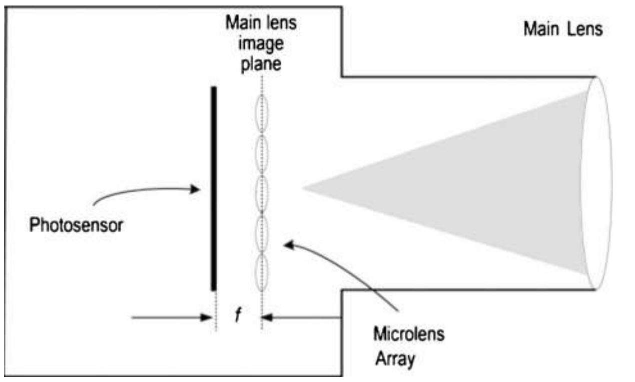

Fig. 1 Conventional plenoptic camera.

The main camera lens is focused one focal length in front of the microlenses. Consider one microlens. We will show that each pixel behind it measures the energy of rays that come at an angle specific to that pixel, and pass through a plane one focal length in front of that microlens.

To see this we compute the matrix $\mathbf{A}$ and $\mathbf{A}^{-1}$ for rays incident to a plane one focal length in front of a given microlens.

$$\mathbf{A} = \begin{bmatrix} 1 & f \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix}\begin{bmatrix} 1 & f \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & f \\ -\frac{1}{f} & 0 \end{bmatrix}, \quad \mathbf{A}^{-1} = \begin{bmatrix} 0 & -f \\ \frac{1}{f} & 0 \end{bmatrix}. \tag{5}$$

Consider Eq. (5). A pixel on the sensor responds approximately equally to rays from all angles. Therefore, its sampling kernel in ray space is represented as a vertical line as thick as the pixel in Fig. 3. Matrix $\mathbf{A}^{-1}$ maps this vertical line to a horizontal line because, due to the bottom right zero matrix element, input p does not influence output p. Moreover, the spatial size of that horizontal line (the amount sampled in the spatial domain) is limited only by

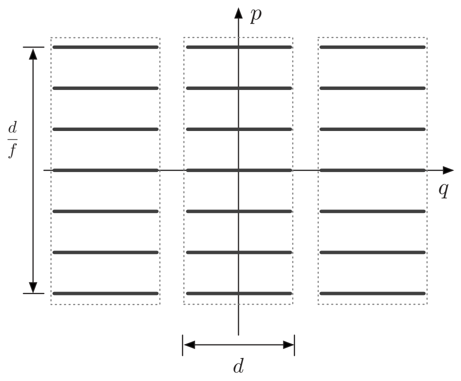

Fig. 2 Sampling of the radiance $r(q,p)$ by the microlens array represented in the 2-D (q,p) plane. Each pixel samples a single direction in the directional coordinate and samples a span of d (the microlens and microimage size) in the positional coordinate.

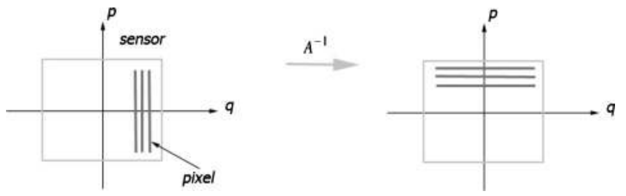

Fig. 3 Sampling pattern of one microlens in Plenoptic 1.0 camera.

the microlens' diameter. This large size of the sampling kernel is the reason for the low resolution of the plenoptic camera (1.0).

Images are rendered from the radiance captured by the traditional plenoptic camera by integrating all angular samples at a particular spatial point. However, each spatial point is sampled by a single microlens, so rendering involves integrating all of the pixels in each microimage. As designed, rendering from the plenoptic camera produces only 1 pixel per microlens, resulting in a rendered image with very low resolution. Even with 100,000 microlenses, the handheld plenoptic camera reported in Ref. 7 produces a final image of only 300×300 pixels.

An example of an image rendered with the plenoptic 1.0 algorithm is shown in Fig. 4. Note that in this case, the plenoptic data was captured with a plenoptic 2.0 camera. However, the rendered image is equivalent to what would be rendered from data captured with a plenoptic 1.0 camera.

3.2 *Plenoptic Camera 2.0*

As shown in Fig. 5, the focused plenoptic camera is based on an array of microlenses focused on the image plane of the main lens. Thus, each microlens captures a portion of the image formed by the main lens. We can think of the sensor as moved back, away from the main lens, so the image is formed some distance a in front of the microlenses. The microlenses serve as an array of real cameras, reimaging parts of that image onto the sensor.

In playing this role, each microlens forms a relay imaging system with the main camera lens. The position of each

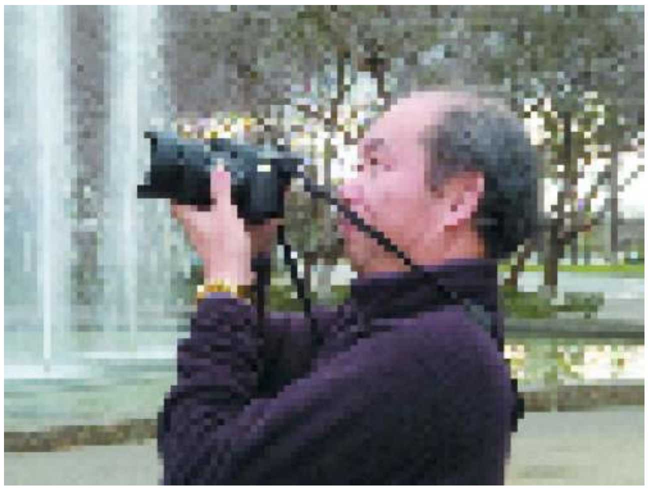

Fig. 4 Image rendered with traditional plenoptic rendering algorithm.

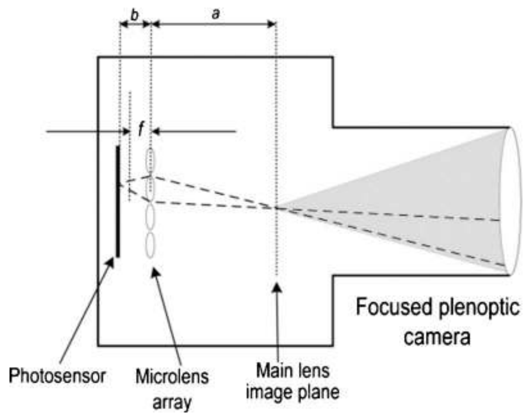

Fig. 5 Focused plenoptic camera.

microlens satisfies the lens equation, $1/a+1/b=1/f$, where a, b, and f are, respectively, the distance from the microlens to the main lens image plane, the distance from the microlens to the sensor, and the focal length of the microlens. In our setting, b is greater than f. A different setting is possible, where the main lens image is a virtual image formed behind the sensor. In this case, a would be negative and b would be less than f. We do not discuss this setting of the camera in this paper, but the treatment of such a case would be similar.

Next, we show that the focused plenoptic camera samples the radiance, as in Fig. 6. Each microlens image is a slanted stack of samples in the (q,p) plane, capturing both angular and positional distribution of the radiance at the image plane.

The total transfer matrix from that plane to the sensor is

$$\mathbf{A} = \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -\frac{b}{a} & 0 \\ -\frac{1}{f} & -\frac{a}{b} \end{bmatrix}. \tag{6}$$

The last equality holds due to focusing. Computing the inverse,

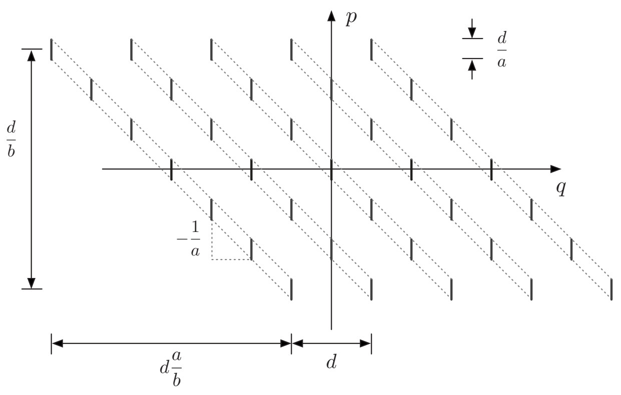

Fig. 6 Sampling of the radiance $r(q,p)$ by the microlens array of the focused plenoptic camera, represented in the 2-D (q,p) plane. The microlens aperture is given by d, and a and b are the spacing from the microlens plane to the image plane and from the microlens plane to the sensor, respectively.

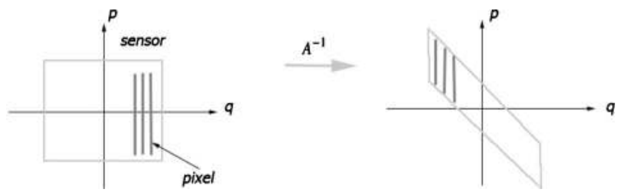

Fig. 7 Sampling pattern of one microlens in the focused plenoptic camera.

$$\mathbf{A}^{-1} = \begin{bmatrix} -\dfrac{a}{b} & 0 \\ \dfrac{1}{f} & -\dfrac{b}{a} \end{bmatrix}. \tag{7}$$

The important observation here is that due to the zero top right element, the sampling kernel for each pixel remains vertical in optical phase space after inverse mapping. As a result, sampling is done by a dense set of thin vertical kernels, and is decoupled from microlens size (see Fig. 7). Considering that minification for each microcamera is a/b, the high spatial resolution achieved is b/a times the sensor resolution, as shown in Fig. 6

An important result is that the spatioangular trade-off for the focused plenoptic camera is not fixed by the number of microlenses. Rather, the spatioangular trade-offs are determined by the optical geometry (a and b). To counter edge effects in the microimages, relatively large microlenses can be used.

As with the traditional plenoptic camera, images are rendered from radiance captured with the focused plenoptic camera by integrating the angular samples at every spatial point (see Fig. 8). Unlike the traditional plenoptic camera, however, the angular samples for a given spatial point are sampled by different microlenses. Therefore, rendering with the plenoptic 2.0 camera data involves integrating across microlens images rather than within microlens images. The difference in resolution can be seen in the example images rendered with the plenoptic 1.0 algorithm (Fig. 4) and the plenoptic 2.0 algorithm [Fig. 16(a) in Sec. 4.2]. Rendering with the plenoptic 2.0 camera is discussed in detail in the following sections.

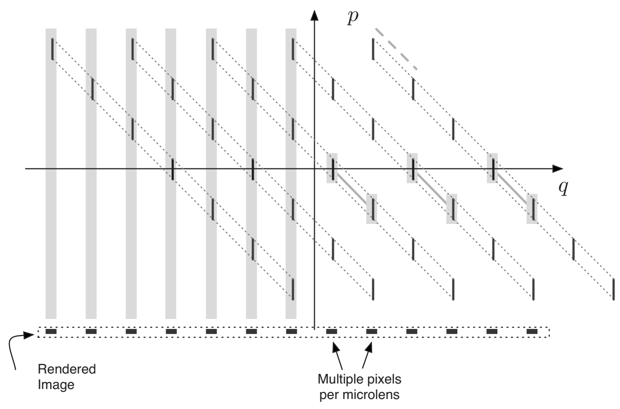

Fig. 8 Image rendering with the focused plenoptic camera. The left half of the figure shows rendering that integrates all directions associated with a given position. Note that integration takes place across microlens images. The right half of the figure shows rendering that only uses a single direction at each position. For the configuration shown, the rendered image has 2 pixels from each microimage.

3.3 *Discussion*

Although the basic designs of the plenoptic 1.0 and plenoptic 2.0 cameras are similar in having a single objective lens and a microlens array in front of the sensor, there are some fundamental differences that become apparent when considering optimal configurations for each. The plenoptic 1.0 camera has fixed spatioangular sampling with each microlens representing one spatial sample and a number of angular samples equal to the number of pixels in the microimage. Obtaining reasonable resolution of the rendered image therefore requires a high density of small microlenses. To meet the F-number-matching criterion,[15] we therefore require that the microlens array be placed close to the sensor, which requires removing the sensor cover glass or using an open sensor. Small microimages also have a large number of boundary pixels relative to interior pixels. Because of various edge effects (e.g., vignetting and noise), boundary pixels are discarded for rendering, resulting in less efficient use of the sensor real estate.

In the plenoptic 2.0 camera, spatioangular sampling is a function of the relay system parameters (a and b). The values of a and b would typically be chosen to produce 5 to 10 views (angular samples) of each spatial point. Spatial resolution of the rendered image is independent of the number of microlenses, enabling larger microlenses to be used and higher quality microimages to be captured. The F-number matching can be achieved, in this case, with larger spacing between the microlens array and the sensor (enabling direct placement of the microlens array on the cover glass of the sensor). Large microimages have a small number of boundary pixels relative to interior pixels, resulting in fewer pixels with edge effects and more efficient use of sensor real estate.

Because both the plenoptic 1.0 and 2.0 cameras follow the F-number-matching principle, they both have the same sensitivity—the same as a normal camera. However, since rendering in both cases combines multiple pixels, there is a reduction in noise proportional to the number of views averaged together at each spatial point in the rendered image.

4 Rendering with the Focused Plenoptic Camera

4.1 *Basic Focused Plenoptic Rendering*

The basic focused plenoptic ("plenoptic 2.0") rendering process is shown in Fig. 8 (right part). If we render with one angular sample for each location, we obtain M samples from each microlens image. In this example, $M=2$ and the rendered image has twice as many pixels as there are microlenses—twice the resolution of the traditional plenoptic camera. In general, the attainable resolution of an image rendered from the focused plenoptic camera depends on the depth of the scene. As derived in Ref. 10, the spatial resolution of a rendered image is b/a times the spatial resolution of the sensor. Resolution increases for image planes closer to the microlens plane (where b/a approaches unity), or equivalently, for planes in the scene that are closer to the main lens (in the foreground). Thus, image planes in the

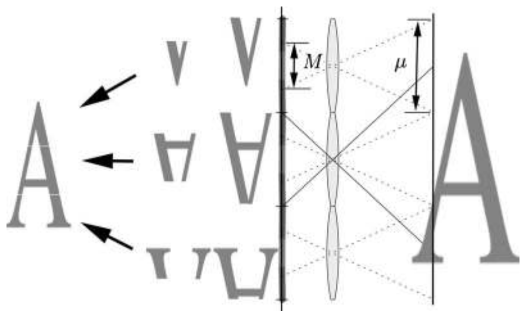

Fig. 9 Image capture geometry. Proper focusing is achieved when $\mu = M(a/b)$.

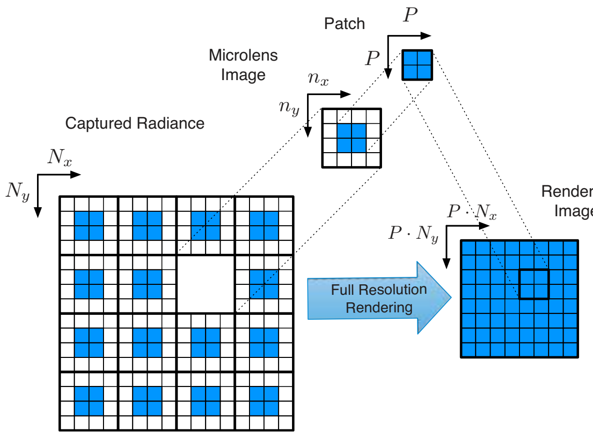

Fig. 10 Basic focused plenoptic rendering algorithm creates a final rendered image from $P \times P$ patches of each $n_x \times n_y$ microimage. With $N_x \times N_y$ microimages in the captured radiance, the final rendered image is $P \cdot N_x \times P \cdot N_y$.

foreground can be rendered with a higher resolution (larger number of pixels per microlens) than image planes in the background.

In the conventional plenoptic camera, all of the directions for a given spatial sample are contained within a single microimage and all of the spatial samples for a given direction are spread across microimages. In the focused plenoptic camera, the different views for a given spatial sample are spread across microimages.

As with the conventional plenoptic camera, an image is rendered from the focused plenoptic camera according to Eq. (4). And, as with the conventional plenoptic camera, we could implement single viewpoint rendering of Eq. (4) by evaluating $r(q,p)$ at some particular value of $p = p_0$, i.e., let $I(q) = r(q,p_0)$. In this case, however, we need to account for the fact that a single microimage samples over a range of q and a range of p. In particular, rather than selecting one spatial sample per microlens that corresponds to a single value of p to render the final image, at each microlens we extract a range of spatial samples corresponding to a range of directions (see Fig. 8, where $M = 2$).

An output image corresponding to a given view (a small range of angles) can be rendered from focused plenoptic radiance data by selecting a contiguous set of pixels (a patch) from each microimage and tiling all such patches together into the final image. The important parameter in this process is the pixel size of the patch to select from each microimage. Consider the image capture geometry in Fig. 9, where we wish to reconstruct the image on the main lens image plane with pieces taken from each microlens image. The distance between microlenses (the pitch of the microlens array) is μ. We divide the main lens image plane into $\mu \times \mu$ sections such that each such section maps to an $M \times M$ portion of a microlens image. The main lens image can be reconstructed by putting together those $M \times M$ portions ("patches"). Strictly speaking, we require a negative value of M to "flip" the patches to their correct orientation before assembling them.

However, there is an alternative interpretation of the image capture geometry. Namely, for a given rendering pitch (the patch size M with which we render), there is an image plane at distance a in front of the microlenses that will satisfy $\mu = M(a/b)$. That plane is "in focus" in the sense that an image picked up from it will be rendered with no artifacts. The patches of that exact size tile together perfectly. In other words, the rendered image is "focused" only for that plane by the choice of the pitch, i.e., the patch size M.

The basic focused plenoptic rendering algorithm is shown schematically in Fig. 10. Intuitively, the algorithm operates as follows. We specify the pitch (defined by the number of pixels per microimage) and select squares of that size from each microlens image. The final image is rendered by tiling the selected squares together. Choosing one pitch or another puts different world planes "in focus." In other words, patches match each other perfectly only for one image plane behind the main lens. By the lens equation, this corresponds to a given depth in the real world. Thus, a different patch size would correspond to a different depth. In other words, the equivalent of refocusing is accomplished in the plenoptic 2.0 camera data through the choice of the patch size (the pitch) in the basic focused plenoptic rendering algorithm. This could be called the "full-resolution" rendering principle, and it is underlying idea in all focused plenoptic rendering methods.

Analysis of rendering methods is illustrated with an example of an image captured with our plenoptic camera, a crop from which is shown in Fig. 11.

4.1.1 *Artifacts in basic full-resolution rendering*

Observe that the basic focused plenoptic rendering process can produce strong artifacts, as we can see in the background in Fig. 12.

These artifacts result because the pitch necessary to produce artifact-free full-resolution rendering is dependent on the depth in the scene. That is, different parts of a scene will require different patch sizes to be properly rendered.

The relationship between focusing and patch size also explains the artifacts that can arise when rendering a scene that has different depths. In particular, if we use a fixed patch size for a scene that has differing depths, there will be parts of the scene where the patch size is not the correct one to bring that part of the scene into focus. In those parts of the scene, the patches will not match at their boundaries. Unless that region of the scene is smooth, obvious artifacts at the microimage boundaries will be apparent. Figures 13

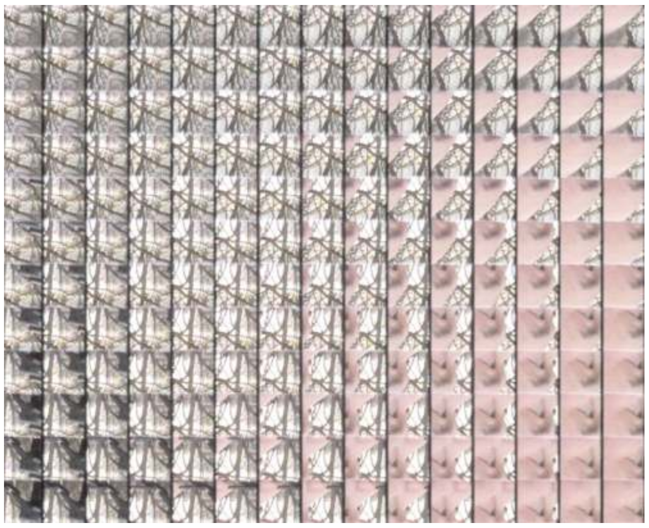

Fig. 11 Small crop from the input radiance data for our photographer rendering examples.

and 14 illustrate the artifacts that arise when the choice of *M* is, respectively, too small or too large for rendering a scene.

These artifacts are indications that the simple rendering approach described in the preceding is not well-matched to the particular task of rendering a focused plenoptic data. Modifying the approach to produce artifact-free images depends on the particular task to be accomplished. Next we will consider depth-based rendering and blended rendering as basic approaches to reduce artifacts.

4.2 *Depth-Based Rendering*

Depth-dependence of focus for the plenoptic 2.0 camera suggests one approach for artifact-free rendering—namely, to use depth information from the scene so that different (and correct) patch sizes can be used for rendering different parts of the scene. We develop an algorithm for extracting

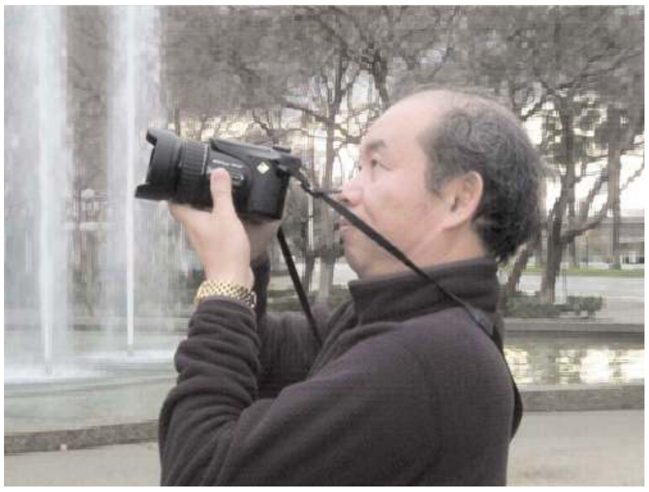

Fig. 12 Out-of-focus rendering of our photographer image. The pitch for the background is too large, resulting in artifacts.

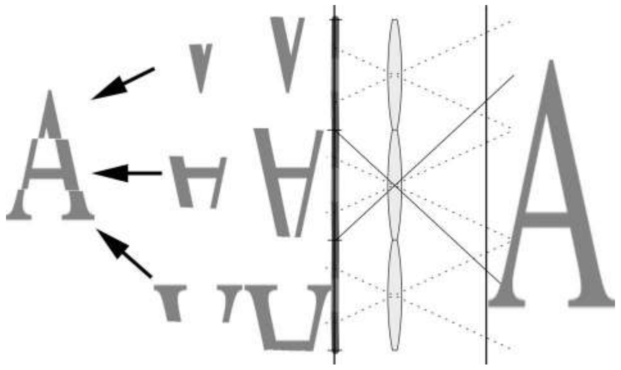

Fig. 13 Out-of-focus rendering regions of the scene. Here, our chosen pitch is too small for the scene being rendered, resulting in pixellation artifacts. Choosing a larger patch size would bring the image into focus, as in Fig. 9.

the depth information of a scene directly from a plenoptic image and then using this depth information to render an artifact-free image.

To determine the correct patch sizes across the rendered image, we take advantage of another important property of the focused plenoptic data, namely, that microimages capture overlapping regions of the scene. The patch size also determines the spacing between patches—the correct patch size for focusing a region of a scene will also be the spacing at which neighboring microimages will overlap. That is, the same portion of the scene that is captured by different microlenses must be rendered to the same position in the output image. This matching condition suggests the following two-pass algorithm for rendering.

1. For each microlens, determine the patch size that results in the best match with all of its neighbors.
2. Render the final image with the saved pitch value for each microlens.

Determining the minification that provides the best match between two microlens images is essentially an image registration problem. We can exploit several aspects of our system to streamline this process. First, the microimages in the captured radiance are precisely determined by

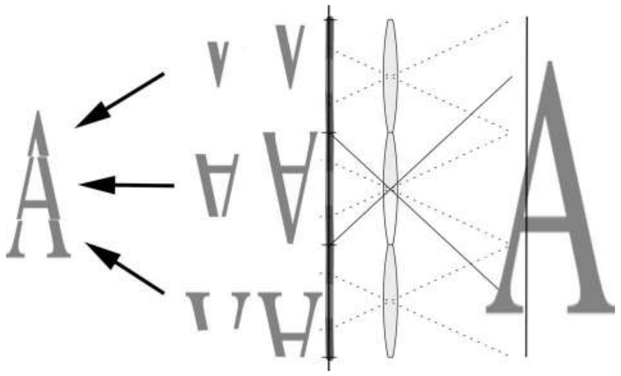

Fig. 14 Out-of-focus rendering regions of the scene. Here, our chosen pitch is too large for the scene being rendered, resulting in "screen door" artifacts. Choosing a smaller patch size would bring the image into focus, as in Fig. 9.

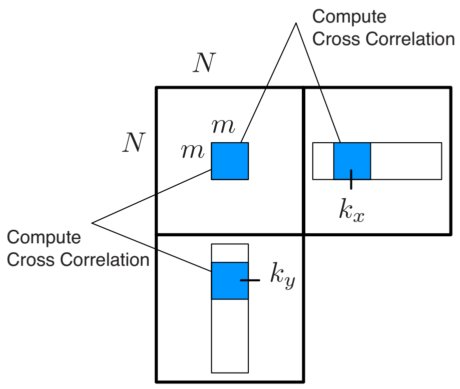

Fig. 15 Depth estimation algorithm.

the microlens geometry, and precisely aligned. Thus, the difference between neighboring microimages along the horizontal and vertical axes of the lens array will be only horizontal and vertical translations, respectively. Moreover, based on the optical design of the camera, there are bounds on how large the shift between microlens images can be. These characteristics of the captured radiance greatly simplify the depth estimation algorithm.

An algorithm for estimating depth is given shortly. The algorithm produces an array of pitch values that we can subsequently use in rendering the final image. The operation of the algorithm is shown graphically in Fig. 15.

4.2.1 *Depth estimation algorithm*

```
1. For each $N\times N$ microlens image
   a. Select an $m\times m$ window from the center of that microlens image.
   b. For $k=-N+m/2$ to $k=N-m/2$
      i. Compute the cross-correlation between the $m\times m$ window and a corresponding window centered at $k_x$ in the neighboring microlens image along the $x$ axis.
      ii. Record the value of $k_x$ with the best correlation.
      iii. Compute the cross-correlation between the $m\times m$ window and a corresponding window centered at $k_y$ in the neighboring microlens image along the $y$ axis.
      iv. Record the value of $k_y$ with the best correlation.
   c. Record value of $k$ equal to average of $k_x$ on left and right boundaries and $k_y$ on top and bottom boundaries
2. Return the array of recorded values of $k$.
```

The depth estimation algorithm produces an array of patch size values that we can subsequently use in rendering the final image. To render the final image, we modify the basic rendering algorithm so that rather than using a fixed pitch value, we look up the precomputed value for the given microlens.

By extracting depth information as already described and then rendering the image with different magnification at each microlens, we produce the image in Fig. 16. Note that regions of the image at all depths are rendered essentially artifact free.

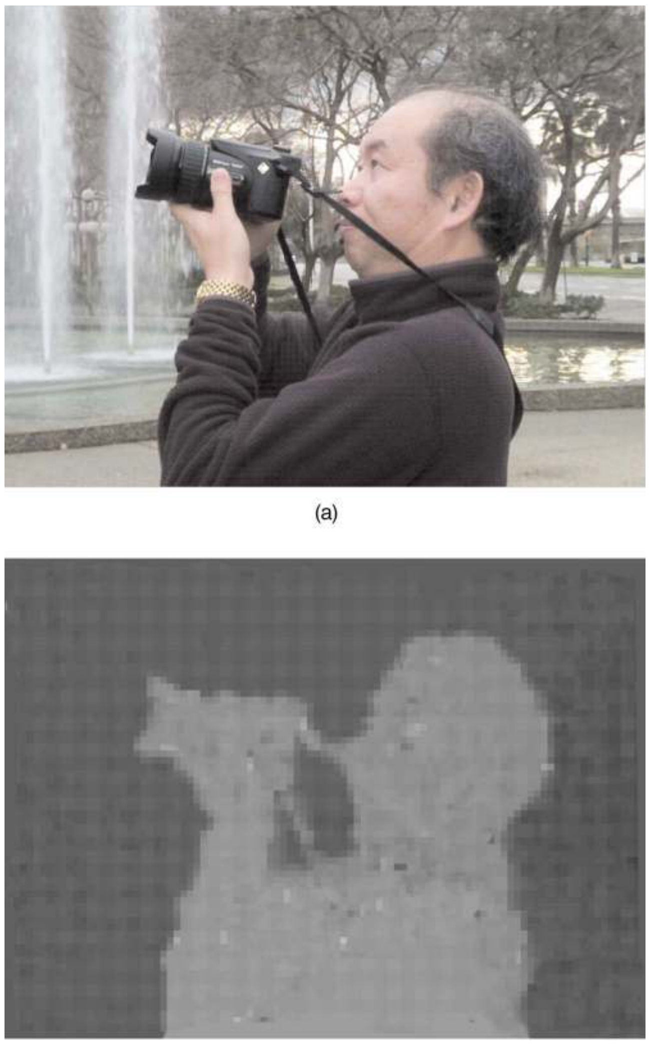

Fig. 16 Image rendered using depth correction: (a) rendered image and (b) estimated depth. Lighter regions correspond to the foreground (and larger pitch values).

4.3 *Rendering with Blending*

Rendering for finite size apertures involves integration in the angular dimensions. Intuitively, this integration process means we must average together ("blend") the same spatial point across different microlens images (see Fig. 8, left). A direct comparison of basic rendering and rendering with blending is shown in Fig. 17. Note the reduction of artifacts and the effect of "focusing" in the blended rendering version of the image.

Although rendering with depth information enables artifact-free rendering of a single view of the scene, the result is an "all in-focus" image, which precludes depth-of-field and refocusing effects. Obtaining these effects requires combining multiple views of a scene, that is, integrating over a range of views as specified in Eq. (4).

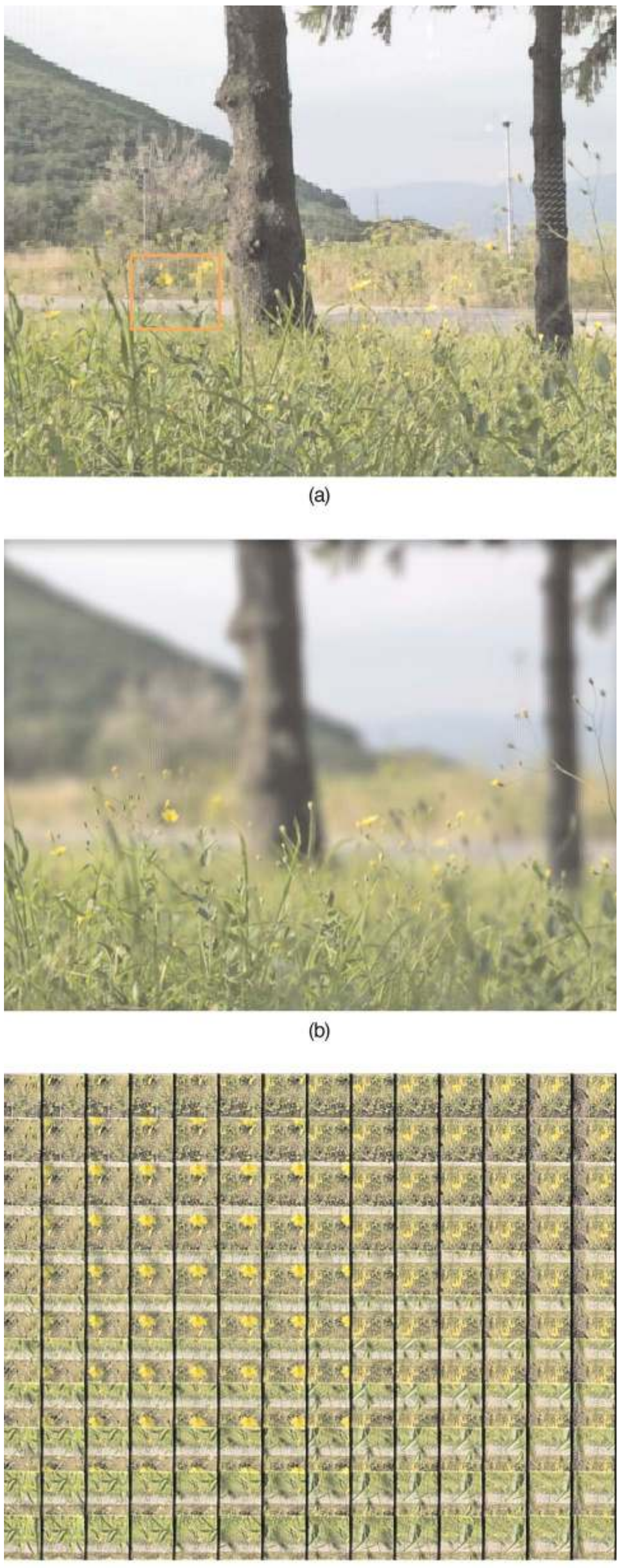

Fig. 17 Comparison of (a) basic rendering and (b) rendering with blending. In (a) the image is rendered using the basic algorithm. Note the presence of artifacts due to nonmatching patches at large depth. In (b), the image is rendered using our blending algorithm. Note that the artifacts are now suppressed and that the out-of-focus regions appear properly blurred. (c) Small crop from the 39-megapixel raw image that was used to render (a) and (b).

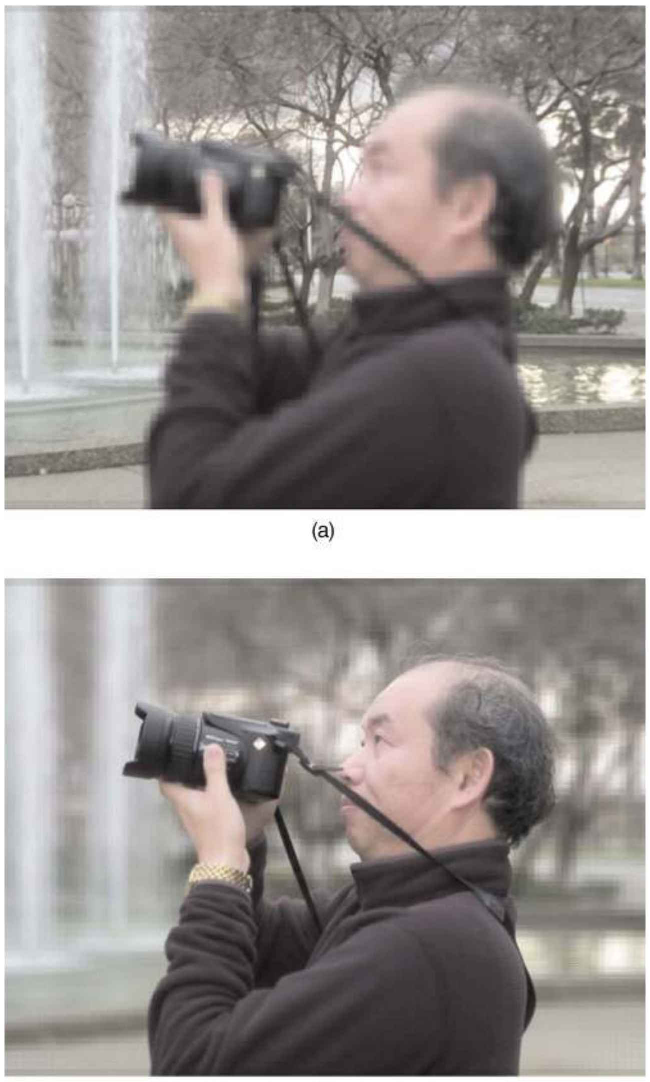

Fig. 18 Comparison of rendering with blending: (a) small pitch and (b) large pitch. Image (a) is rendered using the blending algorithm with small pitch (7 pixels). Note that the background is in focus and the foreground is out of focus. Image (b) is rendered using the blending algorithm with large pitch (10 pixels). Note that the background is out of focus, and the foreground is in focus.

To accomplish the integration over p in Eq. (4) for the focused plenoptic camera, we must average together ("blend") the same spatial point across microlens images. (Averaging across microlens images is in contrast to the conventional plenoptic camera that averages within microlens images). For microlenses spaced μ apart and a patch size of M, the pixels that must be averaged together to a given output pixel will be separated by distance $(\mu - M)$ in the captured raw image.

A comparison of two blended rendering results is shown in Fig. 18. From the phase space diagram Fig. 8 of the process we can see that for small mismatches of the slope of the integration direction, blending should be sufficient to produce a smooth blur. (Note that the slope is vertical in the figure, but since different depths are related by shear in ray space, in general slope is nonvertical, defined by M). For larger mismatches, we might not have enough views and we may instead see ghostlike artifacts due to features being repeated across multiple patches.

4.3.1 *Focusing*

An important characteristic of the plenoptic 2.0 camera was already mentioned: refocusing is accomplished through choice of the pitch size in the full-resolution rendering algorithm. That pitch is related to shear in ray space. Conversely, the value of the pitch determines the plane (the depth) in the image that is in focus. Images captured with the focused plenoptic camera can therefore refocus rendered images by choosing different values of the pitch size. When multiple microlens images are integrated (blended), the out-of-focus regions appear blurred, as would be expected. Because the microlenses have a very small aperture, there is a significant depth of field, i.e., portions of the scene that are in focus for a given value of the pitch size will extend for a large depth.

5 Camera Prototype

In this section, we present details of our current physical camera along with rationale for its design. The camera is medium format with an 80-mm lens and a 39-Mpixel P45 digital back from Phase One. To produce square microimages that tile together with little loss of sensor space we introduced a square aperture at the original aperture location. Since the microlenses are focused on the image rather than the main lens aperture, there is a blur of approximately 3 pixels at the boundary of each microimage, which pixels cannot be used in rendering. Also, we mounted the lens on the camera with a 13-mm extension tube, which provides the needed spacing a.

The microlens array is custom made by Leister Axetris. The microlenses have focal length of 1.5 mm so that they can be placed directly on the cover glass of the sensor. We are able to provide additional spacing of up to 0.5 mm to enable fine-tuning of the microlens focus. The pitch of the microlenses is 500 μm with precision 1 μm. The sensor pixels are 6.8 μm. The value of $b \approx 1.6$ mm was estimated with precision 0.1 mm from known sensor parameters and independently from the microlens images at different F-numbers. Captured plenoptic images are 7308×5494 pixels.

Our microlens apertures are circular, with 100-μm diameters, and are formed by the black chromium mask deposited on the microlenses. This small aperture provides a larger depth of field for the microimages, enabling a greater range for refocusing. While the small apertures extend depth of field and make microlenses diffraction limited, they also introduce high F-number and, associated with it, diffraction blur and longer exposure times.

The relatively large pitch of the microlenses is chosen in order to match the F-number of the main camera lens, which can be as low as $F/3$. This large pitch is needed because our microlenses are at large distance (1.6 mm) from the sensor, defined by the cover glass.

6 Conclusion

We showed that a modified, 2.0 version of the plenoptic camera can be used to capture higher spatial resolution radiance data. By analyzing the ray space model of the camera we derived an algorithm for synthesizing all-in-focus images from different viewpoints, and a method for reducing the artifacts that are well known for this type of imaging. Another algorithm, for synthesizing shallow-depth-of-field images and refocusing on different depths was also presented. All these algorithms are implemented on the GPU, as shaders in OpenGL, and used to render refocused views at interactive rates at full screen resolution.

Looking into the future, our ongoing work indicates that very promising results are possible. We can improve the quality of rendering by applying more sophisticated (but still interactive) algorithms for blending and filtering the radiance data. Such algorithms can produce perfect-quality refocused images from even sparser, in the angular dimensions, input data captured with our camera. Such data have higher spatial resolution that could reach 5 to 10 Mpixels with our 39-Mpixel input, meeting the needs of modern photographers, while being refocusable and generating different views.

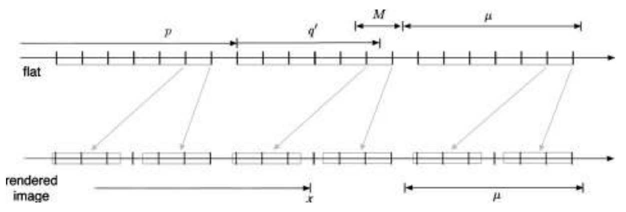

Fig. 19 Basic rendering directly from the plenoptic image, no blending. Note the magnification a/b.

Appendix A: GPU Implementation of Basic Rendering

Our GLSL implementations of lightfield rendering are carried out completely in OpenGL fragment shaders. Although our examples were implemented in Python (via the PyOpenGL library[16]), interfaces to OpenGL are similar across languages. Other than providing support for creating and installing the shader, the main functionality provided by OpenGL were the following:

1. Read in the plenoptic image data (from a stored image).
2. Serialize the lightfield data to a format suitable for OpenGL.
3. Create a 2-D OpenGL texture object for the plenoptic image data.
4. Define the texture in OpenGL, using the serialized image data.

Rendering the plenoptic image data is then accomplished by rendering the installed texture, using our custom shader. To explain the operation of the basic shader, we discuss some of the details of the optical image capture geometry as interpreted by OpenGL.

Consider the rendering geometry shown in Fig. 19. Let μ be the size of one microlens image, measured in pixels. In this example, $\mu = 7$. For a given point x in the output image, we must find the corresponding sampling point on the flat. To accomplish this process, we must perform two computations. First, given x, we must determine from which microlens we will render x. Second, we must determine where in the region of size M the point x lies.

To compute which microlens corresponds to x, we take the integer part of x divided by μ, which gives us the index of the microlens. In other words, this number (let us call it p) is given by

$$p = \left\lfloor \frac{x}{\mu} \right\rfloor. \tag{8}$$

The pixel location of the beginning of that microlens in the flat is then given by multiplying the microlens number by the size of one microlens image, i.e., $p\mu$.

Next, we compute the offset within the region of size M corresponding to x. To do this, we compute the difference between x and the start of microlens (p). This will give the offset in the rendered image, but we need the offset in the flat. Since we are scaling a region of size M in the flat to a region of size μ in the final image, the offset must be scaled by M/μ. That is, the offset q is given by

$$q = \left(x - \left\lfloor \frac{x}{\mu} \right\rfloor \mu \right) \frac{M}{\mu} = \left(\frac{x}{\mu} - p \right) M. \tag{9}$$

Finally, we must make one more adjustment. We want the center of the $M \times M$ region of the flat to render to the center of the corresponding region of the final image. The preceding formulas will map the left edge of the microlens image to the left edge of the corresponding region in the rendered image. To accomplish this centering, we add an offset of $(\mu - M)/2$ to q:

$$q' = q + \frac{\mu - M}{2} = \left(\frac{x}{\mu} - p \right) M + \frac{\mu - M}{2}. \tag{10}$$

Combining Eqs. (8) and (10), the corresponding point in the flat for a given point x in the output image is given by $f(x)$, where

$$f(x) = p\mu + q'. \tag{11}$$

The GLSL fragment shader code to carry out this algorithm is as follows:

```
uniform sampler2DRect flat;   //
Plenoptic image

uniofrm float M, mu;

uniform float XOffset;

uniform float YOffset;

void main()

{

  vec2
x_mu=gl_TexCoord[0].st/mu;      //
x/μ

  vec2 p=floor(x_mu);             // p
=⌊x/μ⌋

  vec2 q=(x_mu−p)*M;              //
(x/μ−p)M

  vec2
qp=q+0.5*(mu−M);              // q′=q
+(μ−M)/2

  vec2
offset=vec2(XOffset,YOffset)*(mu
−M);

  vec2
fx=p*mu+qp+offset;            // f(x)
=pμ+q′

  gl_FragColor
=texture2DRect(flat,fx);

}
```

The plenoptic image, as well as the values for μ and M, are provided by the user program via uniform variables. The shader program computes q, q', and $f(x)$ as q, qp, and fx, respectively. Changing the viewpoint of a synthesized image is enabled by adding user-specified offsets, XOffset and XOffset (both equal to 0 by default), to the coordinates fx. Finally, we look up the value of the pixel in the flat and assign that value to the requested fragment color.

Acknowledgments

The authors gratefully acknowledge the support from Adobe Systems, Inc., for this work. The authors also thank the anonymous reviewers for their helpful comments.

References

1. F. Ives, "Parallax sterogram and the process of making sameU.S. Patent No. 725,567 (1903).
2. G. Lippmann, "Epreuves reversibles. Photographies integrales," *Acad. Sci.* **146**, 446–451 (Mar. 1908).
3. T. Adelson and J. Bergen, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing*, MIT Press, Cambridge, MA (1991).
4. T. Adelson and J. Wang, "Single lens stereo with a plenoptic camera," *IEEE Trans. Pattern Anal. Mach. Intell.* **14**, 99–106 (1992).
5. M. Levoy and P. Hanrahan, "Light field rendering," *ACM Trans. Graphics* **23**, 31–42 (1996).
6. S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," *ACM Trans. Graphics* **23**, 43–54 (1996).
7. R. Ng, M. Levoy, M. Bredif, G. Duval, M. Horowitz, and P. Hanrahan, "Light field photography with a hand-held plenoptic camera," Computer Science Technical Report CSTR (Jan 2005).
8. R. Ng, "Fourier slice photography," *ACM Trans. Graphics* **24**(3), 735–744 (2005).
9. A. Lumsdaine and T. Georgiev, "Full resolution lightfield rendering," Tech. Rep., Adobe Systems (2008).
10. A. Lumsdaine and T. Georgiev, "The focused plenoptic camera," in *Proc. Int. Conf. on Computational Photography*, pp. 1-11, Stanford University (2009).
11. J. Tanida, T. Kumagai, K. Yamada, and S. Miyatake, "Thin observation module by bound optics (tombo): concept and experimental verification," *Appl. Opt.* **40**(11), 1806–1813 (Jan. 2001).
12. K. Fife, A. El Gamal, and H.-S. Wong, "A 3d multi-aperture image sensor architecture," in *Proc. Custom Integrated Circuits Conf.*, pp. 281–284, IEEE (2006).
13. K. Fife, A. El Gamal, and H.-S. P. Wong, "A 3mpixel multi-aperture image sensor with 0.7 um pixels in 0.11 um CMOS," in *IEEE ISSCC Digest of Technical Papers* pp. 48–49 (2008).
14. M. Christensen, M. Haney, D. Rajan, S. Douglas, and S. Wood, "Panoptes: a thin agile multi-resolution imaging sensor," in *Proc. Government Microcircuit Applications and Critical Technology Conf. (GOMACTech-05)*, pp. 1–8, IEEE (Jan. 2005).
15. R. Ng, "Digital light field photography," PhD thesis, Stanford University, Stanford, CA (2006).
16. "PyOpenGL," http://pyopengl.sourceforge.net/, http://pyopengl.sourceforge.net/.

Todor Georgiev is a senior research scientist with Adobe Systems, working closely with the Photoshop group. Having extensive background in theoretical physics, he concentrates on applications of mathematical methods taken from physics to image processing, graphics, and vision. He is the author of the award-winning Healing Brush tool in Photoshop (2002), the method better known as Poisson image editing. He has published a number of papers on different applications of mathematical methods to image processing and vision. Currently he is focusing on a range of theoretical and practical ideas related to optics for plenoptic cameras and capture/manipulations of the optical field. He holds 25 U.S. and International patents.

Andrew Lumsdaine received his PhD degree in electrical engineering and computer science from the Massachusetts Institute of Technology in 1992. He is currently a professor of computer science at Indiana University, where he is also directs the Open Systems Laboratory. His research interests include computational science and engineering, parallel and distributed computing, mathematical software, numerical analysis, and radiance photography. He was a recipient of the National Science Foundation (NSF) CAREER award and is a member of the IEEE, the IEEE Computer Society, the ACM, and SIAM.