Computational Image Formation Based on Surface Light Fields

Georgi N. Chunev

June 4, 2015

Abstract

The substitution of optical processes with computation has been a core goal of computational photography. With the use of light field acquisition and rendering techniques, post-capture refocusing and view selection have been made possible. The existing techniques computationally model how a physical camera would refocus the captured set of rays for some user controlled camera parameters. Nonetheless, when it comes to focusing effects, it is optics itself that provides an indirect and limited way of specifying how different parts of the scene should be refocused. On one hand, in an optical system, focusing can be influenced by many parameters (camera distance, f/stop, and main lens focal length), which makes the choice of focusing indirect and often unintuitive. On the other hand, an optical system specifies focusing only relative to the geometry of the scene, but independently of other content relevant to the final image (textures, materials, salient features, etc.). Alternatively, a semantic specification of focusing can provide important cognitive cues that can both steer attention and enhance artistic expression. In relation to this, the need for generalizing depth of field effects, in a way that allows direct, per-point focusing specification, has been identified. Still, prior literature has considered only non-physical solutions - e.g., by substituting wide-aperture defocus with variable-width blur. Such approaches are fundamentally flawed, as they cannot correctly simulate light propagation effects (e.g., attenuation, partial occlusion, shaped bokeh, positive and negative defocus). Additionally, none of the existing techniques have managed to provide users with the ability to directly make focusing choices at the granularity of individual scene points, and the need for interactivity in the resulting implementations has not been specifically addressed.

The main contribution of the work proposed for this thesis will be a computational image formation model, which allows for physically-accurate, generalized focusing effects. The model incorporates a novel technique for specifying custom ray transmission distances inside a modeled wide-aperture camera. The proposed technique requires explicit knowledge of where in the scene the radiance, carried by a given ray, originated from. This additional information can be stored separately from a standard light field, or can naturally be encoded in a light field parameterized on the scene surfaces. When both the scene and the camera are static, this surface light field can also be reparameterized using pairs of points on the scene's surfaces and the main lens aperture plane, giving rise to a previously unexplored light field parameterization - *surface-aperture light fields*. The associated computational image formation techniques can be adapted to real-time rendering and can find practical applications in a variety of fields, which will be explored for this thesis. In relation to standard light field rendering, passing though the scene's surface light fields can allow for both generalized focusing effects and artifact-free renditions, as view interpolation and extrapolation become straightforward.

The remaining work on this project is largely related to implementing and testing the proposed image formation model in three proposed application areas. First, a virtual scene setting will be best suited for establishing ground truth implementations and for preliminary tests with interactive implementations. I am considering modifying Blender Cycles for this purpose. Blender is also very well suited for implementing offset focusing using texture paint techniques, i.e. a *focus brush*. Second, scanned or multi-view reconstructed scenes would be an easy way to test the renderers on real-world data. Based on multi-view photography, commercial 3D photo-modeling software like Autodesk ReCap/ReCap 360 could provide quality scene reconstructions and calibration data, and mapping a surface light field to the resulting models will require no more than resampling the input imagery. Finally, imagery from Lytro Illum is well

suited for implementing tests with real-world plenoptic camera data. This is also a good setting for testing disparity-space implementations of the proposed renderers.

1 Introduction

A large body of work on light field acquisition and rendering [34, 20] has been developed in the last few decades. Among the practical applications of light fields that have been demonstrated are: synthetic photography [41] (i.e. post-capture refocusing and view selection); single-shot rich image acquisition and processing [17] (e.g., HDR imaging); and single-shot 3D scene reconstruction [4, 19, 58]. On top of this, light fields can now be captured with portable devices like hand-held cameras [42, 41, 46, 47] and dens camera arrays [59, 57], and most of the algorithms for processing light field imagery have been adapted to run interactively even on off-the-shelf graphical processing units [36]. Because of this, and because of how naturally light fields describe all of the rays entering a wide-aperture camera, light field imaging is here to stay and is driving the next revolution in photography.

More formally, a light field is any 4D parameterization of the radiance-carrying light rays that pass through a given 2D manifold in 3D space [33]. For instance, a light field can be used to describe the light rays that leave the surface of a scene object, or the ones that pass through a given aperture in the space away from the scene, or even the ones that intersect a given image plane inside a modeled camera. In fact, the entire image formation pipeline of mapping surface radiance values to sensor irradiance values can be formulated in terms of light fields and geometric transformations applied to them [35], which is what makes light fields so practical. Still, given that light fields are a form of precomputed radiance transfer, it is important to consider what limitations may arise from this precomputation.

In theory, a single light field is only a subset of the more general 5D plenoptic function [1]. A plenoptic function stores the radiance of all rays passing through all points in a section of 3D space, and thus captures all visual information potentially available to an observer of that scene [1]. Aside from some special cases, like modeling radiance from purely volumetric parts of the scene (smoke, gasses, etc.), in most practical cases the plenoptic function is highly redundant, which allows for lower dimensional plenoptic subsets (e.g. light fields) to be used. In the case of light field photography [41, 37] the entire plenoptic function in the cavity of a camera can be reconstructed from a single continuous light field, for instance the one covering the aperture of the desired virtual camera. This light field can subsequently be filtered and transformed to different focal surfaces in the space inside the camera, which allows for the computational synthesis of all standard photographic effects (focus, change of view, DoF). Furthermore, when considering the volume occupied by a scene, redundancies in the plenoptic function also exist, whenever not all scene points actually contribute to changes in radiance. This observation can be exploited and can potentially allow us to avoid having to store the full 5D plenoptic function in practice. If we assume that radiance transmission follows standard ray optics, then no new visual information will be added in the free space between or away from the scene points that emit or reflect radiance. Considering that the relevant scene points can always be described as an arrangement of disjoint surfaces that cut through the volume of the scene, we propose to use the union of the 4D surface light fields [3] (SLFs) of the scene as a very minimalistic, and yet integral representation of its 5D plenoptic function.

But this is not all. The SLF representation has an important advantage over storing the plenoptic function itself, as explicit positional information about the radiance sources in the scene is encoded in the SLF. On one hand, this allows for a clear separation of the spatial and angular components of an SLF, which, in turn, allows for view interpolation and extrapolation, needed to avoid plenoptic artifacts. On the other hand, the positional information for the viewed scene points is key for quantifying and, subsequently, directly controlling how different parts of the scene will be focused by a modeled lens.

The development of algorithms that allow for direct specification of focusing is of great importance to computational photography and will be the core topic of this work. My primary goal will be to develop and implement a generalization of the thin-lens image formation model that can provide explicit control over the focusing of individual scene points. Prior work on generalized DoF effects [30, 29, 40, 25] has not considered such physically-based approaches, and fails to capture a variety of light propagation effects

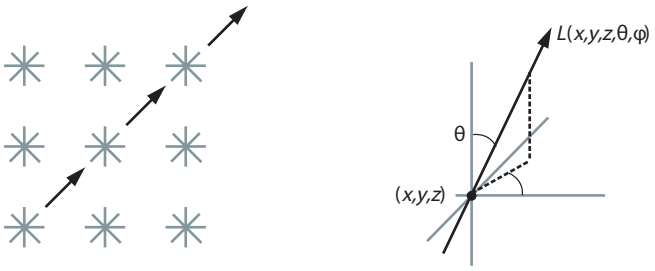

Figure 1: The 5D plenoptic function parameterizes all rays in 3D space and measures the radiance along them. The left sub-figure shows radiant rays leaving every point in space. A ray passing though free space is assumed to carry constant radiance, which creates redundancies in the plenoptic function. The right sub-figure shows a single ray parameterized by a point and a direction.

(e.g., attenuation, occlusion, shaped bokeh, positive and negative defocus). My idea is to use the position of each directly viewed scene point as an additional parameter in a physically-based, wide-aperture image formation model. SLFs are a natural way to store the relevant data (radiance, positions, and directions). While prior work on SLFs has concentrated on pinhole view synthesis and material modeling [60, 8], I will be the first one to consider their use for generalized image formation (wide-aperture effects, generalized DoF, etc.). Since I will be defining a computational camera model, a lot of my work will still be motivated by standard light field photography techniques, based on aperture light fields. In relation to this, I will consider the use of a new, combined light field parameterization that connects scene surface points to aperture points.

The results from this proposed research will be of practical interest in a variety of settings - from purely synthetic scene rendering, where all input data is readily available; to the rendering of scenes that were scanned, or reconstructed from dense, high-resolution, multi-view imagery; to plenoptic camera light field rendering, where scene reconstruction and view calibration are ongoing research efforts by the computer vision community [58].

This proposal starts with a brief introduction to plenoptic theory and terminology in Section 2. This is followed by a larger discussion on light filed capture and rendering in Section 3. More importantly, in Subsection 3.3, some open problems in standard light field rendering are discussed in detail. For each problem, I also suggest a possible solution based on surface light fields. An expanded discussion on surface light fields and rendering from them is presented in Section 4, where I also sketch out the basis of the image formation model that I am proposing. Three case studies for using the proposed renderers are also discussed in 4.3. The main contributions of the proposed work are reiterated in Section 5.

1.1 Thesis Statement

Knowing the positions, from which radiance originates in a scene, allows for the implementation of more general and less artifact-prone refocusing algorithms than what is achievable just with 4D light field data captured away from that scene.

2 Background on Plenoptics

2.1 The Plenoptic Function

The concept of the plenoptic function [1] was introduced as a way of formally describing the visual information potentially available to an observer of a given scene. In brief, the general 7D plenoptic function measures the light intensity, L, at wavelength, λ, that leaves a given point in the scene (x, y, z), in a given direction (θ, φ), at a given time, t:

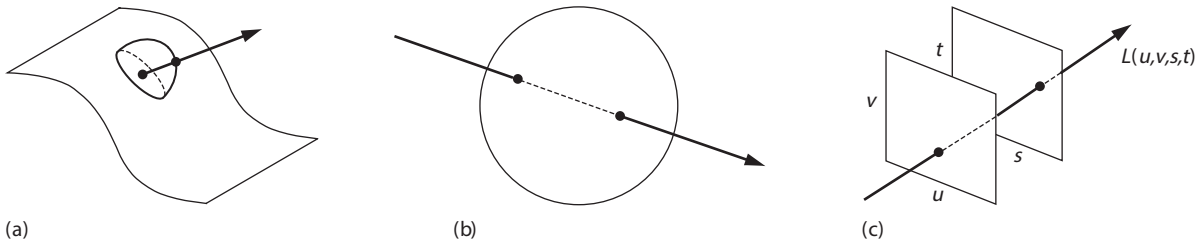

Figure 2: Common light field parameterizations: (a) point-angle parameterization; (b) sphere parameterization; (c) two-plane parameterization.

$$P = L(x, y, z, \theta, \varphi, \lambda, t) \tag{1}$$

For analytical purposes, the color dimension gets dropped. And, for static scenes, time is not relevant. This leaves a 5D plenoptic function, as is illustrated in Figure 1 (taken from [33]). Even as such, the plenoptic function contains redundancies, under the assumption that radiance is conserved in free space. How to take advantage of these redundancies even around non-empty parts of the scene is a topic addressed further in this proposed work.

The main reason to store a plenoptic function, or any subset of it, is that each spatial sample $L(p_0, \theta, \varphi)$ stores all of the rays needed to form a given pinhole image of the scene. Actually forming this image corresponds to taking a pencil of rays from the plenoptic function evaluated at the desired pupil position $p_0 = (x_0, y_0, z_0)$ and projecting these rays onto a 2D image surface. Considering this image formation process, it is easy to see that the plenoptic function captures all possible views of the scene. Wide-aperture views can also be synthesized from plenoptic data simply by taking a slice through the plenoptic function at the set of positions that corresponds to the desired wide-aperture surface $s(u, v)$. The resulting 4D plenoptic subset $L(s(u, v), \theta, \varphi)$ is more generally known as a light field [34] and is of particular interest in computational photography and will be relevant to the work proposed for this thesis.

2.2 Light Fields

Light fields [34], and their equivalents, lumigraphs [20], were first described as 4D plenoptic subsets that can be used to model all non-redundant information for the plenoptic function in free space. The important observation was that, assuming radiance conservation, radiance does not change along a given ray when that ray is traveling through free space. Because of this, capturing rays only at the boundary of a given region of empty space is sufficient to represent the full plenoptic function for that region. To parameterize all of the relevant rays one could pick pairs of distinct points on a surface that surrounds the region of interest and connect each pair with a ray. In practice, not all of these rays are needed. For instance, in computational photography, the region of free space that is of interest is the cavity of a modeled camera. In this case, the relevant rays are only the ones that pass through the main lens aperture and hit the sensor. A more suitable parametrization for these rays would be to use their intersection points with two planes - one corresponding to the camera's aperture and one to the camera's sensor. This is commonly referred to as the two-plane parametrization:

$$LF \quad = \quad L(u, v, s, t) \tag{2}$$

To achieve different photographic effects, the sensor position may be subsequently altered [23], which would require re-parameterizing the light field. In cases like this, it may be better to treat each surface (the

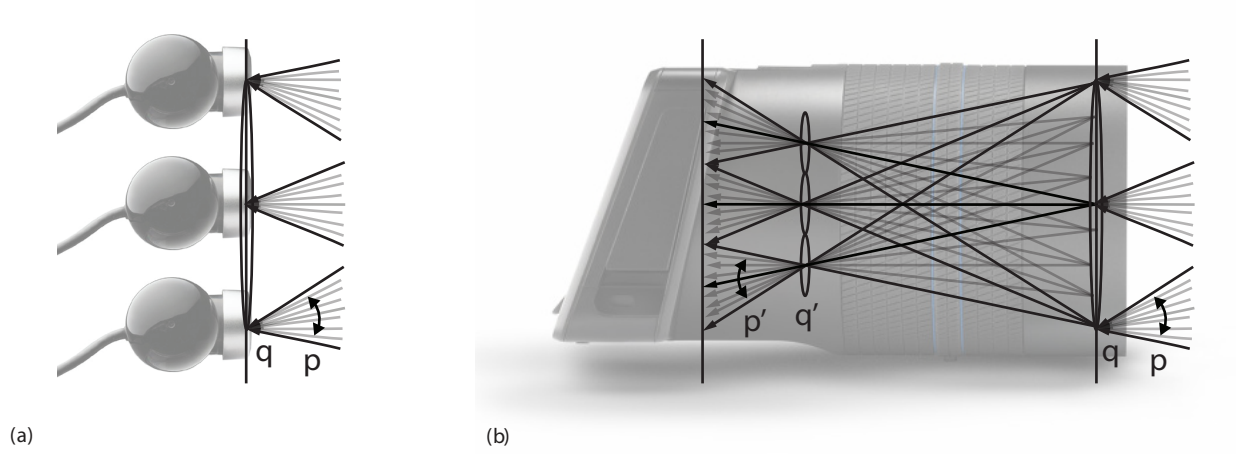

Figure 3: Typical sparse light field capture schemes: (a) camera array, (b) plenoptic camera.

scene, the aperture, the sensor) from the image formation model separately and parameterize all rays that pass through the given surface by their intersection point, q, with the surface and their exitance angle, p, with respect to the surface. This is known as a point-angle parameterization. Since the surface is usually embedded in 3D space, q may be treated as a three component vector (x, y, z). Also, for computational conveniences, the directional component p is often assumed to store the tangents of the ray angles θ and φ. This is expressed in Equation 3.

$$LF \quad = \quad L(q, p) = L(x, y, z, tan(\theta), tan(\varphi)) \tag{3}$$

Figure 2 (taken from [33]) illustrates the above-mentioned standard light field parameterizations. In this work, I will mostly be using the point-angle parameterization. The special case when a point-angle light field is defined over a scene surface is commonly referred to as a surface light field [3]. Most of my work will focus on surface light fields (SLFs). SLFs are simply a view-dependent generalization of texturing [11, 60]. They are a convenient way of modeling the radiance coming from specific scene points. In addition, SLFs are a 4D subset that is suitable for representing the non-redundant information in the plenoptic function of a complex scene. A more detailed discussion of SLFs and their relevance to this proposed work is presented in Section 4.

3 Light Field Photography

3.1 Sparse Light Field Acquisition

There are two common ways of obtaining discrete light fields from real-world scenes: one way is to use an array of cameras [59, 57] and the other is to use a standard camera augmented with a microlens array [2, 41]. Both of these designs are illustrated in Figure 3. It is important to note that, in both cases, the sampled light field can be the same, so either choice is equally suitable for experimenting with light fields in general.

With the camera array approach, the array is usually designed to discretely sample the light field that passes through some desired virtual aperture. For static scenes, a single camera can be used as well, resulting in a multi-view set of images. Refocusing the captured light field requires an additional step of refracting the captured rays (say, according to the thin lens law [48]), before projecting them onto a final focal surface. Not all of the refracted rays will hit the focal surface, so some sensor space may be wasted. Currently, camera arrays are finding use in mobile technologies due to their compactness [57].

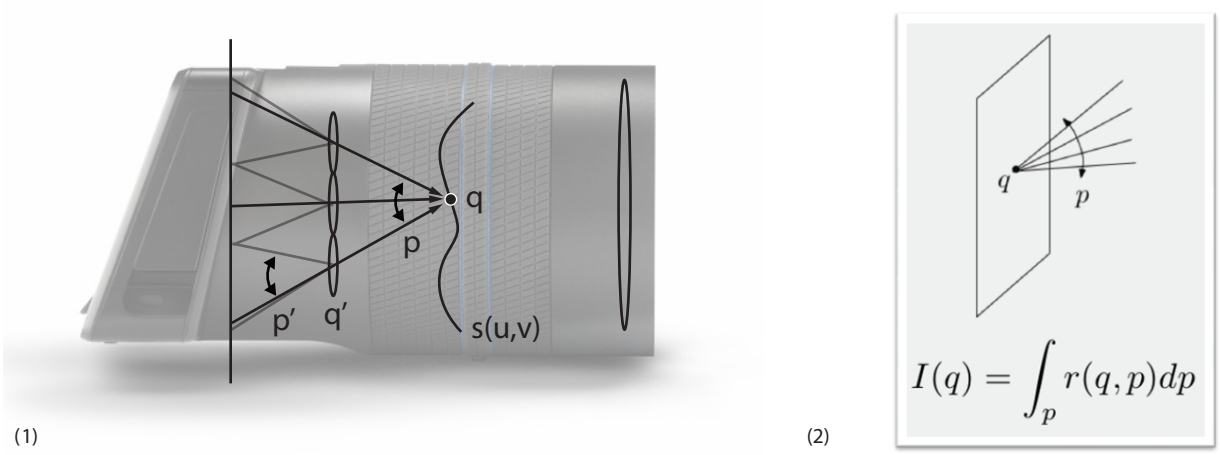

Figure 4: Light field rendering can be thought of as a two-step process: 1) transform a captured light field to a desired focal surface; 2) integrate the irradiance at each point of the focal surface to form an image.

With microlens-based plenoptic cameras, the captured light field is pre-warped by a main lens. This makes it easier to directly synthesize images that correspond to the lens and aperture settings that were chosen before capture. In most cases, the microlenses share a common sensor, which allows for spatio-angular resolution tradeoffs to be made through various re-calibrations [14, 37]. It is also common to calibrate the camera in a way that keeps the f-number of the microlenses matched to that of the main lens. Since the microimage corresponding to each microlens maps the full main lens aperture, f-number matching ensures that neighboring microimages will neither overlap, nor have gaps between each other. One could also think of transforming the microlens array though the main lens and out into the scene [18]. Undoing the effects of the main lens through such a transformation is an important calibration step in relating the captured light field to the actual scene and possibly to other available imagery of the scene. This can be useful for scene reconstructions and for making viewing and focal choices relative to the scene.

For this work, I will most likely be using both multi-view data, in a way that samples an unstructured light field [10], and plenoptic camera imagery, which captures regularly sampled light fields. The specific instrumentation and software are discussed further in Section 4.3.

3.2 Light Field Rendering

Some of the pivotal works on light field rendering include [34, 20, 41, 43, 37]. A lot of the earlier works [34, 20] concentrated on the ability to move a viewer within the bounds of the captured light field. The algorithms for such effects mostly comprise of combining information from neighboring, pre-captured views of the scene that approximate the final desired vantage point. It is exactly such techniques that form the core of image-based rendering [53], which is why light field rendering often falls in this category. Still, later works on light field rendering [41, 47, 37] shifted more towards thinking about individual rays from the captured light field, as they aimed at simulating more complex, wide-aperture photographic effects. In this case, the light field can be treated as a discrete set of radiance-carrying rays from the scene. Using ray optics, each ray can be transformed from the original light field surface to some desired focal surface, effectively re-parameterizing the light field [23]. Subsequently, an image can be formed by integrating the irradiance contributions from all rays incident on a given image point, q:

$$I(q) \quad = \quad \int_p r(q,p)dp \tag{4}$$

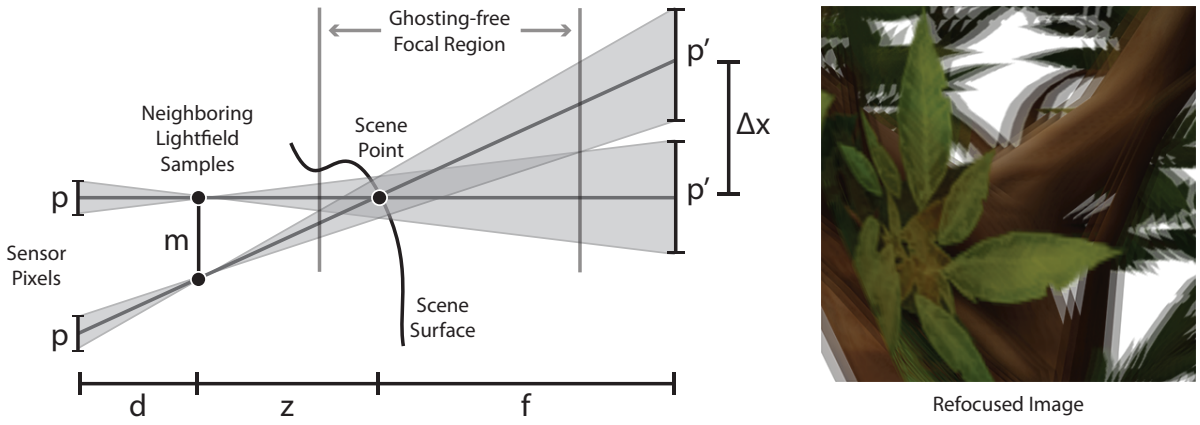

Figure 5: Smooth, ghosting-free defocus blur is achievable only within a limited defocus region when working with sparse light field data. (Left) illustrates the problem diagrammatically; (Right) shows a rendition focused on the foreground and exhibiting severe ghosting artifacts in the background.

This process is illustrated in Figure 4 and expressed in Equation 4 (taken from [37]). Transforming the original rays is an input-centric, or light scattering, approach to light field rendering. Alternatively, the rendering equation can be evaluated by tracing a discrete set of rays into the original light field, for each output point. This output-centric approach has been preferred in practice, as it fits standard graphics pipelines and texture filters, like the ones implemented in OpenGL[52]/GLSL[50] compatible GPUs. Even the original paper on light field rendering [34] took a similar output-centric approach, but demonstrated just the case of pinhole view synthesis. With appropriate filtering applied to the looked-up samples, the input-centric and output-centric approaches can produce equivalent results.

The next sub-section, 3.3, discusses some open problems in light field rendering. One of the contributions of this thesis will be to show that these problems can be avoided when rendering from surface light field models, or any equivalent light field set, which preserves information about the positions of the viewed scene points. This proposition is discussed along with each presented problem.

3.3 Open Problems

3.3.1 Ghosting Artifacts

Ghosting artifacts appear in a light field rendition whenever the projections of neighboring view samples from a given scene feature are far enough apart from each other that gaps are left onto the output focal surface. Figure 5 illustrates this problem. The diagram on the left shows a pair of light field samples, which act as pinhole views of a scene. Because each sample is an all-in-focus view, a given scene point is considered to be resolved by a single pixel on the image plane of each view. It is fine to assume that the pixels are not aliased, or that appropriate filtering is applied when reading them. During refocusing, the captured pixels for the scene point are effectively projected back into the scene and onto a chosen focal surface. There will always be a region of space, around the actual position of the viewed scene feature, that allows for ghosting-free refocusing of that feature. This is also illustrated in Figure 5, where the refocusing limits, for a chosen scene point, are marked in terms of distance along the viewing axis of one of the available light field samples. We can consider changing focus by shifting a focal plane along this viewing axis. This allows us to state the ghosting artifact problem more formally. To begin, Equation 5 expresses the distance between neighboring projections of a refocused feature, and Equation 6 expresses the projected size of a pixel at a given depth. The equations are given in terms of the following parameters: the focal length of the light field samples, d; the distance between neighboring light field samples, m; the distance between the scene

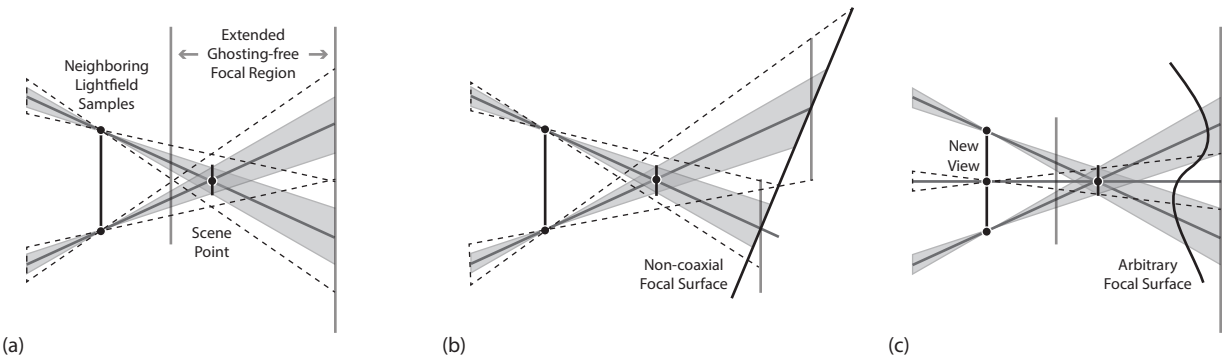

Figure 6: Ghosting artifacts can be removed by pre-filtering the captured light field samples according to their depth value (a). Depth values are view-dependent, which means that, for a general focal surface, the same filter cannot be used for all samples from a given depth layer (b). In addition, pre-filtering blurs the light field and fails to account for partial occlusions. The correct filtering should introduce sharp, in-between views of the scene (c), which, in turn, requires modeling the scene.

point and the light field sample pupils, z; the size of a pixel on the light field sample image plane, p; and the desired focal offset for the imaged scene point, f.

$$\Delta x = \begin{cases} -\frac{fp}{d} & \text{if } z = 0 \\ -\frac{fm}{z} & \text{if } z > 0 \end{cases} \tag{5}$$

$$p' = -(z + f)\frac{p}{d} \tag{6}$$

Based on these expressions, we can state the requirement for artifact-free refocusing using the inequality in 7. Ideally, we want that inequality to hold regardless of our choice of f, which will put no limits on the amount of defocus that we could introduce to any scene point.

$$\|\Delta x\| \leq \|p'\| \tag{7}$$

An important observation is that $\Delta x(f, p, d, z = 0) = p'(f, p, d, z = 0)$, which means that Equation 7 holds for any choice of focusing, whenever the light field is sampled at the scene point of interest, $z = 0$. This tells us that a surface light field sample can be arbitrarily defocused without ever exhibiting ghosting artifacts. This is a novel perspective on artifact-free plenoptic rendering, which I propose in this work.

The artifact problem has long been addressed in prior literature and in some cases has been solved. Earlier works used uniform 4D linear filtering to reconstruct the missing rays from the light field gaps [34, 20]. This worked well for the Plenoptic 1.0 light field acquisition schemes that were standard at the time [41]. Each light field sample was assumed to capture directional information from just one scene point, the way things would actually be in the case of a surface light field. Nonetheless, for the actually captured light fields, this assumption was true for just one focal plane in the scene. By treating objects from other focal planes as though they were captured in the same way, not all available data gets used during rendering and the resulting images have the minimal possible output resolution. This problem was addressed with the introduction of focused plenoptic cameras and rendering [37, 15], which allowed for correct treatment of the light field samples obtained away from a given scene surface. Unfortunately, the ghosting artifact problem becomes apparent with this rendering approach. Uniform filtering can still help remove the artifacts, but at the cost of blurring the focused parts of the scene in proportion to the size of the ghosting gaps that need to be filled in. This is equivalent to reducing the output resolution, which is a problem in itself.

One intuitive approach to artifact removal without over-blurring is to use light field filters that depend on depth [7, 54, 13]. In fact, surface light fields would be an extreme case of depth-dependent filtering, as the entire scene is reconstructed in order to obtain an SLF. Still, any dependence on depth requires obtaining additional information on how the scene is actually laid out. In relation to this, depth extraction from light fields is an ongoing effort in computer vision [55, 58, 4, 19, 56]. Usually, a disparity map can be extracted, which effectively stores z-distances to points in the scene. If all available views in the light field see the same disparity value for a given scene point, then it is possible to meaningfully separate the light field into depth layers. Additionally, if all samples from a depth layer are to be defocused by the same amount, a depth-based filter can be applied to these samples to extend the ghosting-free defocus region. This pre-filtering is illustrated in Figure 6(a) and, in practice, can be used when rendering to focal planes that are parallel to the plane of the light field samples. This corresponds to standard camera refocusing, but is a special case in the context of computational photography, where any focal surface should be allowed. For a general focal surface, the defocus amounts for each light field sample may have to be different, see Figure 6(b). Adding a view-dependence to the used defocus filters can correctly handle this case, but it complicates the renderer. On top of this, this is not the only potential problem with pre-filtering a light field sub-image. When blurring a light field sub-image, in effect, new rays are introduced into the light field, without occlusions being taken into account; so doing this type of filtering is not a good approach, in general.

A correct reconstruction filter for the missing directional samples should keep the sub-image pixels sharp, but should synthesize valid in-between views of the scene. Figure 6(c) shows this idea. Note, how a new view, reconstructed between the original light field samples, extends the defocusable region; allows for refocusing onto arbitrary surfaces; and keeps the original sharpness of the input pixels. Reconstructing in-between views requires modeling the objects in the scene and correctly accounting for parallax effects. Such smooth, novel-view synthesis has been demonstrated in prior work [32, 45] and can solve the artifact problem. The previously proposed approaches segment the scene into content aware layers and apply parallax effects to them to obtain occlusion correct novel views. The used layers could be redefined for every view in the light field, to capture the scene's outlines more accurately. Such view-specific layers would represent more of a viewing proxy, than an actual model of the scene, as a scene point cannot directly be identified in neighboring views. Instead, corresponding layers from different views would simply get morphed into each other during view interpolation. While this is perfectly acceptable for all-in-focus renditions, wide-aperture imaging requires combining information from neighboring views, which, in turn, necessitates a shared view-independent scene model.

Whenever a model for the scene is available, rendering from reconstructed surface light fields would be a natural and practical solution, which needs to be explored further. In the case of static scenes, sophisticated scene models can be obtained through rich, multi-view imagery and 3D scanning. In addition, pre-made and fitted 3D models can be used in all practical cases. For dynamic scenes, depth layers are still the preferable approach to modeling the scene, as they are both obtainable through vision techniques applicable directly to the captured light field, and they work for complex scenes. For instance, commercial plenoptic cameras [42, 46] rely on a single depth map. This is in part due to the limited parallax seen by the main lens aperture. The small aperture prohibits complex scene reconstructions, but at the same time makes them less necessary for visibly correct refocusing. Lytro [42], for instance, generates a single hexagonal arrangement of sub-aperture views, from which a reliable disparity map is extracted for the central view. The layers from the disparity map are a rough model for the scene, but suffice for rendering. A surface light field could potentially be mapped to the layers, and, with it, more advanced rendering and filtering techniques can be applied to the captured data. As is discussed further in this text, there are additional benefits to be gained from working with a surface light field, aside from avoiding ghosting artifacts.

3.3.2 Limited Depth of Field Effects

Light field photography [41] can be used to simulate all types of standard camera focusing. Conversely, all of the refocusing effects that have been demonstrated with light fields have been limited to model the imaging geometry for some real-world camera. Nevertheless, there are focusing effects that cannot be

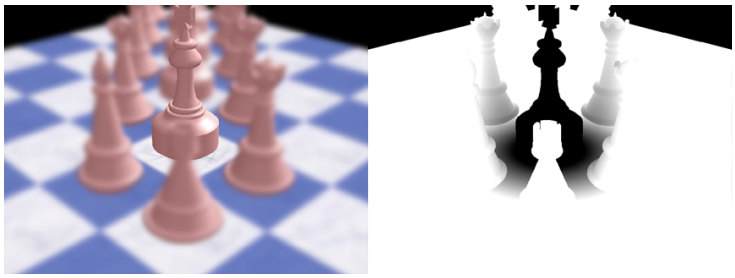

Figure 7: The left image, taken from [29], shows the use of generalized focusing as a means for highlighting a single object in a dens scene. The defocus amounts for different points in the scene are shown in the right image using grayscale.

produced even with a physical camera, and these have not previously been shown in the context of light field rendering either. One possible direction for generalizing the concept of depth of field (DoF) has been to aim at allowing individual points in the scene to be focused independently from the rest of the scene. Figure 7 shows a synthetic rendition that makes use of such focusing to highlight a single object in a crowded scene. The cognitive implications of being able to focus on individual objects, as opposed to entire focal surfaces, were first considered in [28], where the notion of *semantic depth of field* was established. In that initial work, they simply rendered each object from the scene into a separate layer and re-composited the scene, back-to-front, after blurring each layer to a desired amount. The more general problem statement, of independently defocusing individual points from the scene, was first introduced and addressed in [29, 30]. The proposed renderer in [29] is also based on view-dependent layer stacking, but each layer is a depth peel [12] from the scene. Also, they use an iterative blurring approach, which allows for separate points in each layer to be blurred by a different amount. Still, there are plenty of issues with using blur as a substitute for real defocus, which make the presented renderers fundamentally flawed. Since defocusing is not achieved through a realistic physical model, partial occlusions are a problem. Also, it is not obvious how aperture effects (e.g., shaped bokeh) can be modeled. On top of this, in a realistic rendition, some points in the scene may have to contribute light to the output image only when sufficiently defocused. These are points that would normally fall outside of the field of view of the starting all-in-focus rendition and, therefore, cannot be taken into account with this approach. But, the criticism does not end with the plausibility of the output images. Rendering speed can be a problem, too. If the camera's perspective is to change even slightly, the layers need to be recomputed. And even a simple change in focusing can be a very slow operation when it requires progressively re-blurring a set of sparse layers. To top it all, the proposed interface for specifying focusing works for choosing larger sections of the scene and not individual points. While this may be convenient for quickly implementing certain types of focusing that fit this model better, it can become a problem very quickly if we consider any rearrangements of the scene, or if we actually want to specify focusing on a per-point level. In this sense, the proposed approach even fails to meet the core goal of the depth of field generalization - to allow a user to change the focusing of a single point. One alternative approach, which allows per-vertex specifications of focusing, was presented in [40]. Their proposed renderer is applicable to stochastic polygon rasterization and requires high levels of tessellation to produce quality results. It also cannot handle general aperture effects in a physically accurate way and has a limited defocus amount, due to polygon clipping.

In contrast, I propose a generalization to the wide-aperture image formation process that is both physically accurate and can meet the requirement of refocusing each point in the scene separately. My approach is based on computational photography techniques applied to surface light fields. More details are discussed in Section 4, but, before presenting them, it is important to consider why even standard light field rendering has limited focusing capabilities. To begin, Figure 8(a) shows a diagram of standard camera imaging, where a ray, incident on the aperture, is refracted by the main lens and transmitted to a coaxial planar sensor some distance behind the main lens. Using linear optics [21], thin-lens refraction and ray propagation

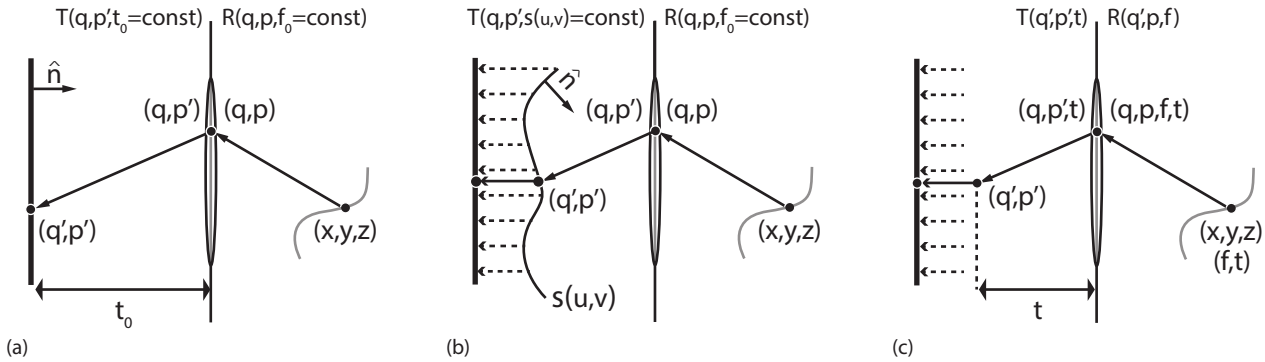

Figure 8: The three figures illustrate the refraction R and transmission T that a ray undergoes when passing through different computational camera models. Sub-figure (a) shows standard camera focusing, based on a constant focal length f_0 and a user-defined sensor position t_0. Sub-figure (b) show an arbitrary focal surface, like the one implied by a depth map in a plenoptic camera rendition. This gives more control over focusing by varying the transmission distance of each refracted ray according to its intersection point with the focal surface. The focal surface itself is still a constant within a single rendition. Sub-figure (c) shows generalized, per-ray focusing, where both f and t are variable parameters.

can be modeled as linear transformations. For standard image formation, all rays are refracted according to a fixed focal length f_0 and transmitted to a user-defined sensor distance t_0, where the image is formed. Increased control over focusing can be achieved by using a non-planar focal surface. This is illustrated in Figure 8(b) and is achievable using light field refocusing techniques [41, 15]. One attempt to take this idea a bit further was made in [25], who take multiple slices through the focal volume behind the main lens and composite the resulting images to form a final image. This is not a very practical approach, as it can easily result in multiple registrations for the same ray, which is actually a ghosting artifact. Regardless of this, no matter what the chosen focal surface is, it remains as a rendering constant for all incoming rays. Contrary to this, for my generalized image formation model, I consider using variable imaging parameters. Figure 8(c) shows a refocusing function that uses a variable focal length f and a variable transmission distance t, defined for each scene ray, separately. The resulting composition of transformations, which takes radiance from a surface light field ray to an image-space position q' is expressed in Equation 8.

$$F = T(t) \circ R(f) \circ T(z) \tag{8}$$

The resulting per-ray focusing may be an impractical over-generalization. Because of this, for the renderer that I propose in Section 4 , the focal length will be kept constant, and defocusing will be specified per-point, using a defocus attribute, t, parameterized on each surface. Depending on the desired visual effects, t could be interpreted in relation to other parameters of the scene or the modeled camera. This approach, of specifying t at each surface point, can also fit nicely with standard 3D texturing workflows, which eventually will allow for a straightforward implementation of a focus brush tool.

3.3.3 Plenoptic Vignetting

In Section 3.2, it was mentioned that one possible way to visualize the light field refocusing process is to consider combining neighboring, sub-aperture views of the scene. This idea is illustrated in Figure 9, using the leftmost and rightmost sub-aperture views captured by a plenoptic camera. The left sub-figure shows a light field captured with a Plenoptic 1.0 calibration, where the microlens array virtually projects to the depth of the viewed objects in the scene. To describe the captured light field, we can use a two-plane parameterization, with on plane corresponding to the aperture and the other to the microlens conjugate plane. A uv point on the aperture plane specifies a sub-aperture pupil, while the allowed range of st

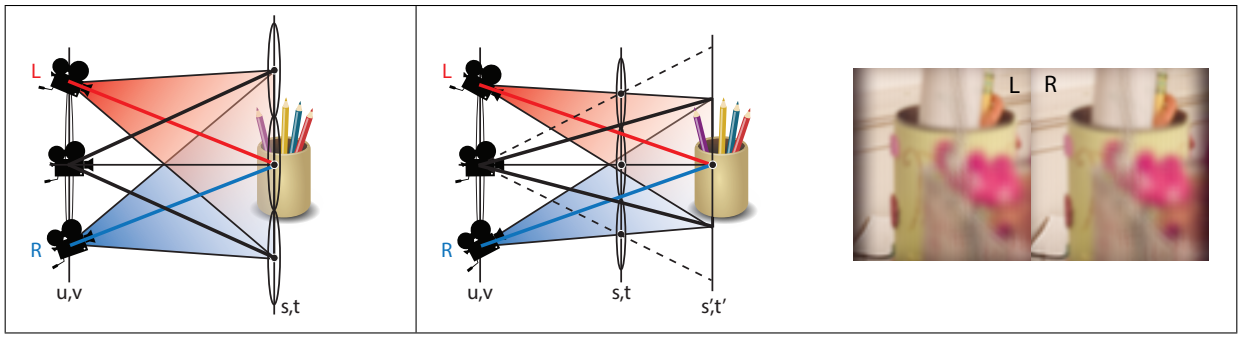

Figure 9: The two diagrams show the viewing geometry for two calibrations of a microlens-based plenoptic camera. The captured light field is shown parameterized between the aperture and microlens array planes in scene-space. The originally acquired views refocus on a scene object only for the Plenoptic 1.0 calibration (left diagram). Focusing on a scene object with a Plenoptic 2.0 light field requires a reparameterization step (right diagram). Light field reparameterization results in plenoptic vignetting, which either needs to be cropped away, resulting in a decreased filed of view, or needs to be filled in by view extrapolation. A stereo pair of vignetted images, corresponding to the illustrated reparameterization, is also shown to the right.

points specifies the stereo window, through which the scene is viewed. Still, this Plenoptic 1.0 calibration is exactly the case that corresponds to the lowest achievable spatial resolution for the objects of interest. Some commercial plenoptic cameras [42] do use a calibration that approaches this, but, nonetheless, the requirement is very restrictive and can at bets be met by only one focal plane in the scene. It is focused plenoptic rendering [37] that applies to objects in the planes away from the microlenses. As illustrated in Figure 9 (right), this calibration places the stereo plane, and thus the original fixation point, away from the scene. With a Keplerian calibration, the captured views will meet in front of the viewed object, while, with a Galilean calibration, the captured views will fall behind the viewed object. In general, the fixation plane will always correspond to the plane in focus, so refocusing will subsequently move the fixation plane by reparameterizing the original light filed [23]. A good side-effect of this is that it is exactly what is needed in the case of stereoscopic plenoptic rendering [2], where the eyes should ideally meet at the objects of interest [38, 22, 27]. The only problem with the needed reparameterization is that not all rays from the reparameterized light field will be available for look-up in the original light field. More specifically, the region of overlap between the left and right views will shrink, requiring us to render at a reduced field of view. This is also illustrated in Figure 9 (right) and results in plenoptic vignetting. The effect is also shown for a stereo pair of rendered images. It is not uncommon to simply reweight the vignetted pixels in the output image [23], as is done for standard vignetting. This would not really fix the issue, since plenoptic vignetting is caused by missing views and not by lowered light sensitivity. The absence of these side views causes the edges of the rendered image to have a warped perspective relative to the rest of the image, which cannot be corrected with reweighting.

Unlike missing in-between rays, which can be reconstructed though interpolation, the rays that are missing after reparameterization need to be obtained through extrapolation. To a certain extent, this can still be done by applying parallax shifts to a layered scene representation, just like it has been done for synthesizing in-between views [32, 45]. An alternative approach, which could potentially capture the reflectance properties of the viewed scene points, is to extrapolate the directional components of the surface light field samples for the scene. There are several assumptions that could be made that would allow meaningful extrapolations of the directional samples from a surface light field. In the absence of additional knowledge about the illumination of the scene and the true reflectance of the viewed surfaces, SLF extrapolation can be an acceptable solution to the plenoptic vignetting problem, while even preserving some specular effects.

3.3.4 Camera-dependent Renderers

Light field rendering can always be implemented by densely casting rays into the captured light field and fetching appropriately filtered samples, as was discussed in the original paper on the topic [34]. At the same time, in practice, plenoptic renderers tend to be implemented differently and in ways that are tightly coupled with the design of the plenoptic camera being used. This is because each camera discretizes and flattens the captured light field differently. For instance, Plenoptic 1.0 mosaics store corresponding directional samples from the light field within each microimage [41, 43], while Plenoptic 2.0 mosaics store these components across microimages [37, 15]. This camera-dependence gets passed onto the renderers, as standard refocusing algorithms are usually designed to render directly from the native image format of the camera. This is preferable both because rendering can be achieved without the use of any proxies and because no repeated resampling of the captured light field is necessary, which could otherwise diminish the final resolution. As different camera designs and light field acquisition schemes have gradually been introduced throughout the years, the need for a more unified approach to rendering from the resulting data sets is re-emerging, possibly unnoticed by the community. While in some cases it may still be possible to transform the captured data to some standardized single light field form, in a lot of cases this is not possible. More specifically, some of the presented acquisition schemes no longer capture a single, structured light field, but can capture multiple [16, 47] and possibly unstructured light fields [10]. Combining the captured data to form a final rendition could depend on depth information from the scene, or even on having a fully reconstructed scene. In the general case, it is best to be able to undo all of the effects of the capture instrumentation. In a sense, this equates to mapping the captured light field data back to the scene surfaces and reconstructing the scene's SLFs. In fact, the techniques currently in use for surface light field acquisition [60, 24] are based on calibrated multi-view photography, for which the capture process is inverted to create an SLF. When avoiding resampling is preferable, this inversion can be done on the fly via view-dependent texture mapping techniques [49, 11] and the SLF could simply be treated as an abstraction or a rendering proxy.

4 Surface Light Field Photography

4.1 The Plenoptic Function and Surface Light Fields

Referring to Figure 10, consider the plenoptic function , P_{scene}, of a given scene. As was mentioned in Section 2.1, the light field passing through the aperture of a camera, $LF_{aperture}$, is a slice from P_{scene}, from which wide-aperture images can be formed through subsequent computational re-projections. In the view-space of the aperture, this slicing is given by Equation 9, where A is the set of points on the aperture.

$$LF_{aperture} = P_{scene}(x, y, 0, \theta, \varphi) = L(q, p), \qquad (x, y) \in A \tag{9}$$

Since all rays that pass through the aperture travel unobstructed inside the camera, the plenoptic function of the free space inside the camera, P_{camera}, can be reconstructed from $LF_{aperture}$ without any losses. Thus, $LF_{aperture}$ is a lower-dimensional representation of P_{camera}. More importantly, the light field at any sensor surface, which would also be a slice from P_{camera}, can be reconstructed by projecting $LF_{aperture}$ onto that surface. This is the basis for light field rendering, discussed in Section 3.2.

We can attempt to draw some analogous relationships in scene-space, knowing that some things will have to be treated differently, as the scene is not just free space. I claim that the plenoptic function of the scene itself, P_{scene}, can be derived from a lower-dimensional light field set, but which is discontinuous and maps to all points in the scene that reflect or emit light in the scene. The transmission of radiance throughout the rest of the scene can subsequently be computed by tracing rays from these points to the desired locations. Again, referring to Figure 10, the radiance sources in the plenoptic function of the scene are given by the discontinuous surface light field that covers the scene. If the scene is comprised of a set of parametric surfaces $s_i(u, v)$, then the scene's surface light field can be written as:

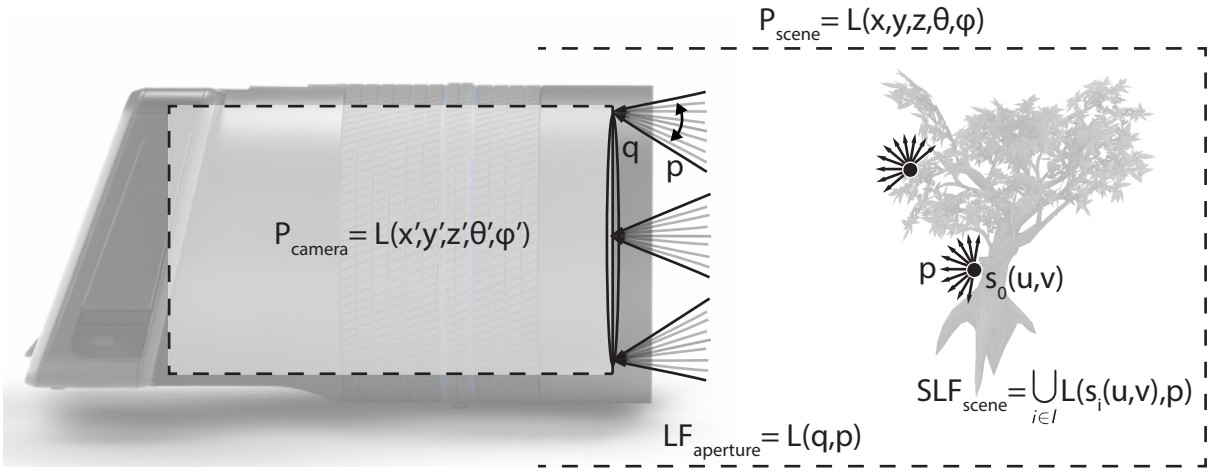

Figure 10: A camera's aperture slices a light field from the plenoptic function away from the viewed scene. This slice captures all rays entering the camera, which makes it a suitable lower-dimensional model for the plenoptic function inside the camera. This same light field is a projection of the light fields defined on the surfaces of the scene. In general, this is a lossy, irreversible projection. To avoid losing information about the radiance sources in the scene, the union of the surface light fields of the scene needs to be captured, or reconstructed. This union can be thought of as a single discontinuous surface light field, SLF_{scene}. Image formation can subsequently follow from this surface light field without any of the data, needed for computational refocusing, being lost.

$$SLF_{scene} = \bigcup_{i \in I} L(s_i(u, v), p) \tag{10}$$

With this in mind, the aperture light field $LF_{aperture}$ can be viewed as a projection of SLF_{scene}, in addition to it being a slice from P_{scene}. Given that the space between the scene points and the aperture is not necessarily empty, this projection can be lossy and irreversible. On one hand, this is due to possible ray occlusions and retransmissions, and on the other, information about the spatial distribution of the radiance sources is lost. Rays that are occluded on their way to the aperture should not contribute to the final image, so it is fine that they are not present in $LF_{aperture}$. The bigger problem is that, without knowing where a given ray originated from, it is impossible to tell where it will end up on the final image in relation to other rays that come from the same scene point. In other words, nothing can be said about how a given scene point will end up focusing after $LF_{aperture}$ is transformed to a given image plane.

To see why this is so, we have to consider what parameters are needed to fully specify focusing. Using the thin-lens law from linear optics [21], focusing can be expressed as a function of both position (x, y, z) and some fixed focal length f. A conjugate point can be computed as $(x', y', z') = F(x, y, z, f = const)$. This is illustrated in Figure 11(a) for a set of unoccluded rays. As was briefly discussed in Section 3.3.2, in relation to Figure 8 and Equation 8, my motivation for considering surface light fields as the starting point for image formation is precisely so that positional information is not lost and focusing can be controlled more directly. In the next sub-section I propose two SLF-based renderers, which allow for the specification of a variable focusing parameter for each point in the scene and can model generalized depth of field effects in a physically-accurate way. Although, in some cases it may be preferable to provide implementations that perform relative defocusing (say, w.r.t. a focal plane), when absolute specification of focusing is needed, positional information for the radiance sources in the scene has to be preserved and associated with each ray being refocused.

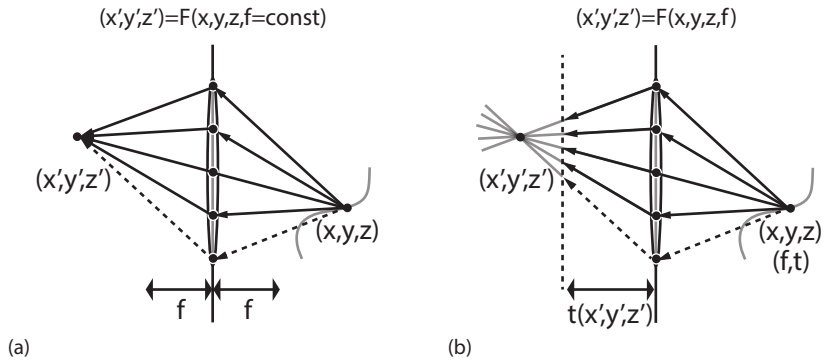

Figure 11: Thin-lens point refocusing. (Left) shows the standard focusing model, where a point (x, y, z) is mapped to its focal conjugate (x', y', z') according to the thin-lens law for some chosen focal length f. (Right) shows the proposed offset focusing model, applied relative to the ideal focal conjugate, which allows a per-point focal offset t to be applied when tracing rays to form an image. One possible choice of t is for it to specify a percentage of traveled distance from the aperture to the conjugate (x', y', z').

4.2 Image Formation with Offset Focusing

In this sub-section, I propose a physically-based image formation model, which allows for generalized depth of field effects. I also consider two possible implementations of this model; one suitable for ground truth renditions, and the other suitable for real-time rendering. The image formation model that I am considering is generally applicable to surface light fields, or any equivalent data. In Equation 8, I described an over-generalized image formation model, where every ray incident on the main lens is refracted and propagated independently. In practice, this model can be constrained, while still preserving the ability to apply defocus to isolated parts of the scene. Since refocusing a scene feature requires transmitting all rays, which correspond to it, in some lockstep fashion, the transmission parameter, t, can be constrained to vary only at the granularity of individual scene points. Also, in order for t to specify a focusing offset, it needs to be defined relative to some, non-offsetted, camera-space focal distance. As illustrated in Figure 11(b), choosing t as an interpolation parameter from the aperture to the ideal focal conjugate of the refocused point is one possible option. The resulting circle of confusion will have a fixed size, relative to the aperture size A, regardless of where the camera is placed. This is expressed in Equation 11.

$$CoC = A(1 - t), \qquad t \in [0; \infty) \tag{11}$$

When using defocus strictly as an attention cue, this approach may work best. At the same time, there are important depth cues that would entirely be lost. To contrast this, when focusing is done on a shared focal plane, different depths map to different defocus amounts, which is valuable in itself. To capture such effects, when necessary, we could, for instance, interpret t as a focal offset relative to some shared focal surface. More importantly, the choice of interpretation for t can also be done on a per-point basis, allowing for defocus to possibly play different roles in different parts of the scene. In addition, conceptually assigning a point of origin, for the purposes of such semantic focusing, may not always be meaningful when it comes to rays that were refracted or retransmitted in the scene. In such cases, it is best to be able to use standard refocusing in combination with the generalized focusing techniques. This leaves us with at least three different image formation models, which we would ideally like to seamlessly mix in the final rendition. For this mixing to be possible, the offset rays needs to undergo an final transformation, which projects them to the same shared focal surface and using the same central view perspective, as would have been used for standard focusing. Finally, knowing that generalized focusing is achievable just by varying the transmission parameter t, we can use a shared focal length $f_0 = const$ for all incoming rays, again, as is done in standard focusing.

4.2.1 Rendering with Offset Focusing

A ground truth implementation of the image formation model with offset focusing can be provided via ray tracing. This can be done as an extension to any engine that can simulate global illumination, e.g., Blender Cycles [5]. In such engines, wide-aperture cameras are usually modeled by randomly choosing positions on the aperture surface, through which to shoot rays for every sample on the output image plane. Note, that this effectively samples a two-plane light field inside the camera. The generated rays are then refracted to point towards some desired focal plane and shot into the scene as primary rays. Usually, both the pixel samples and the aperture samples are taken from some low-discrepancy distribution, which introduces acceptably looking noise, in place of aliasing, and allows for optimal results for the number of used samples [48]. This type of focusing is illustrated in Figure 12(a).

Unfortunately, this output-centric generation of primary rays is not as trivially applicable to offset focusing. The reason is that the focusing offset effectively breaks the path of each ray inside the camera and redirects it before the ray finally reaches the sensor. This means that the only way to know the correct image position for a given ray is to trace its path from its originating surface to the image plane / sensor. This step is shown in Figure 12(b), where the starting point for a given ray is randomly sampled within a texel on a scene surface. Alternatively, the primary rays can be generated as usual, but, upon hitting a surface that requires offset focusing, they can be re-traced back to the sensor. In both cases, the distribution of the samples on the sensor will no longer be predictable, but this is a minor issue.

Having propagated each ray to its transmission distance, t, it remains to decide exactly how to project it to the final image surface. On a global scale, the choice of projection can influence the perspective, with which the scene is viewed. In wide-aperture imaging, the perspective of the final image is usually assumed to be that of the central, sub-aperture view. Likewise, because of aperture symmetry, during refocusing, the center of the perceived circle of confusion is also assumed to fall on the line between the aperture center and the focal conjugate for the point being refocused. Because of this, it seems most natural to use this direction for the final projection of a an offset ray. To do this, information about that ray's assumed point of origin, or equivalently its focal conjugate (x', y', z'), needs to be kept, too. This choice of projection is illustrated in Figure 12(b), for a single ray, and again in Figure 12(c), for an entire circle of confusion. Compared to a standard image, the end result effectively allows the projected size of the CoC to be varied according to the parameter t. This happens without the aperture having to shrink or grow and can be used to recreate intricate bokeh effects. Radiance conservation is not violated by this process. At the same time, radiance transmission is controlled algorithmically, which makes this a form of physically-accurate, computational image formation.

4.2.2 Real-time Rendering with Offset Focusing

Precomputation can significantly speed up the renderer described in 4.2.1. On one hand, when nothing in the scene changes, the primary rays from the scene can be stored in a surface light field. This will allow for the computational camera to move around freely, while at the same time no global illumination will have to be done on the fly. Essentially, the camera will be looking at the scene with a view-dependent texture mapped to each surface. On the other hand, if we also fix the position and orientation of the camera, we can afford to propagate the SLF samples all the way to the aperture. As long as additional information, about how each ray is to be transmitted, is preserved, the achievable focusing and aperture effects will not get limited. Finally, rays that originate from the same point in the scene and are transmitted directly to the main lens can be refocused as a group. This is shown in Figure 12(c), where an aperture image (AI), A_{image}, is captured for a given viewed scene point. Parts of this image may end up being hollow/transparent, whenever the viewed point is partially occluded and some rays do not make it to the aperture. Nonetheless, the AI for that point can be formed; modulated with the shape of the aperture; and, finally, projected to the output image plane as a circle of confusion of a given size and orientation. An AI can even be formed for a single incident ray, so nothing is lost by conceptually grouping rays in this way. Also, note that by describing each ray by its origin on a scene surface and its intersection with the aperture, we have created a new type of light field parameterization. For now, I will be calling it a *surface-aperture light field* (SALF).

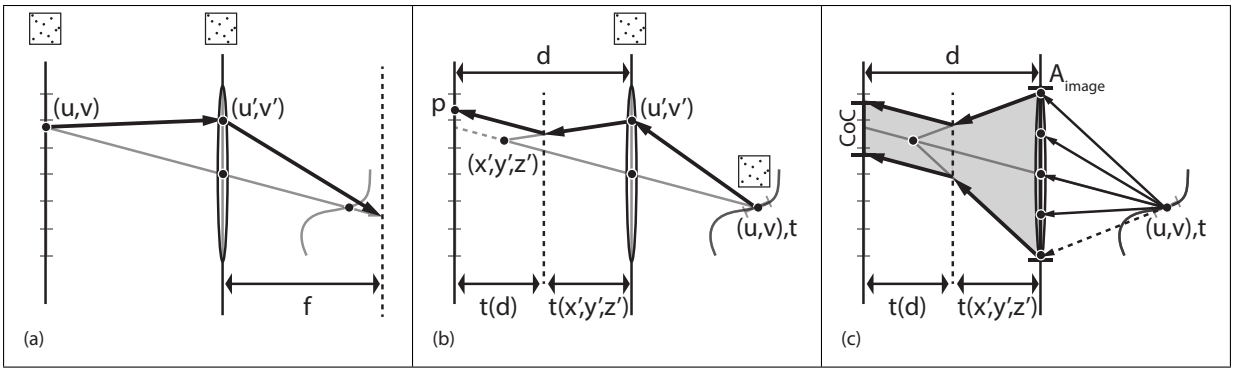

Figure 12: Sub-figure (a) shows standard distributed ray tracing DoF. Both the pixel area and the aperture area are super-sampled and a ray gets cast for a pair of samples on the two planes. Subsequently, the ray leaves the aperture towards the ideal focal point for the chosen focal distance f. Sub-figure (b) shows a comparable setup for refocusing a random ray from a super-sampled SLF texel. The ray is refracted at the aperture and traced to some offset focal distance. The offset t can be interpreted relative to the ideal focal point (x', y', z') , or relative to a chosen sensor position d. To reach the final image surface, the ray sample can be orthographically projected. Sub-figure (c) considers projecting the entire circle of confusion (CoC) for a given scene point, using the same approach as in (b). The choice for the final projection direction is made relative to the central sub-aperture view, so that the CoC will not shift in apparent position during refocusing. When multiple samples from the same scene position hit the aperture, an aperture illumination image A_{image} can be precomputed and rendered using real-time techniques like polygon projection and rasterization.

In practice, all of the intermediate light field representations that I plan to use will have to be discretized. While there is some prior work on reconstructing and representing SLFs [8, 60], my SALF parameterization is sufficiently different. In an SALF, two parameters identify a 3D point in the scene and the remaining two map to points on the aperture. This most closely resembles Plenoptic 1.0 camera imagery [43], but in an idealized case, where each microimage is precisely positioned on a given surface point. Because of this, one option is for a discrete SALF to be flattened into a plenoptic mosaic. Plenoptic mosaics have successfully been used for interactive rendering on GPUs [36] as they can be stored as textures and passed through a programmable graphics pipeline. What remains to decide is how exactly to discretize the scene surfaces and the aperture - what discretization grids to use. Given that a surface light field is simply a view dependent texture, one option is to take advantage of existing workflows and techniques from standard texturing [52]. We can consider unwrapping all surfaces in our scene to a normalized texture space and rasterizing the resulting projections. This also allows for good control over how densely different parts of a given surface will be sampled. Conceptually, we can even think of projecting the entire discontinuous SLF into a single texture space, but with each individual surface mapped to its own sub-region, just like in a texture atlas [44]. This analogy makes the 2D nature of the positional data even more obvious. The aperture surface can also be mapped to a normalized texture space. Without loss of generality, we can assume having a square aperture, which can subsequently be masked with any desired bokeh shape. To form our SALF mosaic, the aperture image for each scene point can be stored as a microimage/super-texel at a position corresponding to the texel covering the given scene point. Filtering and mipmapping the resulting texture is a topic that can be explored further.

With a SALF representation suitable for GPU rendering, achieving real-time performance for generalized focusing effects is a conceivable goal. Previously presented light field rendering on GPUs [36] has been output-centric and oriented towards fragment shader implementations. This approach works fine when the light field samples that can contribute to a given output pixel are known beforehand and are simply looked up during rendering. With generalized focusing, any point in the scene can potentially contribute to any output pixel. While such extreme defocusing would rarely be practical, an implementation that is capable

of handling such cases would have to scan through all scene points for potential radiance contributions to all output pixels. If the amount of defocusing is limited to some reasonable levels, spatial locality could be used to narrow down the search. Alternatively, input-centric approaches can be considered, too. As an example, standard depth of field effects have been demonstrated in real time using surface sample splatting [31]. A very similar approach can be explored for use in SALF refocusing, where individual aperture images can be splatted onto the final output image.

4.3 Practical Case Studies

The renderers that I propose can be applied in a variety of cases, whenever two core requirements are met. First of all, a model for the viewed scene has to be present. Ideally, this would be a parametric model, on which surface light field samples can be mapped and related to each other. This mapping would specify the positional components of the SLF. In addition, the directional components of the SLF also need to be mapped correctly with respect to the reference frame of the given scene point. For this to happen, the directional samples either need to be synthesized (e.g., from global illumination simulations), or have to be reconstructed (e.g., from calibrated photography). I next discuss three practical settings, in which these two conditions (having a scene model; and having calibrated directional data) can often be met, allowing for my generalized image formation model to be applied.

4.3.1 CG Rendered Photography

On one hand, physically based rendering (PBR) provides the ideal environment for testing my algorithms. Having exact scene models with assigned physically based materials allows for precise radiance computations that can provide ground truth renditions. At the same time, my image formation model has practical application within the PBR field itself. Physically-accurate, generalized focusing has not previously been demonstrated even in a virtual scene setting. As was discussed earlier in Section 4.2, I could either go for a ray-traced implementation, which would minimally change the existing wide-aperture camera models in a PBR engine; or, I could use the engine to precompute the relevant surface light field data, which could then be used for interactive refocusing. Both of these approaches remain to be implemented and tested. At the moment, Blender Cycles [5] is my primary candidate for testing these two implementations.

4.3.2 Photography of 3D Scanned and Multi-view Reconstructed Static Scenes

Obtaining a surface light field from a real scene is a difficult task that could involve the use of intricate reconstruction and calibration algorithms, often not applicable to general scenes and requiring dense input imagery. The most simplified case is to consider static scenes, with limited spatial extent, and with only opaque, textured objects in them. For cases like this, a much more complete dataset can be recorded and vision algorithms can be used more robustly. The first papers on surface light fields [3, 60, 24] considered an even more special case, of scanning and reconstructing isolated objects. While this can give valid SLFs for my renderers, DoF effects are more practical when the scene is comprised of more than one object. At the same time, a scanned SLF is valid only when lighting in the scene does not change, which prohibits freely composing new scenes from individually captured SLFs.

Still, larger scenes can be reconstructed using computer vision techniques. Large bodies of work exist on techniques like structure from motion and shape from X [55]. Calibration algorithms also exists and even auto-calibration is possible [55] for multi-view data. Most importantly, a lot of commercial 3D photo-modeling software, seamlessly offering such functionality, is becoming available (e.g., Autodesk ReCap). Even obtaining the starting multi-view imagery is becoming automated through the use of off-the-shelf unmanned aerial systems (drones). For my purposes, the only limitation in current 3D reconstruction software is that only a single diffuse texture gets mapped to the final reconstructed surface. To reconstruct an SLF, the acquired directional components from the starting multi-view data need to be mapped to the scene reconstruction, too. This should be an easy step that can be done on top of the standard output form a 3D photo-modeler.

4.3.3 Plenoptic Camera Photography

Obtaining surface light fields from dynamic and open scenes is also an approachable task, through the use of commercial plenoptic cameras [42, 46]. More precisely, it is a surface-aperture light field that can be captured with a single plenoptic camera snapshot. Since the range of views, obtainable with a single shot, is limited by the size of the main lens aperture, the scene reconstruction task becomes even more difficult. As was discussed in Section 3.3.1, because of this, depth layers are the most commonly used scene model in plenoptic rendering. Scene depth can approximately be inferred from the captured imagery by using depth extraction techniques [55, 58, 4, 19, 56]. Usually this takes the form of some generalization of stereo disparity matching and produces a disparity map for some chosen central sub-aperture view. The disparity map can subsequently be transformed to surfaces in the scene, but only when both the camera's intrinsic parameters (focal lengths and relative positions of the main lens and microlenses) and extrinsic parameters (scene-space position and orientation) are known. Some work has been done on inferring these parameters through calibration procedures [9, 6]. At least one work [6] claims to have results that are good enough for scene reconstruction. The main problem is that neighboring views captured with a plenoptic camera exhibit different imaging geometries, as they share the same, stationary sensor. This leaves additional global parameters that need to be optimized for, which, in turn, produces less robust results.

Given that standard light field rendering can be done directly from a disparity map [15, 13] a much simpler approach would be to consider disparity-space equivalents to my generalized focusing algorithms. The mapping from scene-space surface points to camera-space focal points is linear (a depth-dependent perspective warp); and the mapping from camera-space focal distances to microimage disparities can also be shown to be a linear warp. With this in mind, we can expect that ray casts, and thus SLF propagation, are valid in disparity-space.

I am currently working with a Lytro Illum [42] camera, which comes with software for extracting a color-encoded disparity map. The software (Lytro Desktop) also generates a hexagonal grid of seven sub-aperture images. The values in the disparity map could potentially be decoded to actual disparities for a set of recognized features in these seven views. Still, the disparity map is valid for just one central view of the scene, and other views will observe disocclusion gaps, which need to be filled in. Lytro Desktop has its own, undisclosed, solution to the problem, but other solutions have been proposed in published literature, too [26, 39, 51, 61].

5 Dissertation Plan

5.1 Proposed Contributions and Remaining Work

In summary, I propose a new physically-based, computational image formation model, which allows for generalized focusing, in addition to incorporating standard wide-aperture image formation as a special case. My proposed work draws upon results from both standard light field photography and surface light field reconstruction and rendering. I show that SLFs are suitable as a starting point for artifact-free, generalized image formation. At the same time, inspired by existing GPU-based light field renderers, I propose a new, mixed light field parameterization - the surface-aperture light field (SALF), which can potentially be used for real-time, physically-accurate, generalized refocusing.

The proposed renderers remain to be tested in practice and in several different areas. I plan on implementing a ground truth renderer, based on distributed ray tracing. For special cases, the results from this renderer can be compared against standard renditions, achievable with a wide-aperture camera model. At the moment, I intend on modifying Blender Cycles for this purpose, although any comparable engine would do. Having some ground truth, ray-traced renditions will also allow me to evaluate the quality of my renditions from precomputed SLFs. I also plan to test my renderers on real-world data using captured SLFs. 3D photo-modeling software (e.g., Autodesk ReCap/ReCap 360) can provide me with scene reconstructions and view calibrations from multi-view imagery. This is a good starting point for reconstructing SLFs for static scenes. To cover dynamic scenes, I plan to use plenoptic camera data, from a Lytro Illum.

I will also aim at achieving a real-time implementation, based on SALF mosaics, for images with Lytro's resolution. Finally, I will consider a texture-paint-based interface for specifying per-point focal offsets, i.e. a focus brush implementation.

5.2 Timeline

Task	**Timeline**
Consider a Paper	Fall 2015
Implement Renderers: - Lytro Renditions - Blender GE / Focus Brush - Cycles Implementations - 3D Photo-modeling Results	Now - May 2016
Thesis Writing	Fall 2015 - Fall 2016
Thesis Defense	Fall 2016

References

[1] E. Adelson and J. Bergen. The plenoptic function and the elements of early vision. In *Computational Models of Visual Processing*, pages 3–20. MIT Press, Cambridge, MA, 1991.

[2] E. Adelson and J. Wang. Single lens stereo with a plenoptic camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 99–106, 1992.

[3] Daniel Azuma. Surface Light Fields Project Page : http://grail.cs.washington.edu/projects/slf/.

[4] Tom E. Bishop and Paolo Favaro. Full-resolution depth map estimation from an aliased plenoptic light field. In Ron Kimmel, Reinhard Klette, and Akihiro Sugimoto, editors, *Computer Vision - ACCV 2010 - 10th Asian Conference on Computer Vision, Queenstown, New Zealand, November 8-12, 2010, Revised Selected Papers, Part II*, volume 6493 of *Lecture Notes in Computer Science*, pages 186–200. Springer, 2010.

[5] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2014.

[6] Yunsu Bok, Hae-Gon Jeon, and In So Kweon. Geometric calibration of micro-lens-based light-field cameras using line features. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2014.

[7] J. Chai, S. Chan, H. Shum, and X. Tong. Plenoptic sampling. *ACM Trans. Graph.*, pages 307–318, 2000.

[8] Wei-Chao Chen, Jean-Yves Bouguet, Michael H. Chu, and Radek Grzeszczuk. Light field mapping: Efficient representation and hardware rendering of surface light fields. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 447–456, New York, NY, USA, 2002. ACM.

[9] Donald G. Dansereau, Oscar Pizarro, and Stefan B. Williams. Decoding, calibration and rectification for lenselet-based plenoptic cameras. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '13, pages 1027–1034, Washington, DC, USA, 2013. IEEE Computer Society.

[10] Abe Davis, Marc Levoy, and Fredo Durand. Unstructured light fields. *Comput. Graph. Forum*, 31(2):305–314, 2012.

[11] Paul Debevec, Yizhou Yu, and George Boshokov. Efficient view-dependent image-based rendering with projective texture-mapping. Technical Report UCB/CSD-98-1003, EECS Department, University of California, Berkeley, 1998.

[12] Cass Everitt. Interactive order-independent transparency, 2001.

[13] T. Georgiev and A. Lumsdaine. Reducing plenoptic camera artifacts. *Computer Graphics Forum*, 29(6):1955–1968, 09/2010 2010.

[14] T Georgiev, K Zheng, B Curless, D Salesin, and et al. Spatio-angular resolution tradeoff in integral photography. *Proc. Eurographics Symposium on Rendering*, Jan 2006.

[15] Todor Georgiev and Andrew Lumsdaine. Focused plenoptic camera and rendering. *J. Electronic Imaging*, 19(2):021106, 2010.

[16] Todor Georgiev and Andrew Lumsdaine. The multifocus plenoptic camera, 2012.

[17] Todor Georgiev, Andrew Lumsdaine, and Georgi Chunev. Using focused plenoptic cameras for rich image capture. *IEEE Computer Graphics and Applications*, 31(1):62–73, 2011.

[18] Todor Georgiev, Andrew Lumsdaine, and Sergio Goma. Plenoptic principal planes. In *Imaging and Applied Optics*, page JTuD3. Optical Society of America, 2011.

[19] Bastian Goldluecke. Globally consistent depth labeling of 4d light fields. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 41–48, Washington, DC, USA, 2012. IEEE Computer Society.

[20] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. *ACM Trans. Graph.*, pages 43–54, 1996.

[21] V. Guillemin and S. Sternberg. *Symplectic Techniques in Physics*. Cambridge University Press, 1990.

[22] Lou Harrison, David McAllister, and Martin Dulberg. Stereo Computer Graphics for Virtual Reality. In *SIGGRAPH '97 Course Notes 6*. Multimedia Lab Department of Computer Science North Carolina State University, 1997.

[23] Aaron Isaksen, Leonard McMillan, and Steven J. Gortler. Dynamically reparameterized light fields. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 297–306, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[24] Jan Jachnik, Richard A. Newcombe, and Andrew J. Davison. Real-time surface light-field capture for augmentation of planar specular surfaces. In *Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, ISMAR '12, pages 91–97, Washington, DC, USA, 2012. IEEE Computer Society.

[25] David Jacobs, Jongmin Baek, and Marc Levoy. Focal stack compositing for depth of field control. Technical Report 2012-01, Stanford University Computer Science, Jan 2012.

[26] Vincent Jantet, Christine Guillemot, and Luce Morin. Joint projection filling method for occlusion handling in depth-image-based rendering. *3D Research*, 2(4), 2011.

[27] G. Kim. *Designing Virtual Reality Systems: The Structured Approach*. Springer London, 2007.

[28] Robert Kosara, Silvia Miksch, and Helwig Hauser. Semantic depth of field. In Keith Andrews, Steven F. Roth, and Pak Chung Wong, editors, *INFOVIS*, pages 97–104. IEEE Computer Society, 2001.

[29] Todd J. Kosloff and Brian A. Barsky. An algorithm for rendering generalized depth of field effects based on simulated heat diffusion. In *International Conference on Computational Science and Its Applications*, 2007.

[30] Todd J. Kosloff and Brian A. Barsky. Three techniques for rendering generalized depth of field effects. In *Proceedings of the Fourth SIAM Conference on Mathematics for Industry: Challenges and Frontiers (MI09)*, pages 42–48, October 2009.

[31] Jaroslav Křivánek, Jiří Žára, and Kadi Bouatouch. Fast depth of field rendering with surface splatting. In *Computer Graphics International, CGI 2003. Proceedings*, pages 196 – 201, july 2003.

[32] Jed Lengyel and John Snyder. Rendering with coherent layers. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 233–242, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[33] Marc Levoy. Light fields and computational imaging. *Computer*, 39(8):46–55, August 2006.

[34] Marc Levoy and Pat Hanrahan. Light field rendering. *ACM Transactions on Graphics*, pages 31–42, 1996.

[35] Chia-Kai Liang, Yi-Chang Shih, and Homer H. Chen. Light field analysis for modeling image formation. *IEEE Transactions on Image Processing*, 20(2):446–460, 2011.

[36] Andrew Lumsdaine, Georgi Chunev, and Todor Georgiev. Plenoptic rendering with interactive performance using gpus. In *Proc. SPIE 8295, Image Processing: Algorithms and Systems X; and Parallel Processing for Imaging Applications II*, pages 829513–829513–15, 2012.

[37] Andrew Lumsdaine and Todor Georgiev. The focused plenoptic camera. In *IEEE International Conference on Computational Photography (ICCP)*. IEEE, 2009.

[38] David F. McAllister, editor. *Stereo Computer Graphics: And Other True 3D Technologies*. Princeton University Press, Princeton, NJ, USA, 1993.

[39] Leonard McMillan. Image-based rendering using image warping. In *COMPUTER GRAPHICS Proceedings, Annual Conference Series, ACM SIGGRAP, New Orleans, Louisiana*, 2009.

[40] Jacob Munkberg, Robert Toth, and Tomas Akenine-Möller. Per-Vertex Defocus Blur for Stochastic Rasterization. *Computer Graphics Forum*, 2012.

[41] R Ng, M Levoy, M Bredif, G Duval, M Horowitz, et al. Light field photography with a hand-held plenoptic camera. Technical Report 2005-02, Stanford University Computer Science, Jan 2005.

[42] Ren Ng. The Lytro Camera: https://www.lytro.com.

[43] Ren Ng. *Digital light field photography*. PhD thesis, Stanford University, Stanford, CA, USA, 2006. Adviser-Patrick Hanrahan.

[44] Nvidia. Improve Batching Using Texture Atlases. (July), 2004.

[45] James Pearson, Mike Brookes, and Pier Luigi Dragotti. Plenoptic layer-based modeling for image based rendering. *IEEE Transactions on Image Processing*, 22(9):3405–3419, 2013.

[46] Christian Perwass and Lennart Wietzke. The Raytrix Camera: http://www.raytrix.de/.

[47] Christian Perwass and Lennart Wietzke. Single lens 3d-camera with extended depth-of-field. pages 829108–829108–15, 2012.

[48] Matt Pharr and Greg Humphreys. *Physically Based Rendering, Second Edition: From Theory To Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edition, 2010.

[49] Kari Pulli, Michael Cohen, Tom Duchamp, Hugues Hoppe, Linda Shapiro, and Werner Stuetzle. View-based rendering: Visualizing real objects from scanned range and color data. In *In Eurographics Rendering Workshop*, pages 23–34, 1997.

[50] R.J. Rost, B.M. Licea-Kane, D. Ginsburg, J.M. Kessenich, B. Lichtenbelt, H. Malan, and M. Weiblen. *OpenGL Shading Language*. Pearson Education, 2009.

[51] M. Schmeing, X.Y. Jiang, M. Schmeing, and X.Y. Jiang. Depth image based rendering: A faithful approach for the disocclusion problem. In *3DTV10*, pages 1–4, 2010.

[52] D. Shreiner, G. Sellers, J.M. Kessenich, and B.M. Licea-Kane. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.3*. OpenGL. Pearson Education, 2013.

[53] H.Y. Shum, S.C. Chan, and S.B. Kang. *Image-Based Rendering*. Springer US, 2008.

[54] J. Stewart, J. Yu, S. J. Gortler, and L. McMillan. A new reconstruction filter for undersampled light fields. In *Proceedings of the 14th Eurographics Workshop on Rendering*, EGRW '03, pages 150–156, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[55] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

[56] Michael W. Tao, Sunil Hadap, Jitendra Malik, and Ravi Ramamoorthi. Depth from combining defocus and correspondence using light-field cameras. December 2013.

[57] Kartik Venkataraman, Dan Lelescu, Jacques Duparré, Andrew McMahon, Gabriel Molina, Priyam Chatterjee, Robert Mullis, and Shree Nayar. Picam: An ultra-thin high performance monolithic camera array. *ACM Trans. Graph.*, 32(6):166:1–166:13, November 2013.

[58] S. Wanner. *Orientation Analysis in 4D Light Fields*. 2014.

[59] Bennett Wilburn, Neel Joshi, Vaibhav Vaish, Eino ville Talvala, Emilio Antunez, Adam Barth, Andrew Adams, Mark Horowitz, and Marc Levoy. High performance imaging using large camera arrays. *ACM Trans. Graph*, pages 765–776, 2005.

[60] Daniel N. Wood, Daniel I. Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H. Salesin, and Werner Stuetzle. Surface light fields for 3d photography. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 287–296, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[61] Zhan Yu, Jingyi Yu, Andrew Lumsdaine, and Todor Georgiev. Plenoptic depth map in the case of occlusions, 2013.