

# Online Construction of Surface Light Fields

Greg Coombe<sup>1</sup>, Chad Hantak<sup>1</sup>, Anselmo Lastra<sup>1</sup>, Radek Grzeszczuk<sup>2</sup>

<sup>1</sup>University of North Carolina at Chapel Hill, <sup>2</sup> Intel Corporation

---

## Abstract

*We present a system for interactively capturing, constructing, and rendering surface light fields by incrementally building a low rank approximation to the surface light field. Each image is incorporated into the lighting model as it is captured, providing the user with real-time feedback. This feedback enables the user to preview the lighting model and direct the image acquisition towards undersampled areas of the object. We also provide a novel data-driven quality heuristic to aid the user in identifying undersampled regions. Our system is an order of magnitude faster than previous systems, and reduces the time necessary to capture the images and construct a surface light field from hours to minutes.*

Categories and Subject Descriptors (according to ACM CCS): I.4.1 [Digitization and Image Capture]: Reflectance

---

## 1. Introduction

Collecting the numerous images needed for the construction of surface light fields is a time-consuming and tedious process. Since the result can be viewed only after a lengthy post-process is complete, it can be difficult to determine when the light field is sufficiently sampled. It is not enough to uniformly sample the hemisphere, as this may miss high-frequency information such as highlights. Often, uncertainty about the sampling density leads users to capture many more images than necessary in order to guarantee adequate coverage. If undersampling artifacts are visible in the result, more images must be acquired and the entire factorization post-process must be repeated. These data-acquisition problems can be traced to the lack of feedback during the image capture process, as humans are quite adept at recognizing undersampling errors. An incremental algorithm provides the user with feedback as the lighting model is being constructed, allowing the user to obtain the necessary quality and avoid taking many extra pictures. Examples of several models can be seen in Figure 1.

In this paper we present a system for incrementally capturing, constructing, and rendering a surface light field with fixed illumination conditions and known geometry. Each image is incorporated into the lighting model as it is captured, providing the user with real-time feedback. This feedback enables the user to preview the surface light field and direct the image acquisition towards undersampled regions.



**Figure 1:** A heart figurine, a marble pestle, and a copper pitcher captured and rendered with our online system.

We also introduce a novel data-driven quality heuristic to highlight these areas.

We were inspired by Rusinkiewicz [RHHL02], who described a system to interactively capture geometry. The user was incorporated into the processing loop, which enabled

them to view the sampling and to steer the solution to eliminate holes in the model. Our goal is to make a similarly easy-to-use system to capture the reflectance properties of an object. Our online light-field capture process enables the user to examine the surface light field as it is being acquired, add images where necessary, and make the final determination as to whether the quality is sufficient. A sequence demonstrating this process is shown in Figure 2.

### 1.1. Contributions

We believe this paper makes the following contributions to image-based modeling and rendering.

- A fast, incremental algorithm to construct surface light fields. Since each image is processed as it is captured, the storage overhead is low, and new images can be incorporated at rates of more than one per second.
- An interactive modeling/rendering paradigm. As a new image is incorporated into the surface light field, the rendering data structures are immediately updated and displayed, enabling the user to continually evaluate the light field quality.
- A data-driven quality heuristic to guide sampling. An error metric is interactively computed and displayed to aid in determining which parts of the 4D space are undersampled.
- A structured method for dealing with incomplete data. Due to occlusion, many surface patches are only partially visible. Rather than discarding these data, we use the current approximation to fill in these holes.

The paper proceeds as follows. We first discuss image-based modeling methods for capture and display of view-dependent illumination. We then describe the Online SVD [Bra03], which is the core of our online approach. The Online SVD is a fast, memory-efficient algorithm for constructing an incremental low-rank singular value decomposition. In Section 4 we present details of the implementation, followed by results and conclusions.

## 2. Background

### 2.1. Surface Modeling

A good overview of the state of the art in material modeling by image acquisition, and potentially inverse rendering, is provided by the recent Siggraph course on Material Modeling [RM02], and the Eurographics State of the Art Report on Acquisition, Synthesis and Rendering of Bidirectional Texture Functions [MMS\*04]. Our system captures the exitant radiance of an object from images, and is based on BRDF capture systems [DvGNK99, LFTG97, MWL\*99, DHT\*00] and view dependent texture maps [LYS01].

Representation of this captured data is crucial for interactive rendering. Malzbender et al. [MGW01] fit acquired data to a bi-quadratic polynomial to estimate a reflectance



**Figure 2:** This sequence shows the image acquisition process. The left frame shows an image from the camera. In the middle is the reconstruction before this image is incorporated, with highlights interpolated from nearby views. The right image shows the reconstruction after it is incorporated, with the correct highlights.

field. Lensch et al. [LKG\*01] use the Lafortune [LFTG97] representation and clustered BRDFs from acquired data in order to create spatially-varying BRDFs. McAllister et al. [MLH02] describes a device for scanning 6D spatially varying BRDFs and methods for fitting the data to a Lafortune representation. Gardner et al. [GTHD03] describe a BRDF capture device that uses a linear light source (as opposed to a point source), which can also estimate surface normals and a height field.

### 2.2. Surface Light Fields

Surface light fields [MRP98] represent the exitant radiance under fixed illumination conditions on the surface of a known geometric model. This parameterization results in a compact representation that enables the capture and display of complex, view-dependent illumination of real-world objects. This category of approaches includes regular parameterizations of radiance [LH96, GGSC96] as well as approaches which handle very sparse and scattered samples [DTM96, DYB98, BBM\*01].

Surface light fields can be represented as the function  $f(s, t, \theta, \phi)$ . The variables  $s$  and  $t$  represent surface location on the mesh, and  $\theta$  and  $\phi$  represent view directions. This function describes the exitant radiance at every point on the surface from every direction. We can discretize this function over the surface patches and solid angles and represent it as a matrix. The columns of this matrix are the camera views, and the rows are the surface locations. Since storing these full data matrices would be impractical, several techniques have been developed to compress the data. Factorization approaches represent the 4D surface light field  $f(s, t, \theta, \phi)$  as a sum of products of lower-dimensional functions

$$f(s, t, \theta, \phi) \approx \sum_{r=1}^{\text{rank}} g(s, t) h(\theta, \phi)$$

The number of terms  $r$  is the rank of the approximation.

This factorization attempts to decouple the variation in surface texture from the variation in lighting. These functions can be constructed by using Principal Component Analysis [WAA\*00, CBCG02, NSI01] or non-linear optimization [HMG03, MAA01]. The function parameters can be stored in texture maps and rendered in real-time [CBCG02].

### 2.3. Online Methods

Most of the research in image-based modeling has focused on *batch-processing* systems. These systems process the set of images over multiple passes, and consequently require that the entire set of images be available. For detailed capture of light fields, this requires significant storage (around  $10^6$  data samples [HMG03]). In addition, incorporating additional images into these models requires recomputing the model from the beginning. Formulating surface light field construction as an *online processing* approach avoids these problems by incrementally constructing the model as the images become available. Matusik [MLP04] used this approach with a kd-tree basis system to progressively refine a radiance model from a fixed viewpoint. Schirmacher [SHS99] adaptively meshed the  $uv$  and  $st$  planes of a light field, and used an error metric along the triangle edges to determine the locations of new camera positions. Hillesland [HMG03] updated a non-linear solution in an online method, but required multiple passes over the data.

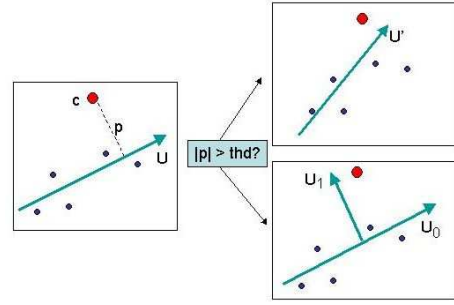
Our work has a strong machine learning component and depends on some of the recent algorithms developed in the context of data mining [Bra03, Row97].

### 3. Algorithm

The goal of this project is to build an interactive system that enables the rapid capture of exitant radiance under fixed illumination conditions. The user interacts with the system by moving a video camera around the object that is being captured. The video camera is tracked in real-time, and the images are incorporated into the model and displayed on the screen. This enables the user to view the surface light field as it is being constructed and to correct for undersampling errors. In addition, the user can view the data-driven quality heuristic as a scalar value for each surface patch, which provides additional statistics about the reconstruction quality. The key to this interaction is the method which we describe in this section, which can incrementally build a compressed representation of the surface light field.

#### 3.1. Online SVD

Chen et al. [CBCG02] use Principal Component Analysis [GL96] to extract the low-dimensional functions  $g(s,t)$  and  $h(\theta,\phi)$  from the full data matrices. PCA is a powerful and widely-used compression technique, but it requires that the full set of radiance data be available during processing. Since this data is usually extensively resampled to fit the



**Figure 3:** The Online SVD. A new point  $c$  is incorporated into the existing SVD. The orthogonal component  $p$  is computed by projecting onto  $U$ . If  $\|p\|$  is below a threshold, then we can incorporate this new sample by simply rotating  $U$ . Otherwise, we must increase the rank of the approximation.

columns and rows of the data matrices, this can be a significant storage cost.

The Online Singular Value Decomposition [Bra03] enables us to incrementally build a compressed representation of a surface light field. The Online SVD is an incremental PCA algorithm [HMM00, CMW\*97] that computes the principal eigenvectors of a matrix without storing the entire matrix in memory. The results are built up from a series of simple operations on the output eigenvectors, which are low-rank approximations to the full matrix. If the rank  $r$  is much smaller than the size of the matrices, this is a considerable saving, and reduces the computational complexity from quadratic to linear [Bra03].

The Online SVD works as follows (see Figure 3). Consider a rank- $r$  PCA

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

where  $\mathbf{U} \in \mathbb{R}^{m \times r}$ ,  $\mathbf{S} \in \mathbb{R}^{r \times r}$ , and  $\mathbf{V} \in \mathbb{R}^{n \times r}$ . As each new image is captured, it is resampled into a per-vertex column vector, which represents every pixel in a surface patch from one camera view. This column of samples  $c$  is projected onto the eigenspace:

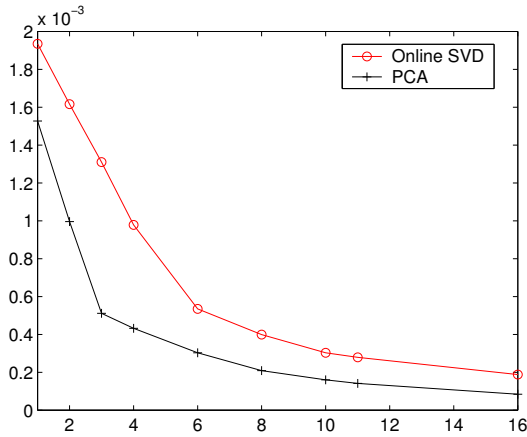
$$\mathbf{j} = \mathbf{U}^T c$$

The amount that is orthogonal to the eigenspace is given by:

$$\mathbf{p} = c - \mathbf{U}\mathbf{j}$$

The norm of this vector,  $\|p\|$ , is a measure of how different the pixels in this new image are from our current approximation. If the pixels are similar (that is,  $\|p\|$  is below a threshold) we can incorporate this new sample by simply rotating the existing eigenspaces.

$$\mathbf{U}' = \mathbf{U}\mathbf{R}_U \quad \mathbf{V}' = \mathbf{V}\mathbf{R}_V$$



**Figure 4:** The Mean Squared Error of a reconstructed light field as a function of the rank. The majority of the light field is captured after 4-6 terms. For the same number of terms, the Online SVD has more error than the PCA.

Otherwise, the current rank  $r$  of the approximation is insufficient, and we increase the rank to  $r + 1$  and append the column  $\mathbf{j}$  to our approximation.

$$\mathbf{U}' = [\mathbf{U}; \mathbf{j}] \mathbf{R}_U \quad \mathbf{V}' = \mathbf{V} \mathbf{R}_V$$

The rotations are computed by re-diagonalizing the  $(r + 1) \times (r + 1)$  matrix

$$\begin{bmatrix} \mathbf{S} & \mathbf{j} \\ 0 & \|\mathbf{p}\| \end{bmatrix} \rightarrow [\mathbf{R}_U, \mathbf{R}_V]$$

These rotations can be computed in  $O(r^2)$ . Since only the output matrices  $\mathbf{U}$ ,  $\mathbf{S}$ , and  $\mathbf{V}$  are stored, this representation results in significant storage savings.

### 3.1.1. Error

Since the Online SVD is constructed incrementally, it tends to have more error than the PCA for the same rank. Figure 4 shows the error as a function of rank. A rank that is too low biases the Online SVD computation by forcing it to select sub-optimal eigenvectors. Brand [Bra03] suggests computing the Online SVD at twice the actual rank and only using the largest eigenvalues. This allows the eigenvectors more degrees of freedom to fit to the incoming data.

### 3.2. Missing Values

Acquired radiance data is often incomplete because of occlusion. Many systems are forced to discard surface patches with missing data, or fill in the holes with incorrect values, such as zeros or mean values. A better approach is to estimate the missing data using a process known as *imputation*. Imputation uses the current Online SVD estimate of the



**Figure 5:** The left image shows the bust model constructed from 282 pictures, and the right image shows the same set of images using imputation to fill in the missing values. Areas with missing data have been colored red (they would normally be black).

light field to fill in missing values [Bra03]. The known samples are projected onto the current eigenspace, and the unknown values are estimated by solving the under-determined system using Linear Least Squares [GL96]. This fills in the missing values with the nearest plausible values using the Mahalanobis metric (a metric defined in the scaled eigenspace) [Bra03].

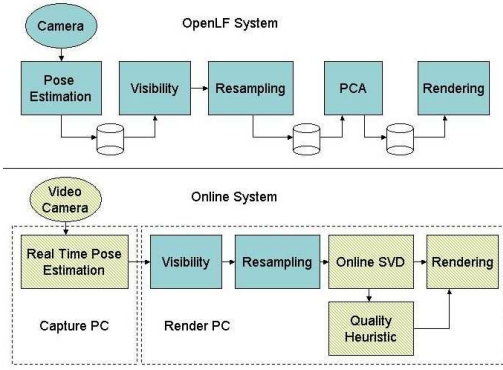
Figure 5 shows the advantage of imputation for surface light fields. In practice, we only impute missing values when at least half of the surface patch is visible in an image. In addition, we only impute values after 8-10 initial images, to allow the system to establish a reasonable approximation.

### 3.3. Data-Driven Quality Heuristic

The Online SVD enables the user to view the light field model as it is being captured. This visual feedback is helpful, but it can still be difficult to recognize undersampling errors during image acquisition. To aid the user in collecting high-quality radiance information, we developed a data-driven quality heuristic that uses the information obtained from the estimate to indicate whether more data are needed. This is displayed as the user views the model, and provides additional statistics about the reconstruction quality.

One possible way to do this is to assume a fixed BRDF and measure the error between this and the collected samples. This is the approach taken by Lensch [LLSS03], who used an uncertainty minimization technique to guide image acquisition. We wanted to avoid this fixed BRDF assumption in order to capture a wide variety of reflectance properties.

There are two statistics that we want to provide feedback about: the variation over the surface, and the variation



**Figure 6:** Top: The OpenLF system. Bottom: Our online system. The two systems share several components, including the visibility, resampling, and rendering. We introduce several new components to convert light field construction into an online process.

over the viewing direction. To this end, we developed a per-triangle scalar quality heuristic that is computed as the combination of two quality functions:

$$\Psi(s, t, \theta, \phi) = \omega \sqrt{\Psi_p(s, t)^2 + \Psi_h(\theta, \phi)^2}$$

The surface quality function  $\Psi_p$  measures the quality of the surface approximation by tracking the projection error of the Online SVD. The projection error is computed as part of the updating step as the quantity  $\|\mathbf{p}\|$ . This value is a measure of how much the new image differs from our approximation. This scalar quantity is smoothed using an exponential fall-off filter.

The hemisphere quality function  $\Psi_h$  is a measure of the sampling density of the hemisphere. The value is computed by using the areas of the triangles in the Delaunay triangulation of the hemisphere. The Delaunay triangulation, which is used for interpolation, is described in more detail in Sec. 4.2.

The quality heuristic  $\Psi(s, t, \theta, \phi)$  is displayed as a scalar value at each triangle. We display this value in red in our system, since it is easily visible from a distance. As the user moves the tracked camera around the object, the object rotates on the screen and the brightness of the heuristic changes. As images are captured in a region, the heuristic darkens and the user moves to a different area. As the light field is being acquired, the user can either view the light field, the heuristic, or a split-screen view of both. In practice, we typically view the camera output and tracking information on one screen and switch between the heuristic and the light field on the other.

## 4. Implementation

We have implemented this system within the framework of OpenLF [Opeb], an open source light field capture and display system developed by Intel. In Figure 6, we show a diagram of OpenLF and our prototype system. This section describes the shared elements of the system: visibility, resampling, and rendering.

### 4.1. Rendering

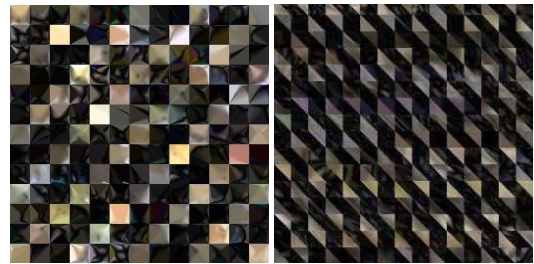
One of the advantages of the factorization approach to surface light fields is that the functions  $g(s, t)$  and  $h(\theta, \phi)$  can be stored in texture maps and rendered in real-time [CBCG02]. The functions  $g(s, t)$  are known as *surface maps* and  $h(\theta, \phi)$  are known as *view maps*.

The surface locations  $(s, t)$  are parameterized over the triangle rings centered at each vertex. Using triangle rings instead of individual triangles avoids discontinuities at the edges [CBCG02], but requires that each triangle be represented by three surface maps (one per vertex  $v$ ). The camera locations  $(\theta, \phi)$  are parameterized over the hemisphere above each vertex.

The rendering algorithm, which is executed on graphics hardware, requires evaluating the following function:

$$f_v(s, t, \theta, \phi) = \sum_{r=1}^{\text{rank}} \sum_{v=1}^3 \beta(s, t, v) h_v^r(\theta, \phi) g_v^r(s, t)$$

at each of the triangles. A view vector is computed from the eye to each of the three vertices of the triangle. These vectors are projected onto the local basis system, and interpolated across the triangle. At each fragment, the vectors are normalized and used to look up in the view maps  $h_{(1,2,3)}^r(\theta, \phi)$ . The functions  $g_{(1,2,3)}^r(s, t)$  are computed by linearly interpolating the surface maps across the triangle. The two functions are multiplied together and weighted by a barycentric function  $\beta(s, t, v)$  to get the final exitant radiance at this point. Each term  $r$  of the approximation is rendered in one pass and accumulated in a floating-point frame buffer.



**Figure 7:** Left: The viewing functions  $h(\theta, \phi)$  are parameterized over the hemisphere and stored in viewmaps. Right: The surface functions  $g(r, s)$  are parameterized over the triangles and stored in surface maps.

The surface maps and view maps for all of the triangles are tiled into larger texture maps, as shown in Figure 7. We typically use surface maps of size 16x16 and view maps of size 32x32. All textures are stored in floating point format, which avoids quantization error and scale/bias artifacts. This requires more storage than the 8-bit textures used in OpenLF [Opeb], but the signed representation and high dynamic range are important as the principal components can be either negative or positive.

#### 4.2. Visibility and Resampling

The matrix factorization approach to compression of surface light fields requires that the image data be resampled from the input images to fit into the rows and columns of the data matrices. This resampling occurs in both dimensions; not only are the camera locations at arbitrary locations in the hemisphere, but the projected areas of the triangles vary in each image.

Resampling the images to fill the columns of the data matrix is straightforward. First, the visibility is tested by projecting the triangle mesh using the camera's position and orientation. Visibility is computed on a per-vertex basis by back-projecting the vertex into the camera. If a triangle ring is determined to be visible, the colors are sampled from the input image using bilinear interpolation.

Resampling the camera views to cover the hemisphere is more complicated. In the OpenLF system, camera locations are treated as points in a Delaunay triangulation. This triangulation is used to weight each of the camera positions, which are stored in the texture maps. Our system uses a similar approach by building the Delaunay triangulation incrementally. The triangulation is initialized with four points at the corners of the view map. When a new image is captured, the camera location is projected onto the vertex basis vectors and normalized to get a point on the hemisphere. This point is projected down onto the plane and inserted into the Delaunay triangulation. The triangulation is then rendered using these points as the colors, and the graphics hardware resamples the colors across the hemisphere. The result is stored in the view map.

#### 4.3. Pose Estimation

In order to project the image samples onto the geometry, the camera's position and orientation must be known. Our system uses a tracked video camera to capture images of the object. The camera was calibrated with Bouguet's Camera Calibration Toolbox [Bou], and images are rectified using Intel's Open Source Computer Vision Library [Opea]. To determine the pose of the camera with respect to the object, a stage was created with fiducials along the border. The 3D positions of the fiducials are located in the camera's coordinate system in real-time using the ARToolkit Library [ART]. This library uses image segmentation, corner extraction, and



**Figure 8:** The buddha model processed using PCA, on left, and the Online SVD, on the right. Online SVD is an approximation, and there are some visible artifacts in the chest area. The PSNR between the images is 33.2dB.

matching techniques for tracking the fiducials. Knowing the 3D position of an imaged fiducial allows the pose of the camera to be computed. When multiple fiducials are present in an image, the camera pose can be refined by minimizing the differences between the pose estimates for each fiducial.

### 5. Results

Our system uses two PCs, one for camera tracking and one for visibility, resampling, and rendering. The tracking PC, a 1.8GHz Intel Pentium 4, is connected to a Point Grey Research Flea VIDEO camera via a IEEE-1394 interface. This camera captures  $1024 \times 768$  color images at 30 frames per second. The images and pose estimates are sent over the network to the second PC, a 2.3GHz AMD Athlon64, where they are incorporated into the matrix factorization and rendered. The geometry of the object is captured as a preprocess using a Faro digitizing arm. The model is registered with the camera by sampling the fiducial locations with the same tool.

#### 5.1. Results

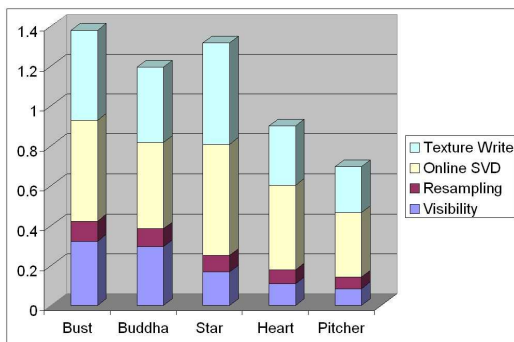
One of the advantages of our method is the reduced storage cost, which translates to improved speed of model construction. In our tests, this resulted in an order of magnitude increase in speed over the OpenLF system. For the buddha dataset shown in Figure 8, the processing time is reduced from 67 minutes in OpenLF to 7 minutes in our system. Timings for several datasets are presented in Figure 9.

As we noted in Sec. 3.1.1, the Online SVD can have more error than the PCA. We ran several experiments to compare the quality of the Online SVD to a batch PCA. Using a stored dataset, the images were fed one at a time through the Online

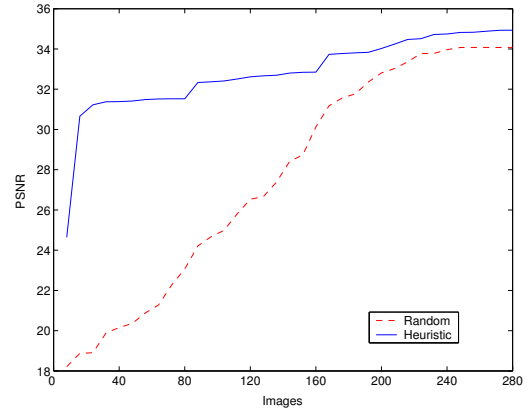
SVD solver to mimic the conditions of an incremental construction. Since this was implemented within the OpenLF framework, we can directly compare the quality of the reconstruction. A side-by-side-comparison of two images is shown in Figure 8.

During image acquisition, the data-driven heuristic provides visual feedback to the user to highlight undersampled areas. We have found this works well in practice, but we are interested in formally evaluating the usefulness of this heuristic. In lieu of a user study, we developed an automated experiment to measure the convergence of a surface light field using the heuristic. Using an archived dataset, we compared random image selection to image selection guided by the heuristic. To add a new image, the computer calculates the heuristic from the point-of-view of all of the remaining images, and selects the image with the highest error. The results of this experiment are shown in Figure 10. This experiment indicates that the heuristic can accurately predict which images will contribute the most content to the final result. Capturing images where the heuristic is large should likewise enable users to efficiently capture light fields. We are interested in conducting a user study to evaluate how well this heuristic information is communicated to the user.

One factor that can affect the quality of incremental approximations is dependence upon the ordering of data. In a different experiment, we measured the error over different permutations of a light field dataset. The results of this experiment are shown in Figure 11. For reference, we included the ground truth estimate from PCA and a sorted permutation of the samples. Note that the worst error is from the sorted permutation. Brand [Bra03] points out that small, gradual rotations tend to introduce floating-point error, which causes the eigenvectors to lose orthogonality. He suggests periodically re-orthogonalizing the matrices using Gram-Schmidt orthonormalization. In a different pa-



**Figure 9:** Timing results in seconds for several models. The bust has 7K triangles, the buddha has 12K triangles, the star has 5K triangles, the heart has 2.7K triangles, and the pitcher has 3K triangles.



**Figure 10:** Measuring the convergence of surface light field construction using the data-driven heuristic. Compared to random selection, the use of the data-driven heuristic dramatically speeds convergence. The PSNR of the reconstruction was compared to a reference PCA implementation at 10 random viewpoints.

per [Bra02], he provides an alternative matrix formulation that reduces this error.

## 6. Conclusions

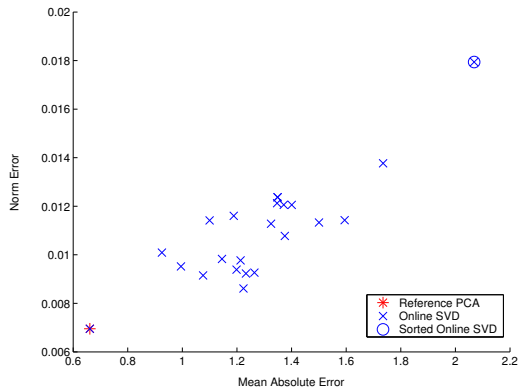
We have presented a method for incremental construction of surface light fields, which uses the Online SVD to build a surface light field approximation interactively during the acquisition stage. This online approach provides real-time feedback to the user, which enables the user to direct the image acquisition towards the undersampled areas of the model. This significantly reduces the acquisition time and helps build higher quality models. This approach can also decrease the processing time by an order of magnitude.

To assist the user in the capture process, we present a novel data-driven heuristic that provides an extra channel of information. We conducted several experiments to demonstrate that the data-driven quality heuristic, a tool for providing feedback to the user during sampling, can significantly increase the convergence of the reconstruction. Since the order in which the images are acquired affects the Online SVD solution, we also investigated the errors caused by permutations of the images.

We believe that incremental construction of surface light fields is a powerful tool for the capture and rendering of photorealistic models.

## 7. Future Work

In addition to the incremental construction, there are several properties of the Online SVD that are very useful for surface



**Figure 11:** The total reconstruction error of the Online SVD from 20 random permutations of the original dataset. The reference PCA is shown in the lower left corner. A permutation of the SVD where the samples were sorted by norm is shown in the upper right corner. This indicates that the error is greatest when the samples are correlated.

light fields. One property that we are interested in exploring is that the Online SVD can be “down-dated” to remove a previous value from the approximation. This would allow the user to undo mistakes that occur during the image acquisition process.

The timing results show that it takes about one second to incorporate a new image into the approximation. We are interested in decreasing this time by implementing the system on the GPU as in Hillesland [HMG03]. While this is possible for the visibility and resampling components of the process, it may be difficult to map the Online SVD matrix operations.

An interesting future approach is evaluating and improving the data-driven quality heuristic. In the current implementation, the heuristic is displayed as a scalar value, but it should be possible to use all of the color channels to convey more information about the sampling. We would also like to investigate other quality functions that take into account the incident light or the reflected light directions. In addition, we would like to formally show that the quality heuristic can decrease the number of images required.

Finally, one of the limitations of surface light fields is that the geometry must be known a priori. In our current system this forces us into a two-step process — first create a geometric model, then the surface light field. We are interested in investigating ways to collect geometry and color simultaneously. We are also interested in removing the restriction of fixed lighting and developing an incremental approach for acquisition of surface reflectance fields.

## References

[ART] ARTOOLKIT: [www.hitl.washington.edu/artoolkit/](http://www.hitl.washington.edu/artoolkit/).

- [BBM\*01] BUEHLER C., BOSSE M., MCMILLAN L., GORTLER S., COHEN M.: Unstructured lumigraph rendering. In *SIGGRAPH* (2001), pp. 425–432.
- [Bou] BOUGUET J.-Y.: Matlab Camera Calibration Toolbox. [www.vision.caltech.edu/bouguetj](http://www.vision.caltech.edu/bouguetj).
- [Bra02] BRAND M.: Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision* (2002), pp. 707–720.
- [Bra03] BRAND M.: Fast online svd revisions for lightweight recommender systems. In *SIAM International Conference on Data Mining* (2003).
- [CBCG02] CHEN W.-C., BOUGUET J.-Y., CHU M., GRZESZCZUK R.: Light field mapping: Efficient representation and hardware rendering of surface light fields. In *SIGGRAPH* (2002).
- [CMW\*97] CHANDRASEKAREN S., MANJUNATH B., WANG Y., WINKLER J., ZHANG H.: An eigenspace update algorithm for image analysis. In *Graphical Models and Image Processing* (1997).
- [DHT\*00] DEBEVEC P., HAWKINS T., TCHOU C., DUIKER H.-P., SAROKIN W., SAGAR M.: Acquiring the reflectance field of a human face. In *SIGGRAPH* (2000).
- [DTM96] DEBEVEC P. E., TAYLOR C. J., MALIK J.: Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *SIGGRAPH* (1996), pp. 11–20.
- [DVGNK99] DANA K., VAN GINNEKEN B., NAYAR S., KOENDERINK J. J.: Reflectance and texture of real world surfaces. In *ACM TOG* (1999).
- [DYB98] DEBEVEC P. E., YU Y., BORSHUKOV G. D.: Efficient view-dependent image-based rendering with projective texture-mapping. In *Eurographics Rendering Workshop* (1998), pp. 105–116.
- [GGSC96] GORTLER S. J., GRZESZCZUK R., SZELISKI R., COHEN M. F.: The Lumigraph. In *SIGGRAPH* (1996), pp. 43–54.
- [GL96] GOLUB G. H., LOAN C. F. V.: *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.
- [GTHD03] GARDNER A., TCHOU C., HAWKINS T., DEBEVEC P.: Linear light source reflectometry. In *SIGGRAPH* (2003).
- [HMG03] HILLESLAND K., MOLINOV S., GRZESZCZUK R.: Nonlinear optimization framework for image-based modeling on programmable graphics hardware. In *SIGGRAPH* (2003).
- [HMM00] HALL P., MARSHALL D., MARTIN R.: Merging and splitting eigenspace models. In *IEEE Trans. on Pattern Analysis and Machine Intelligence* (2000).
- [LFTG97] LAFORTUNE E. P. F., FOO S.-C., TORRANCE K. E., GREENBERG D. P.: Non-linear approximation of reflectance functions. In *SIGGRAPH* (1997), pp. 117–126.
- [LH96] LEVOY M., HANRAHAN P.: Light field rendering. In *SIGGRAPH* (1996).
- [LKG\*01] LENSCH H., KAUTZ J., GOESELE M., HEIDRICH W., SEIDEL H.: Image-based reconstruction of spatially varying materials. In *Eurographics Rendering Workshop* (2001).

- [LLSS03] LENSCH H. P. A., LANG J., SA A. M., SEIDEL H.-P.: Planned sampling of spatially varying brdfs. In *Workshop on Rendering* (2003).
- [LYS01] LIU X., YU Y., SHUM H.-Y.: Synthesizing bidirectional texture functions for real-world surfaces. In *SIGGRAPH* (2001).
- [MAA01] MCCOOL M. D., ANG J., AHMAD A.: Homomorphic factorizations of brdfs for high-performance rendering. In *SIGGRAPH* (2001), pp. 171–178.
- [MGW01] MALZBENDER T., GELB D., WOLTERS H.: Polynomial texture maps. In *SIGGRAPH* (2001).
- [MLH02] MCALLISTER D., LASTRA A., HEIDRICH W.: Efficient rendering of spatial bi-directional reflectance distribution functions. In *Graphics Hardware* (2002).
- [MLP04] MATUSIK W., LOPER M., PFISTER H.: Progressively-refined reflectance functions from natural illumination. In *Eurographics Symposium on Rendering* (2004).
- [MMS\*04] MUELLER G., MESETH J., SATTLER M., SARLETTE R., KLEIN R.: Acquisition, synthesis and rendering of bidirectional texture functions. In *Eurographics* (2004), State of the Art Reports.
- [MRP98] MILLER G. S. P., RUBIN S. M., PONCELEON D.: Lazy decompression of surface light fields for precomputed global illumination. In *Eurographics Workshop on Rendering* (1998).
- [MWL\*99] MARSCHNER S. R., WESTIN S. H., LAFORTUNE E. P. F., TORRANCE K. E., GREENBERG D. P.: Image-based BRDF measurement including human skin. In *Eurographics Workshop on Rendering* (1999).
- [NSI01] NISHINO K., SATO Y., IKEUCHI K.: Eigen-texture method: Appearance compression and synthesis based on a 3d model. *IEEE Trans. Pattern Anal. Mach. Intell.* 23, 11 (2001), 1257–1265.
- [Opea] OPENCV: OpenCV: The Open Computer Vision Library. [sourceforge.net/projects/opencvlibrary](http://sourceforge.net/projects/opencvlibrary).
- [Opeb] OPENLF: Openlf: The open lightfield library. [sourceforge.net/projects/openlf/](http://sourceforge.net/projects/openlf/).
- [RHH02] RUSINKIEWICZ S., HALL-HOLT O., LEVOY M.: Real-time 3d model acquisition. In *SIGGRAPH* (2002).
- [RM02] RAMAMOORTHY R., MARSCHNER S.: Acquiring material models using inverse rendering. In *SIGGRAPH Course 39* (2002).
- [Row97] ROWEIS S.: EM Algorithms for PCA and SPCA. In *Neural Information Processing Systems* (1997), pp. 626–632.
- [SHS99] SCHIRMACHER H., HEIDRICH W., SEIDEL H.-P.: Adaptive acquisition of lumigraphs from synthetic scenes. In *Computer Graphics Forum* (1999), vol. 18(3), pp. 151–160.
- [WAA\*00] WOOD D., AZUMA D., ALDINGER W., CURLESS B., DUCHAMP T., SALESIN D., STUETZLE W.: Surface light fields for 3d photography. In *SIGGRAPH* (2000).