Fast Incident Light Field Acquisition and Rendering

Ivo Ihrke
MPI Informatik, Germany
ihrke@mpi-sb.mpg.de

Timo Stich
TU Braunschweig, Germany
stich@graphics.tu-bs.de

Heiko Gottschlich
TU Braunschweig, Germany
gottschlich@graphics.tu-bs.de

Marcus Magnor
TU Braunschweig, Germany
magnor@graphics.tu-bs.de

Hans-Peter Seidel
MPI Informatik, Germany
hpseidel@mpi-sb.mpg.de

ABSTRACT

We present a practical and inexpensive approach for the acquisition and rendering of static incident light fields. Incident light fields can be used for lighting virtual scenes or to insert virtual objects into real world video footage. The virtual objects are correctly lit and cast shadows in the same way as real objects in the scene. We propose to use an inexpensive planar mirror and a high dynamic range video camera to record incident light fields quickly, making our method suitable for outdoor use. The mirror serves as a moving virtual camera sampling the light field. To render with the acquired data we propose a hardware accelerated rendering algorithm that reproduces the complex lighting and shadows of the 4D light field. The algorithm is scalable and allows continuous trade between quality and rendering speed.

Keywords: Light Fields, Illumination, Image-Based Rendering, Reflectance and Shading, Image-Based Lighting

1 INTRODUCTION

As the boundaries of traditional photography are shifted light fields become an increasingly important tool in visual modeling. Light fields are used as object representations, replacing geometric descriptions of appearance [LH96, GGSC96]. They are also used to represent light sources [GGHS03a, HKSS98], compute synthetic apertures [WJV$^+$05] and refocus images after they have been taken [NLB$^+$05].

In this paper we focus on capturing the lighting environment in a scene and illuminating virtual objects with real light. We acquire static incident light fields in a couple of minutes using a light weight system consisting of a laptop, a mirror and a USB-high dynamic range video camera. This makes our system applicable to indoor as well as outdoor scenes under constant illumination. We light virtual objects and place them into real environments using a newly developed hardware accelerated rendering algorithm based on orthographic projective texture mapping and shadow mapping. The proposed rendering algorithm includes an importance sampling scheme allowing continuous trade between quality and rendering speed.

Lighting scenes with light fields has received little attention due to non-trivial acquisition and time consum-

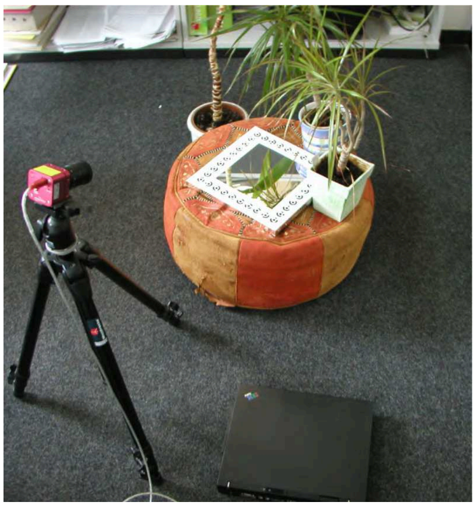

Figure 1: Our recording setup consists of a HDR video camera, an acquisition laptop and a planar optical front surface mirror with a frame of rotationally invariant binary coded patterns.

portable. In [UWH+03a] an array of mirroring spheres is presented. This approach suffers from resolution problems since only one image is acquired.

We propose to use a static video camera and a moving mirror to acquire static light fields. The advantages of this method are flexibility, affordability and robustness. The acquisition is nearly as flexible as for SfM-approaches, however the implementation is much simpler because calibration source code is freely available, e.g. [Bou05]. Our method is as robust as standard camera calibration methods and the only devices necessary are a camera and a planar mirror, resulting in a portable setup. Acquisition times are usually less than 5 minutes.

The closest approach to our rendering algorithm is [HKSS98]. They were the first to propose the use of projective texturing and shadowmaps to render simulated light sources using graphics hardware. However, their results are computed per vertex and limited to phong shading. We show how the special case of orthographic projective texturing gives the advantage of handling more complex BRDFs. Our proposed per direction rendering is further extended to include importance sampling to achieve interactive frame rates. Another related hardware accelerated algorithm is reported in [GGHS03b] for real world light sources represented in an optical filter basis. The application to general light field rendering is however limited as the optical basis is optimized for the representation of a single light source.

Most of the approaches used in light field lighting apply ray tracing techniques e.g. [UWH+03b, GGHS03a]. They usually employ photon-mapping or Monte-Carlo integration. The complexity of these rendering approaches make interactivity hard to achieve.

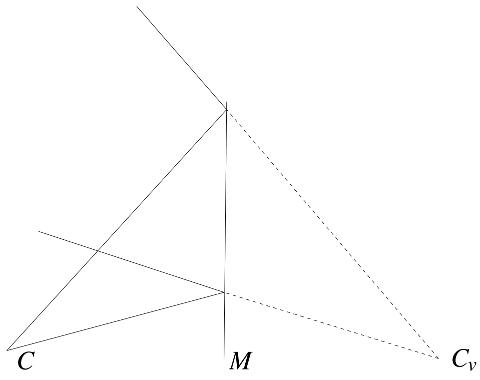

Figure 2: Light Field sampling using a planar mirror and a static camera: The moving mirror M causes different viewpoints to be seen by the virtual camera C_v.

In the derivation of our algorithm we discuss a per vertex rendering algorithm which is closely related to the approach from [NRH04] where precomputed scenes are relit at interactive frame rates.

3 CAPTURING INCIDENT LIGHT FIELDS

The basic idea of our light field sampling technique is shown in Figure 2. We use a static camera to observe a planar mirror in the scene. The pixels on the mirror correspond to virtual viewpoints behind the mirror plane. When moved around in the scene, the mirror generates a number of virtual viewpoints which are used to sample the light field. This results in a non-uniform sampling of the light field similar to Structure-from-Motion (SfM) based techniques [PGV+04, LA03]. and essentially realizes a dynamic catadioptric camera system with multiple centers of projection. The advantage over SfM-based techniques is two-fold. Firstly we achieve a calibration accuracy as in methods using a known calibration target [Zha99] which is higher than in SfM-approaches because the uncertainty in the feature points' 3D positions does not influence the calibration parameters. Secondly the mandatory bundle adjustment process [TMHF00] involves significantly fewer free parameters allowing for larger problems to be solved.

3.1 Acquisition Setup

Our hardware setup for capturing incident light fields is shown in Figure 1. The camera is a Photonfocus Hurricane 40 one mega-pixel HDR video camera with 12 bit A/D conversion and a programmable response curve. It can record up to 37 frames per second depending on the exposure time required to capture the image. The mirror is an optical front surface foil mirror originally intended for use in rear projection screens. These mirrors are inexpensive and are available in sizes up to several square meters. In order to track the mirror through the acquired video sequences we add a specially designed

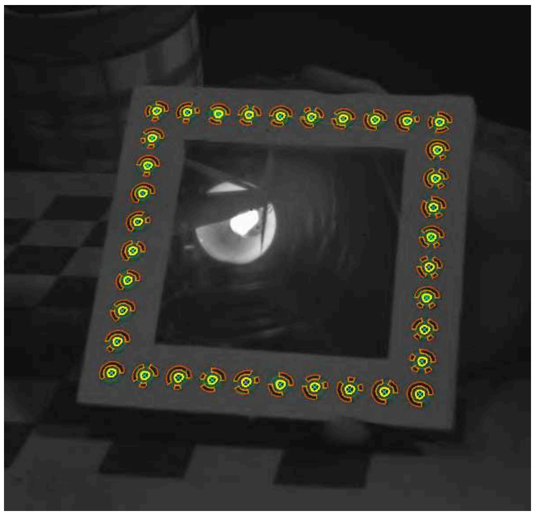

Figure 3: The tracking process: We find the rotationally invariant features (correctly identified features are shown in orange) and track them through the video sequence. The features encode positions on the mirror plane. This allows us to compute a homography between mirror coordinates and image coordinates which in turn facilitates camera calibration.

frame consisting of self-identifying features. The feature patterns are rotationally invariant and encode unit positions in the mirror coordinate system. The design of the patterns is inspired by the work of Forbes et al. [FVB02].

3.2 Tracking

In order to calibrate the mirror planes with respect to the static camera we track the frame with the binary encoded positions of the mirror coordinate system through the video sequence. The features encode their positions in the mirror coordinate system. This allows us to uniquely identify a feature point even if it is leaving and re-entering the field of view of the camera.

The tracking is performed using a combination of robust homography estimation using RANSAC [FB81] and non-linear optimization with the Levenberg-Marquardt method [MNT04]. We

1. identify initial feature positions,
2. estimate the mirror plane to screen homography using RANSAC [HZ00],
3. update the guessed feature positions using the estimated homography and
4. perform a non-linear optimization using Levenberg-Marquardt to refine the ellipse positions of the features' centers, shown in yellow in Figure 3.

The features are then tracked to the next frame using a cross-correlation measure between adjacent frames. The resulting positions are used as an initial guess for step 1 of the feature detection process in the next frame.

3.3 Geometric Calibration

The homographies computed in the previous subsection are used to compute the geometric calibration of the video sequence. We use Zhang's calibration method [Zha99] which relies on planar calibration targets. The homographies can be directly used to compute a closed form solution of the internal and external calibration parameters. These initial guesses are refined by a global optimization step. We use a sparse implementation of the Levenberg-Marquardt (LM) algorithm similar to the sparse bundle adjustment presented in [LA04].

It is very important to employ an efficient implementation of the global optimization step because the number of free parameters becomes quite large with a higher sampling rate of the light field. In the LM-iterations we compute only the non-zero entries of the Jacobian matrix $\mathbf{J}$ of the cost function, i.e. the projection error function. The iteration involves the solution of a linear system which is performed using the sparse iterative solution method CGLS (Conjugate Gradients for Least Squares) [Han98]. The CGLS method avoids computing $\mathbf{J}^T\mathbf{J}$ for the normal equations explicitly which can be computationally expensive and memory consuming in its own right.

The calibration process results in a camera internal parameter matrix $\mathbf{K}$ and for each frame of the video sequence a pose $\mathbf{R}_i, \mathbf{t}_i$ of the camera with respect to the mirror plane. The mirror plane is defined to lie in the x,y-plane. This resembles a fixed calibration pattern with a moving camera. World points $\mathbf{x}$ project to image coordinates x_i in the ith video frame via

$$x_i = \mathbf{K}[\mathbf{R}_i \mid \mathbf{t}_i]\mathbf{x}. \tag{1}$$

Since the camera is static we use the camera coordinate system as the frame of reference for our calibration. We mirror the camera pose at the mirror plane:

$$\mathbf{M}_i = \begin{pmatrix} \mathbf{R}_i & | \, \mathbf{t}_i \\ 0 & | \, 1 \end{pmatrix} \tag{2}$$

$$\mathbf{F} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

$$\hat{\mathbf{M}}_i = \mathbf{M}_i\mathbf{F}\mathbf{M}_i^{-1}, \tag{4}$$

to yield the projection matrices.

$$\hat{\mathbf{P}}_i = \mathbf{K}\hat{\mathbf{M}}_i^{-1} \tag{5}$$

for the virtual views generated by the mirror. The mirror operation correctly changes the handedness of the coordinate system for the virtual views. The matrices $\hat{\mathbf{P}}_i$ allow the projection of world coordinates into the virtual views. We use this fact in Section 4.3 to resample the light field into a uniform representation.

3.4 Photometric Calibration

For rendering with incident light fields a linear camera response is needed, i.e. the pixel values should be proportional to the measured radiance. We use a high dynamic range video camera for our measurements. These cameras in general exhibit a non-linear response to the incoming radiance. Pixel values are compressed using a logarithmic function prior to A/D conversion. Our camera allows a programmable compression function. To obtain linearized images the camera response function has to be measured.

We use the method of Robertson et al. [RBS03] to estimate the camera response curve. The different exposure times required by [RBS03] are simulated using optical neutral density filters [KGS05]. The recovered response curve is then applied to the input images.

4 RENDERING INCIDENT LIGHT FIELDS

Lighting with light fields is a very time consuming task in general due to the large amount of data that must be processed. As the computational power of GPUs and hardware increases, lighting with light fields becomes more and more tractable. However, making the most use of the processing power available is an important property when designing such rendering algorithms. We like to pronounce that the proposed rendering algorithm poses no restrictions onto the scene content, material properties or lighting.

Lighting from light fields, as many rendering approaches, can be derived from the rendering equation for a point $\mathbf{x}$ in euclidean space,

$$L(\omega_o, \mathbf{x}) = \int_\Omega f_\mathbf{x}(\omega_i, \omega_o) L_i(\mathbf{x}, \omega_i) v(\mathbf{x}, \omega_i)(N \cdot \omega_i)\, d\omega_i \tag{6}$$

were $f_\mathbf{x}$ is the bidirectional reflectance function (BRDF), L_i denotes the incoming radiance and v the visibility for each direction. In the following sections we will discuss two rendering approaches aiming at implementing this equation as efficiently as possible on graphics hardware using recent OpenGL extensions.

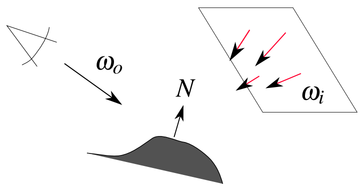

Figure 4: Visualization of the per vertex lighting approach. In each iteration for one vertex all directions of the light field are integrated (red).

4.1 Per Vertex Lighting

Our first approach solves the directional integral of Equation 6 for one vertex, for all directions of the light field simultaneously. We loop over all vertices to compute the solution. Translated to meshes and hardware accelerated shading, this can be achieved by rendering a hemisphere from each vertex' point of view, multiplying with the incoming light field and BRDF, and interpolating in-between.

The algorithm can be implemented completely in graphics hardware by rendering a hemisphere or cube map from the vertex' point of view and blending with textures for the BRDF and the incoming radiance interpolated from the light field. Although this approach is able to produce nice results (see Figure 5), it has several drawbacks. For example, the input meshes must be densely sampled in order to reproduce fine shadowing details. Furthermore, a cube map per vertex has to be re-computed from the incident light field data and uploaded to the graphics card when changing the relative pose of the object with respect to the incident light or in case of rendering dynamic objects. Our implementation of this algorithm, with the integration and rendering of the hemispheres implemented on the GPU, revealed that the rendering time per vertex is still too high to scale to satisfying mesh resolutions. The scene depicted in Figure 5 has 8000 faces and took already about 15s to compute.

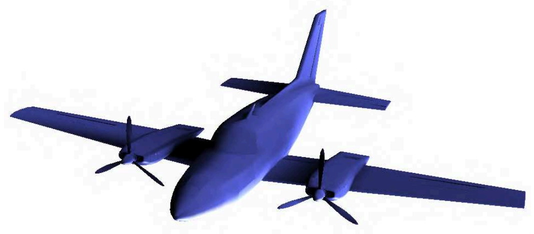

Figure 5: Rendering result of a per vertex light field illumination approach. Although the overall impression of the lighting is plausible, the shadow lacks detail due to insufficient mesh tessellation.

4.2 Per Direction Lighting

Since the per vertex approach proved computationally expensive, we investigated the dual approach, involving lighting computations per direction for all fragments simultaneously. Different rendering passes are then performed for every lighting direction. Equation 6 is thus solved simultaneously for one direction and all points in the model, similar to [HKSS98]. The complexity of this approach scales with the number of directions and the output quality is less dependent on the tessellation of the meshes. This idea translates to hardware accelerated rendering through projective texturing [SKvW$^+$92]. Visibility testing is straight forward to implement via shadow maps [Wil78]. Generally,

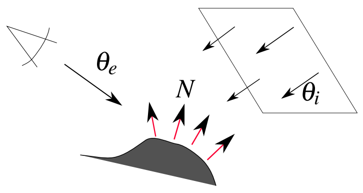

Figure 6: Visualization of the per direction lighting. In each iteration for all fragments one direction of the light field is integrated (red).

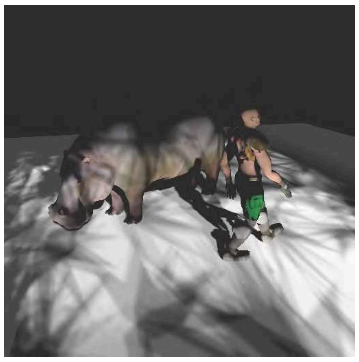

Figure 7: Rendering result of per direction light field illumination. Fine shadow details and complex lighting details are visible.

incident light fields have a higher spatial than directional resolution which matches the hardware capabilities, since high texture resolutions are less expensive to handle than more rendering passes. The detailed description of our algorithm is stated in Section 5. We will further show how importance sampling of the directions gives a good control over rendering speed vs. quality in Section 5.1.

Since light fields have inherently a very high dynamic range (HDR), it is necessary to make use of HDR rendering. Fortunately, recent GPUs have a floating point pipeline and support hardware accelerated rendering to floating point textures and their blending.

4.3 Light Field Resampling

In order to use our acquired, non-uniform light field data with the proposed rendering algorithm we have to resample it into a uniform light field representation. For this purpose we apply a variant of Unstructured Lumigraph Rendering [BBM+01].

As a first step we define a light field sampling plane as our proxy geometry. We use a plane parallel to the ground plane but specifying a more complex proxy is also possible. As a next step we project the boundaries of the virtual views, i.e. the region surrounded by the calibration frame, onto the light field sampling plane. This allows us to compute a bounding area in which the resampling process is performed. We choose a rectangular bounding area and sample it uniformly yielding a number of sample positions on the light field plane. We employ Unstructured Lumigraph Rendering to render orthographic views of the unstructured light field data

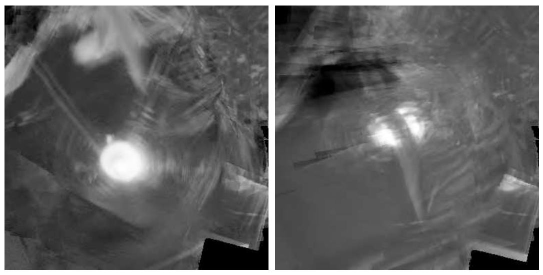

Figure 8: Two of 9057 resampled light field directions. The images are gamma corrected to show the details in the acquired light field. 15 original camera views are weighted for every pixel in the resampled images.

obtained in the acquisition process for each directional component of the resampled light field. The directions are distributed according to a subdivided icosahedron to ensure a uniform directional sampling of the hemisphere.

For every direction being resampled the implementation performs the following steps:

1. project the uniform samples into all virtual views
2. compute the field of view penalty
3. look up the intensity value if the projected sample falls into the field of view
4. determine the angular penalty by computing the angle between the direction currently being resampled and the direction from the sample position towards the center of projection of the virtual view
5. compute and normalize the weights from the penalty values
6. compute the resampled light field value

Two examples of a directionally resampled light field are shown in Figure 8. The light field contains high dynamic range data which has been re-mapped for visualization purposes. In the next section we discuss the lighting of virtual scenes with acquired incident light fields.

5 IMPLEMENTATION

Our proposed rendering algorithm is conceptually simple and can be efficiently implemented on graphics hardware. All our rendering results are computed in half float precision to account for the high dynamic range content when lighting with light fields.

We render two passes per direction of the light field. The first pass produces the shadowmap for the direction. In the second pass a skewed orthographic projective texturing is computed for the actual lighting. Results of the iterations are accumulated directly in GPU memory.

Our implementation is based on OpenGL using custom GLSL shaders. Placement and sizing of the light field is modeled via a rectangular shape in the scene. Since we do not rely on precomputation we are able to render fully animated scenes such as presented in the accompanying videos.

Since the incoming light in one lighting iteration is parallel with the orthographic projection, ω_i in equation 6 is constant for each iteration. This property allows the implementation of a variety of materials in the lighting pass efficiently. We implemented three BRDF models, which are diffuse shading, Blinn shading and others via cubemaps.

The rendering speed of our algorithm is mainly dependent on two limiting factors. One factor is the size of the output images, because our rendering algorithm is heaviliy dependent on the fragment shader performance. The most important limitation however, is the number of light field directions that can be handled, since each additional light field direction adds two more rendering passes which can not be parallelized. In the next section we explore a sampling technique which chooses the most important lighting directions for the current view and scene and helps to find a good trade-off between rendering quality and speed.

5.1 Importance Sampling

The most limiting speed factor of our implementation is the number of directions used for the lighting computation. To reduce the impact of this limitation we propose an importance sampling technique to allow for trading image quality and speed. The idea to importance sampling is to find a measure of the influence of each light field direction on the final rendering result and to use the most important ones for rendering. Generally the light contributing to the fragments is dependent on the incoming light field direction and the BRDF of the fragment. This property can be reformulated from Equation 6 to

$$I(\omega_o,\omega_i) = \int_X f_{\mathbf{x}}(\omega_o,\omega_i) L_i(\mathbf{x},\omega_i) v(\mathbf{x},\omega_i)(N \cdot \omega_i)\, d\mathbf{x}. \tag{7}$$

To speed up the computation of the importance I we propose some simplifications. First we assume each fragment has a diffuse BRDF $f_{\mathbf{x}}(\omega_o,\omega_i) = \frac{1}{2\pi}$ which results in

$$I(\omega_o,\omega_i) \approx \int_X L_i(\mathbf{x},\omega_i) v(\mathbf{x},\omega_i)(N \cdot \omega_i)\, d\mathbf{x}. \tag{8}$$

Further, we assume each $\mathbf{x}$ is always visible, i.e. $v(\mathbf{x},\omega_i) = 1$, and that the amount of light for each direction in the light field can be approximated by a constant $l_{\omega_i} \approx L_i(\mathbf{x},\omega_i)$. This results in our final approximation

$$I(\omega_o,\omega_i) \approx \int_X l_{\omega_i} \cdot \omega_i \cdot N\, d\mathbf{x}. \tag{9}$$

We like to stress that all approximations introduced above are only used to get a fast estimation of the importance I for each light field direction and do not restrict the generality of our rendering algorithm.

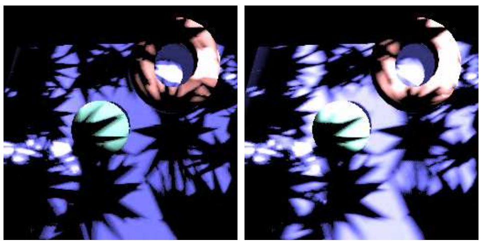

Figure 9: The rendering quality increases with the number of samples (100 vs 200).Our importance sampling strategy ensures an intelligent selection process for a continuous trade between rendering quality and speed.

The computation of I is implemented as an extension to the previously proposed per-direction rendering algorithm. One rendering pass for each frame in the animation sequence is added, where the normals of each fragment are rendered to a float texture with a custom shader. We then download the normal map onto the CPU and process the importance sampling as described above in parallel with the GPU. The computation takes only a few milliseconds for 9000 directions and normal map sizes of 64×64. By ordering the available directions according to their importance, we can impose a best first approximation of the final results by processing only the N most important directions in the light field.

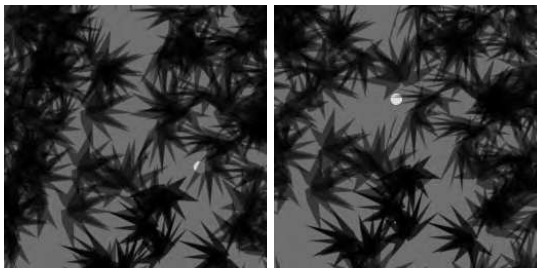

Figure 10: Sample images of the synthetic light field generated with a ray tracer. The resolution is 256×256 texels per light field direction.

6 RESULTS

We implemented our approach on a Linux workstation with a AMD64 dual-core processor and 2 GB of memory. The GPU is a NVIDIA GeForce7800 GTX. Our implementation is based on the OpenSceneGraph library [osg], which allows us to render animated scenes modeled with standard 3D modeling tools such as Maya. In the scenes, the light field plane is represented by a rectangle, to easily allow for placement. Table 1 shows the timings for different scenes and the number of directions used in the renderings.

# directions	100	1000	5000
car (8402 prim.)	0.02s	0.28s	1.9s
complex (18096 prim.)	0.03s	0.32s	2.3s

Table 1: Render timings for the recorded light field and different scenes. The spatial light field resolution is 128×128 Texels per light field direction. The resulting animations are shown in the accompanying video.

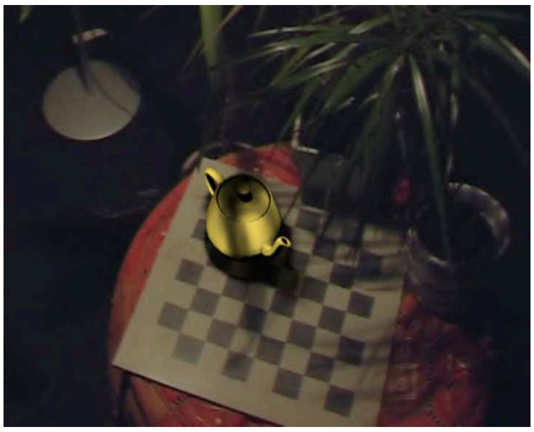

Figure 11: Real world scene with augmented teapot. The shadows of the recorded light field are faithfully rendered on the modeled geometry. Shadows cast by the teapot are generated by the recorded light source and blended with the background image.

Since the rendered light fields have 100 to 10000 directions, the produced shadows are very realistic without the need of special techniques such as percentage closer filtering [RSC87].

6.1 Incident Light Fields

The synthetic light field was generated by ray tracing a scene with one directional and two additional point light sources. Two samples from the light field are depicted in Figure 10.

For our real-world test data we acquired a sequence of 2200 high dynamic range images sampling a scene containing a checkerboard lit by a desk lamp with leaves of office plants casting shadows onto the checkerboard. The acquisition time was 3 minutes. We also recorded a camcorder sequence that was calibrated using the checkerboard in the scene. The calibration of the sequence has a mean reprojection error of ≈ 0.5 pixels. The non-uniform light field data was resampled into 9057 directions covering the area of the checkerboard and the spatial resolution per direction was 128×128 pixels. We render the virtual model from the tracked camera's point of view using a virtual ground plane that acts as a shadow receiver. The rendering result is composited into the original camera images. As shown in the images, Figure 11, the shadows of the real scene are convincingly reproduced on the teapot. The spatially varying incident lighting is captured very well. Additional results can be found in the accompanying video.

7 CONCLUSIONS AND FUTURE WORK

We have presented a pipeline for the acquisition and rendering of incident light fields. The presented algorithms allow for fast acquisition and rendering of complex real world lighting scenarios. Our acquisition scheme requires only a couple of minutes to record an unstructured incident light field. It combines several desirable properties of a light field acquisition method. The hardware setup is easily portable and lends itself to indoor as well as outdoor acquisition. The calibration accuracy is on par with fixed camera setups and the calibration pattern is not visible in the light field images.

Our rendering algorithm does not impose restrictions on geometry, animation or material properties of the scene. We can trade off rendering accuracy against speed and achieve interactive frame rates at lower quality settings. This is desirable for previewing and planning animations under complex illumination.

The main aspect for future work is an investigation into the sampling scheme in the light field acquisition phase. Because the mirror is a hand-held device it would be desirable to have an on line system to help the user choosing sample directions of the light field that have not been covered so far. This kind of system requires an on line tracking and calibration approach. Further the acquisition approach could be developed into a cheap scanner for standard light fields using a web cam and a standard mirror. It needs to be seen if the noise characteristics of web cams permit this kind of use.

On the rendering side we plan to investigate the possibility of avoiding the light field resampling step while maintaining the rendering speed and the quality control.

REFERENCES

[BBM+01] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured Lumigraph Rendering. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2001)*, pages 425 – 432, 2001.

[Bou05] Jean-Yves Bouguet. Camera Calibration Toolbox for Matlab, 2005.

[FB81] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[FVB02] Keith Forbes, Anthon Voigt, and Ndimi Bodika. An Inexpensive, Automatic and Accurate Camera Calibration Method. In *Proceedings of the Thirteenth Annual South African*

Workshop on Pattern Recognition. PRASA, 2002.

[GGHS03a] Michael Goesele, Xavier Granier, Wolfgang Heidrich, and Hans-Peter Seidel. Accurate Light Source Acquisition and Rendering. In *SIGGRAPH '03: Proceedings of the 30rd annual conference on Computer Graphics and Interactive Techniques*, pages 621–630, 2003.

[GGHS03b] Xavier Granier, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Interactive Visualization of Complex Real-World Light Sources. In *Proceedings of Pacific Graphics*, pages 59–66, 2003.

[GGSC96] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen. The Lumigraph. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 1996)*, pages 43–54, 1996.

[Han98] Per Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems*. Society of Industrial and Applied Mathematics, 1998.

[HKSS98] Wolfgang Heidrich, Jan Kautz, Philipp Slusallek, and Hans-Peter Seidel. Canned Lightsources. In *Rendering Techniques*, pages 293–300, 1998.

[HZ00] Richard Hartley and Andrew Zisserman. *Multiple View Geometry*. Cambridge University Press, 2000.

[KGS05] Grzegorz Krawczyk, Michael Goesele, and Hans-Peter Seidel. Photometric Calibration of High Dynamic Range Cameras. Research Report MPI-I-2005-4-005, Max-Planck-Institut für Informatik, May 2005.

[LA03] Manolis I.A. Lourakis and Antonis A. Argyros. Efficient 3D Camera Matchmoving Using Markerless, Segmentation-Free Plane Tracking. Technical Report 324, FORTH, Sept 2003.

[LA04] Manolis I.A. Lourakis and Antonis A. Argyros. The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm. Technical Report 340, FORTH, Aug 2004.

[LH96] Marc Levoy and Pat Hanrahan. Light Field Rendering. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 1996)*, pages 31–42, 1996.

[MNT04] K. Madsen, H. B. Nielsen, and O. Tingleff. Methods for non-linear least squares problems, 2nd edition, 2004.

[NLB$^+$05] Ren Ng, Marc Levoy, Mathieu Brédif, Gene Duval, Mark Horowitz, and Pat Hanrahan. Light Field Photography with a Hand-held Plenoptic Camera. Research Report CTSR 2005-02, Stanford University, May 2005.

[NRH04] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. Triple product wavelet integrals for all-frequency relighting. *ACM Trans. Graph.*, 23(3):477–487, 2004.

[osg] OpenSceneGraph. http://www.openscenegraph.org/.

[PGV$^+$04] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. In *International Journal of Computer Vision*, volume 59, pages 207–232, Sept 2004.

[RBS03] Mark A. Robertson, Sean Borman, and Robert L. Stevenson. Estimation-theoretic approach to dynamic range enhancement using mutliple exposures. *SPIE Journal of Electronic Imaging*, 12(2):219–228, April 2003.

[RSC87] William T. Reeves, David H. Salesin, and Robert L. Cook. Rendering antialiased shadows with depth maps. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 283–291, 1987.

[SKvW$^+$92] Mark Segal, Carl Korobkin, Rolf van Widenfelt, Jim Foran, and Paul Haeberli. Fast shadows and lighting effects using texture mapping. *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, 26(2):249–252, 1992.

[TMHF00] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

[UWH$^+$03a] J. Unger, A. Wenger, T. Hawkins, A.Gardner, and P. Debevec. Capturing and Rendering With Incident Light Fields. In *EGSR03 14th Eurographics Symposium on Rendering 2003*, pages 141–149, 2003.

[UWH$^+$03b] Jonas Unger, Andreas Wenger, Tim Hawkins, Andrew Gardner, and Paul Debevec. Capturing and Rendering with Incident Light Fields. In *Eurographics Symposium on Rendering (EGSR'03)*, pages 141–149, 2003.

[Wil78] Lance Williams. Casting curved shadows on curved surfaces. In *SIGGRAPH '78: Proceedings of the 6th annual conference on Computer graphics and interactive techniques*, pages 270–274, 1978.

[WJV$^+$05] 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. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2005)*, volume 24, pages 765–776, July 2005.

[Zha99] Zhengyou Zhang. Flexible Camera Calibration By Viewing a Plane From Unknown Orientations. In *International Conference on Computer Vision (ICCV'99)*, pages 666–673, Sept 1999.