Enhancing Light Fields through Ray-Space Stitching

Xinqing Guo*, Zhan Yu*, Sing Bing Kang, *Fellow, IEEE,* Haiting Lin, and Jingyi Yu

Abstract—Light fields (LFs) have been shown to enable photorealistic visualization of complex scenes. In practice, however, an LF tends to have a relatively small angular range or spatial resolution, which limits the scope of virtual navigation. In this paper, we show how seamless virtual navigation can be enhanced by stitching multiple LFs. Our technique consists of two key components: LF registration and LF stitching. To register LFs, we use what we call the *ray-space motion matrix (RSMM)* to establish pairwise ray-ray correspondences. Using Plücker coordinates, we show that the RSMM is a 5×6 matrix, which reduces to a 5×5 matrix under pure translation and/or in-plane rotation. The final LF stitching is done using multi-resolution, high-dimensional graph-cut in order to account for possible scene motion, imperfect RSMM estimation, and/or undersampling. We show how our technique allows us to create LFs with various enhanced features: extended horizontal and/or vertical field-of-view, larger synthetic aperture and defocus blur, and larger parallax.

Index Terms—Light Field Enhancement, Image Based Rendering

1 Introduction

A light field (LF) consists of a large collection of rays that store radiance information in both spatial and angular dimensions. It has been convincingly shown to be capable of high-quality visualization of complex scenes. Early approaches to capture LFs involve the use of camera arrays or sequence of closely sampled camera views. As an example, the Stanford light field array [1] uses a 2D grid composed of 128 1.3 megapixel firewire cameras that can stream live video to a striped disk array. Unfortunately, such systems are bulky and expensive, and hence not very practical. The alternative of using closely sampled views from a single camera [2] precludes the capture of dynamic scenes. More recent approaches emulate a multi-camera system by using a lenslet array. Commodity LF cameras [3] such as Lytro, Raytrix, and Pelican chip-based cameras are poised to be game changers on providing compact, inexpensive, and single-shot solutions.

To date, no LF capture technique is able to satisfy both high spatial and high angular requirements. LFs captured by a camera array have high spatial resolution, but its angular resolution is low. This is because of the large baseline between neighboring cameras (e.g., 3.3 cm in [1]). In contrast, LF cameras that can capture at a higher angular resolution (e.g., $14\mu m$ in Lytro and $207\mu m$ in Raytrix) have low spatial resolution. The problem is inherent to its design of using a 2D sensor to capture a 4D LF: the total number of pixels (rays) that can be captured is fixed and one has to trade off between spatial and angular domains [4]. Even though the Lytro camera has an 11 mega pixel sensor, it can only capture 0.11 million spatial samples and 100 angular samples on the plane of the lenslet array. The Raytrix R11 camera uses a 10.7 mega pixel sensor to capture 0.47 million spatial samples, and has 23 angular samples. Image super-resolution techniques [5]–[7] can ameliorate the resolution problem and are complementary to our work.

The effective baselines of LF cameras also tend to be much smaller than those of camera arrays. This is because the effective baseline in an LF camera is limited to the size of its aperture. Consequently, the captured LF generally exhibits a much smaller parallax. In practice, when using the LF camera for post-capture (virtual) refocusing, the user will need to position the camera close to the target object to acquire sufficient parallax for synthesizing noticeable defocus blur effects. As a result, it is difficult to capture the entire object within the camera's field-of-view (FoV) and simultaneously generate significant depth-based effects.

In this paper, we present a new technique to enhance an LF by merging multiple LFs captured using an LF camera. This is analogous to stitching multiple 2D images into a panorama. We first derive what we call the *ray-space motion matrix (RSMM)* that describes how LF ray parametrization are transformed under different LF coordinates. We show that under Plücker-type coordinates, the RSMM is a 5×6 matrix analogous to the projective transformation between two images. Further,

* *These authors contribute to the work equally.*

- *Xinqing Guo is with the Department of Computer and Information Sciences, University of Delaware, Newark, DE*
  E-mail: xinqing@udel.edu
- *Zhan Yu is with Adobe Systems Inc, San Jose, CA*
  E-mail: yshmzhan@udel.edu
- *Sing Bing Kang is with Microsoft Research, Redmond, WA*
  E-mail: sbkang@microsoft.com
- *Haiting Lin is with the Department of Computer and Information Sciences, University of Delaware, Newark, DE*
  E-mail: haiting@udel.edu
- *Jingyi Yu is the corresponding author with the Department of Computer and Information Sciences, University of Delaware, Newark, DE*
  E-mail: yu@eecis.udel.edu

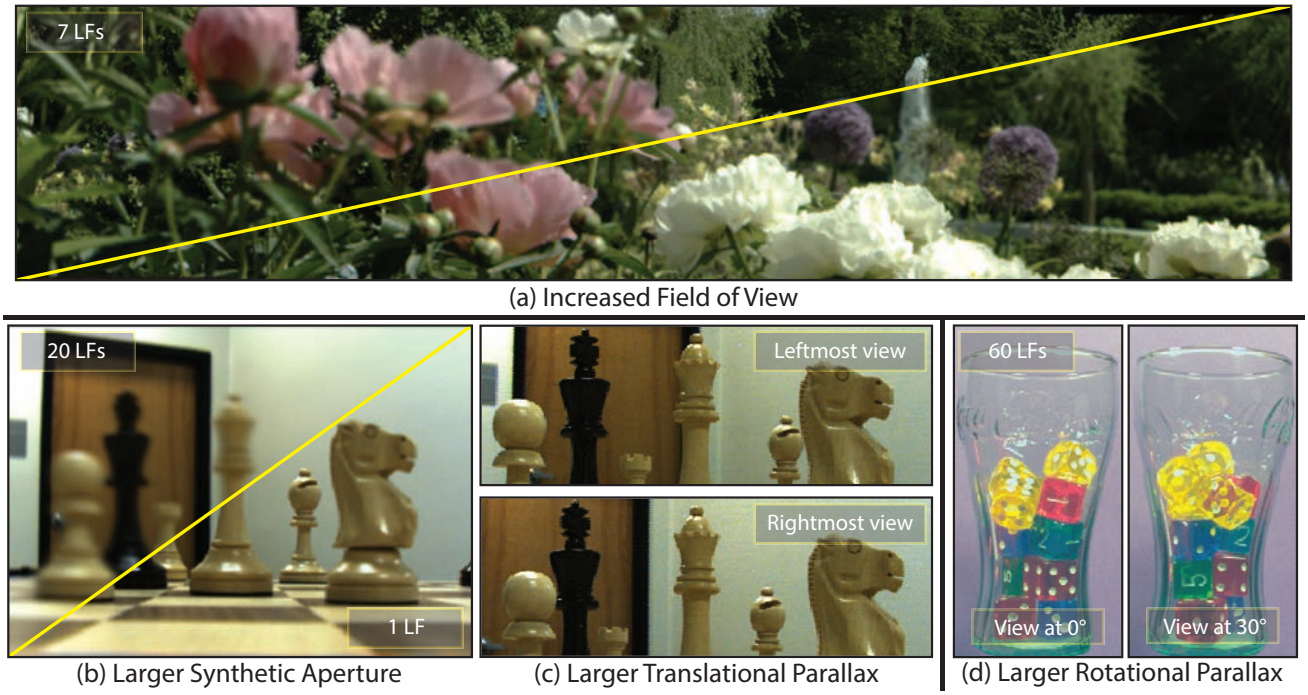


Fig. 1. Enhanced light field (LF) rendering using our LF stitching framework. Examples: (a) A stitched panoramic LF with a wide field of view (FoV); the upper/lower triangles show shallow depth of field (DoF) rendering on different focal planes. (b) and (c) A translation-based LF mosaic with increased virtual aperture size; the upper/lower triangles show enhanced/original synthetic defocus blurs and (c) shows larger parallax. (d) A rotation-based LF mosaic for illustrating view-dependency of transparent scenes.

we show how the RSMM is extracted through a high-dimensional correspondence matching step without explicit knowledge or estimation of camera motion. Note that extracting ground truth RSMM require exact ray-ray correspondences, which could be difficult due to undersampling and imperfect camera placement. Instead we recover an approximated RSMM from approximated ray-ray correspondences based on 2D SIFT matching. Once we have aligned the LFs using the estimated RSMMs, we then apply an additional stitching procedure as refinement to handle misalignments caused by errors in estimating RSMMs, slight scene motion between captures, and undersampling. This is analogous to stitching for 2D images [8], [9], whose solution may be based on graph-cut. Brute-force implementation of graph-cut on the 4D LF, however, is computationally expensive. We instead use a coarse-to-fine approach: we compute the cut on a coarse 4D graph; when upsampling the graph to a finer level, we effectively prune unnecessary nodes by using the estimated coarse cut. We demonstrate our LF stitching framework on creating LFs with various enhanced features: extended horizontal and/or vertical field-of-view, larger synthetic aperture and defocus blur, and larger parallax.

2 Related Work

Our work is related to image-based modeling and rendering (IBMR), and image stitching. In this section, we review representative approaches in these areas.

IBMR. Dense and uniform sampling of rays for LF rendering is not easy in practice. There are many techniques proposed to handle aliasing as a result of undersampling, especially in angular dimension. Approaches such as [10]–[13] pre-filter the LF with different kernels, but this is done at the cost of high frequency loss. The Lumigraph [2], [14] requires a sparser set of images because it relies on simple scene geometry proxy for ray interpolation. However, its rendering quality is highly dependent on the quality of the proxy. In addition, the input views are assumed to be accurately calibrated. As a result, the Lumigraph is less effective for highly non-Lambertian scenes. As an image-based modeling tool, LFs have shown promising results in stereo matching and 3D reconstruction [15]–[18].

Extending LFs is not without a precedent. The system described in [19] uses a large collection of omnidirectional images to render views within a large space. The input images are linked via correspondences (hence implicitly geometry). The idea presented in [20] is to "stitch" differently captured LFs (views captured along a 1D path, concentric mosaics), but the stitching and transition is based on heuristics. Lehtinen et al. [21] exploit the anisotropy in the temporal light field to significantly accelerate depth-of-field rendering. Our approach capitalizes on the commercial availability of LF cameras: we capture a sparse set of LFs using one such camera, and

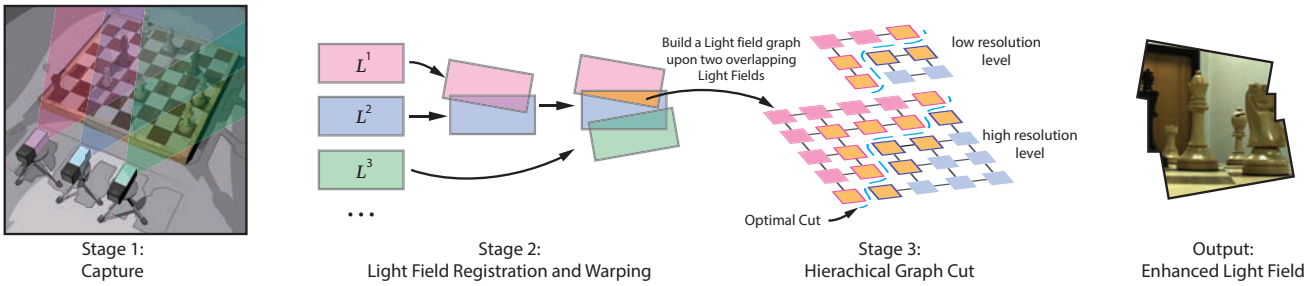


Fig. 2. The pipeline of our proposed LF stitching algorithm. We represent each 4D LF as 2D planes for ease of visualization.

generate a global enhanced LF for rendering effects that are more dramatic than each input LF.

Image Stitching. Our light field stitching idea can be thought of as an extension of 2D image stitching for panorama generation with increased field of view. A good survey on 2D stitching can be found in [22]. In addition to generating panoramas from a rotating camera [8], [23]–[25], variants include strip panorama [26], pushbroom panorama [27], and X-slit panorama [28]. Recently, Pirk et al. [29] proposed a system to deal with challenges of accessing appropriate parts from a gigapixel video.

There are techniques to generate 3D panoramas, mostly in the context of stereoscopic images and videos. In the angular domain, Peleg et al. [30] proposed to generate omnistereo panoramas by mounting the camera on a rotating arm. Shum et al. [31] use a similar rig to create a cylindrical light field, or concentric mosaics. Kim et al. [32] synthesize multi-perspective stereoscopy by cutting through the LF. While simple and effective, the strategy of capturing a light field from a single rotating camera can lead to visible seams and undesirable vertical parallax. Richardt et al. [33] correct for perspective distortion and vertical parallax, then apply an optical-flow-based technique to reduce aliasing. Such image panorama techniques have also been extended to videos [34]–[37]. We adopt their core technique, i.e., high-dimensional graph-cut, to resolve our LF stitching problem.

The approach closest to ours is that of Birklbauer and Bimber [38], [39], since it also acquires and fuses multiple LFs. Their approach targets at registering and transforming multiple rotated two-plane parametrized (2PP) light fields into a global cylindrical coordinate system (composed of two nested cylinders rather than parallel planes). They require the camera moves precisely during light field capturing in order to minimize registration artifacts. In contrast, our approach allows the user to freely rotate or translate the LF camera and aligns two 2PP light field into a common 2PP parameterization through matrix transformation. We then conduct a high-dimensional graph-cut to compensate for misalignment. This parameterization allows us to create LFs with various enhanced features not restricted to panorama. Furthermore, because of our refinement technique, our approach is more tolerant to imperfect registrations.

3 Overview of our Technique

Fig. 2 shows our system for generating enhanced LF from multiple displaced LFs. We show that a 5×6 matrix is sufficient to transform rays from one LF to another; we call this matrix the *ray-space motion matrix (RSMM)*. We assume the LFs are captured in sequence, and we use the first LF as the reference. The RSMM is computed for all adjacent LFs; by chaining RSMMs, all LFs can then be transformed to the reference coordinate system.

At each iteration after the two LFs are aligned (Sec. 4), we refine the stitching process (Sec. 5). The refinement is necessary to account for slight errors in estimating the RSMMs and to handle slight scene motion, since the LFs were sequentially captured. To stitch each pair of LFs, we map their overlapping ray subspace to a 4D graph and rely on hierarchical graph-cut to efficiently find a seamless boundary.

4 Ray-Space Motion Matrix

We use the conventional two-plane parametrization to represent an LF. Each ray r is parameterized by its intersection points with two planes: u, v as one intersection with the sensor plane Π_{uv}, and s, t as the other intersection with the (virtual) camera plane Π_{st}. The two planes are a unit distance apart. Since we acquire multiple LFs using the same LF camera with fixed configuration, the transformation between LFs is simply determined by the change (rotation and translation) of the 2PP.

To align the LFs, we need to align all the rays. One obvious approach would be to compute structure-from-motion (SFM) to estimate LF motion. Unfortunately, the stability of SFM (which is non-linear and local) is heavily dependent on scene composition and camera motion. Instead, to align the rays, we solve for a matrix per pair of LFs without explicit estimation of camera motion. To align all LFs, we start with the first LF and iteratively go through all the LFs to estimate pairwise warping function and subsequently align them w.r.t. the first LF. The overall warping between an arbitrary pair of LFs can be computed by concatenating the in-between matrices.

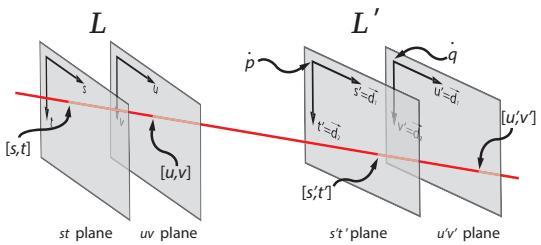


Fig. 3. Our pairwise RSMM maps each ray $[s,t,u,v]$ from L to $[s',t',u',v']$ in L'.

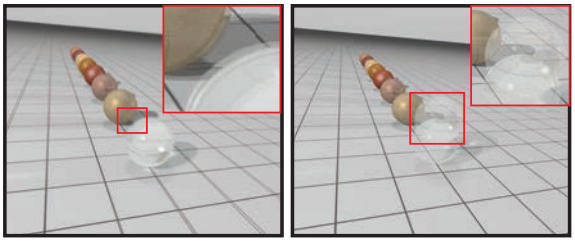


Fig. 4. Synthetic examples show the misalignment due to undersampling (left) and ghosting artifacts due to motion (right).

We now derive the pairwise warping function. We start with the analysis in [40] and extend it to handle LF alignment. Without loss of generality, we set the coordinates of one light field L as global coordinates. The 2PP for the second light field L' are specified by two points $\dot{p}$ and $\dot{q}$ (with one on each plane), and two spanning vectors (u', v' or s', t' directions) $\vec{d}_1$, $\vec{d}_2$ of the planes, as shown in Fig. 3. We further simplify the derivation by using the relative coordinate $\sigma = s - u$ and $\tau = t - v$ to represent the rays. For every ray $r[\sigma,\tau,u,v]$ in L, we compute $r[\sigma',\tau',u',v']$ in L'. This is done by intersecting r with the 2PP of L':

$$\begin{aligned}[u,v,0] + \lambda_1[\sigma,\tau,1] &= \dot{p} + \vec{d}_1 s' + \vec{d}_2 t', \\ [u,v,0] + \lambda_2[\sigma,\tau,1] &= \dot{q} + \vec{d}_1 u' + \vec{d}_2 v'. \end{aligned} \quad (1)$$

From Eqn. (1), we have:

$$\begin{aligned}
\sigma' = s' - u' &= \frac{1}{\gamma}(d_2^x(p_y - q_y) - d_2^y(p_x - q_x) \\
&+ \sigma(d_2^y(p_z - q_z) - d_2^z(p_y - q_y)) \\
&+ \tau(d_2^z(p_x - q_x) - d_2^x(p_z - q_z))) \\
\tau' = t' - v' &= \frac{1}{\gamma}(d_1^y(p_x - q_x) - d_1^x(p_y - q_y) \\
&+ \sigma(d_1^z(p_y - q_y) - d_1^y(p_z - q_z)) \\
&+ \tau(d_1^x(p_z - q_z) - d_1^z(p_x - q_x))) \\
u' &= \frac{1}{\gamma}(d_2^x q_y - d_2^y q_x + d_2^y u - d_2^x v \\
&+ \sigma(d_2^y q_z - d_2^z q_y) + \tau(d_2^z q_x - d_2^x q_z) + d_2^z(\sigma v - \tau u)) \\
v' &= \frac{1}{\gamma}(d_1^y q_x - d_1^x q_y - d_1^y u + d_1^x v \\
&+ \sigma(d_1^z q_y - d_1^y q_z) + \tau(d_1^x q_z - d_1^z q_x) - d_1^z(\sigma v - \tau u)) \quad (2)
\end{aligned}$$

where superscripts indicate the components of a vector and γ is a determinant:

$$\gamma = \begin{vmatrix} d_1^x & d_1^y & d_1^z \\ d_2^x & d_2^y & d_2^z \\ \sigma & \tau & 1 \end{vmatrix}. \quad (3)$$

Notice that $[\sigma',\tau',u',v']$ are bilinear rational functions of σ, τ, u, v. The transformation also has a singularity where γ is zero. This happens when a ray in the original parametrization is parallel to the new parametrization plane, caused by a large rotation between two LFs (say around $90°$). Such large rotations are not done in any of our experiments.

We rewrite Eqn. (2) in a matrix format:

$$\begin{pmatrix} \gamma\sigma' \\ \gamma\tau' \\ \gamma u' \\ \gamma v' \\ \gamma \end{pmatrix} = \begin{pmatrix} m_{00} & m_{01} & 0 & 0 & 0 & m_{05} \\ m_{10} & m_{11} & 0 & 0 & 0 & m_{15} \\ m_{20} & m_{21} & m_{22} & m_{23} & m_{24} & m_{25} \\ m_{30} & m_{31} & m_{32} & m_{33} & m_{34} & m_{35} \\ m_{40} & m_{41} & 0 & 0 & 0 & m_{45} \end{pmatrix} \begin{pmatrix} \sigma \\ \tau \\ u \\ v \\ \lambda \\ 1 \end{pmatrix} \quad (4)$$

where $m_{00} = d_2^y(p_z - q_z) - d_2^z(p_y - q_y)$, $m_{01} = d_2^z(p_x - q_x) - d_2^x(p_z - q_z)$, $m_{05} = d_2^x(p_y - q_y) - d_2^y(p_x - q_x)$, $m_{10} = d_1^z(p_y - q_y) - d_1^y(p_z - q_z)$, $m_{11} = d_1^x(p_z - q_z) - d_1^z(p_x - q_x)$, $m_{15} = d_1^y(p_x - q_x) - d_1^x(p_y - q_y)$, $m_{20} = d_2^y q_z - d_2^z q_y$, $m_{21} = d_2^z q_x - d_2^x q_z$, $m_{22} = d_2^y$, $m_{23} = -d_2^x$, $m_{24} = d_2^z$, $m_{25} = d_2^x q_y - d_2^y q_x$, $m_{30} = d_1^z q_y - d_1^y q_z$, $m_{31} = d_1^x q_z - d_1^z q_x$, $m_{32} = -d_1^y$, $m_{33} = d_1^x$, $m_{34} = -d_1^z$, $m_{35} = d_1^y q_x - d_1^x q_y$, $m_{40} = d_1^y d_2^z - d_2^y d_1^z$, $m_{41} = d_2^x d_1^z - d_1^x d_2^z$, $m_{45} = d_1^x d_2^y - d_2^x d_1^y$, and $\lambda = \sigma v - \tau u$. We call this 5×6 matrix transform the *Ray Space Motion Matrix (RSMM)*. It determines the warping from L' to L and has 21 non-zero entries.

In Eqn. (4), by replacing γ in the first four rows by its value expressed in the fifth row, we obtain 4 linear equations for each pair of corresponding rays. To compute the RSMM between L and L', we need a minimum of 6 pairs of ray correspondences. For robustness, we use a much larger set of correspondences. Specifically, we find potential ray correspondences based on ray features, and then apply RANSAC to remove outliers. While ideally, a SIFT-like feature defined on 4D ray space should be used, in our case, it may not be robust due to the relatively coarse angular sampling. For the Lytro camera (which we use), the sampling is 10×10. Hence we assume small camera motions between two consecutive LF's and directly use 2D SIFT features extracted from sub-aperture views as the ray features. If we group SIFT features according to their sub-aperture views, one group of features only matches to a group of features in the other LF. In other words, one group of features can not match to the features across different groups of the other LF. This implies that there should be no out-of-plane rotation or a translation normal to the 2PP plane. However, our graph-cut refinement step compensates for the errors caused by these small motions. The detailed algorithm is in Algorithm 1.

Algorithm 1 RSMM Estimation

Require: feature ray r'_i from L' and r_i from L ($i \in N$)
Minimum error $Err_{min} = \infty$.
Best warping matrix M_{min} = all zero matrix.
Iteration $it = 0$.
Assign K, T, D and Err_{thresh} empirically.
while $it < K$ **do**
  Initialize the set of corresponding rays C with randomly selected $m \geq 5$ pairs of feature rays from L and L' with color distance smaller than T.
  Estimate warping matrix M via SVD from current set C.
  for every ray r_j in L not selected in C **do**
    Warp r_j onto L' with M.
    if it corresponds with a ray r'_j in L' with averaged difference on RGB channels smaller than T **then**
      Add r'_j, r_j to C.
    end if
  end for
  if the number of pairs in C is larger than D **then**
    Recompute the warping matrix M via SVD from C.
    Measure the current error E by warping each ray r_i onto r'_i and measure the color distance.
    if $E < Err_{min}$ **then**
      $Err_{min} = E$.
      $M_{min} = M$.
    end if
    if $Err_{min} < Err_{thresh}$ **then**
      break
    end if
  end if
end while

In our experiments, to guarantee the best estimation, we set K to be infinity and the algorithm runs until the error is less than a sufficiently small value Err_{thresh}. Since raw Lytro images are usually rather noisy, we set T to a relatively high value of $\frac{20}{255}$. We choose D as 30% of the total number of the rays associated with the SIFT features. On average, it took slightly under 1.5 minutes to extract the RSMMs for two LFs with a resolution of $10 \times 10 \times 800 \times 800$ on a 3.2 GHz CPU.

Note that when the motion between two LFs is pure translation or z-plane rotation ($d_1^z = 0, d_2^z = 0$), the RSMM reduces to a 5×5 matrix where the original column 5 is removed. For robustness, we always compute 5×6 RSMM in our experiments.

Given image noise as well as quantization in space and angle, it is likely the computed RSMMs are not perfect. Fig. 4 (a) shows a view of the warping result of two sparsely sampled LF with 5×5 views each. The two light fields have large overlapping subspace but the rays do not match exactly between the two LFs due to undersampling. Therefore our estimated RSMM is not perfect. In addition, since we are capturing LFs sequentially, the scene may have changed a little over time (e.g., in the case of capturing a human scene). As shown in Fig. 4, even the perfect RSMM still could not represent the shifting on temporal domain. All these factors will cause noticeable ghosting artifacts if we choose to blend the rays from both LFs. To reduce these artifacts, we add a stitching refinement step by choosing from one of the LFs.

5 Seamless Stitching

By applying the RSMM transformation, two LFs are registered under a same 2PP coordinate. The stitching of two LFs is then performed by searching for an optimal 4D cut in the overlapped region that minimizes the inconsistency along the cutting boundary. This problem can be solved precisely by graph-cut. As shown in Fig 2, we warp L towards L' using the estimated RSMM and denote the overlapped LF space as $\hat{L}$. Within $\hat{L}$, we assign each ray r a binary label l_r, specifying if it belongs to L or L'. To do so, we construct a 4D graph on the new LF $\bar{L}$ and map the inconsistencies along spatial and angular dimensions as edges. We formulate the problem of finding the cut as to minimize the following energy:

$$E = \sum_{\bar{r} \in \bar{L}} E(l_{\bar{r}}) + \sum_{\bar{r}, \tilde{r} \in \mathcal{N}} E(l_{\bar{r}}, l_{\tilde{r}}), \tag{5}$$

where $\bar{r}$ denotes a ray in $\bar{L}$ and $\mathcal{N}$ defines the 8 direct neighbors from 4 dimensions. $l_{\bar{r}} = 0$ for $\bar{r}$ is assigned to L and $l_{\bar{r}} = 1$ for $\bar{r}$ to L'. $E(l_{\bar{r}})$ denotes the cost of assigning $\bar{r}$ with label $l_{\bar{r}}$ and forces rays lying outside the overlapped regions to keep their original label:

$$E(l_{\bar{r}}) = \begin{cases} \infty & , \quad \bar{r} \notin \hat{L} \wedge \\ & \quad ((\bar{r} \in L \wedge l_{\bar{r}} = 1) \vee (\bar{r} \in L' \wedge l_{\bar{r}} = 0)) \\ 0 & , \quad \text{otherwise} \end{cases} \tag{6}$$

$E(l_{\bar{r}}, l_{\tilde{r}})$ denotes the cost of assigning labels $l_{\bar{r}}$ and $l_{\tilde{r}}$ to $\bar{r}$ and its adjacent ray $\tilde{r}$. The key observation here is that to reliably stitch two LFs, we need to measure the differences of adjacent rays in both spatial and angular dimensions. We adopt the following energy function similar to [26], [34], [41] as:

$$E(l_{\bar{r}}, l_{\tilde{r}}) = \frac{|I_{\bar{r}} - I'_{\bar{r}}| + |I_{\tilde{r}} - I'_{\tilde{r}}|}{|\mathrm{G}_u(\bar{r})| + |\mathrm{G}_v(\bar{r})|}, \tag{7}$$

where $|\cdot|$ denotes the norm (e.g., L_1 or L_2) and $(I_r - I'_r)$ is the radiance difference between the two LFs for ray r. In the denominator, $\mathrm{G}_x(\bar{r})$ measures the gradient of $\bar{r}$ on dimension x. This gradient prior [34] leads the cut through high frequency regions where the inconsistency will be less noticeable.

Notice that it is necessary to conduct high-dimensional (4D) graph-cut which incorporates the consistency requirement along angular dimension. Individually stitching corresponding 2D images in the LFs will not produce comparable results. Without angular coherence,

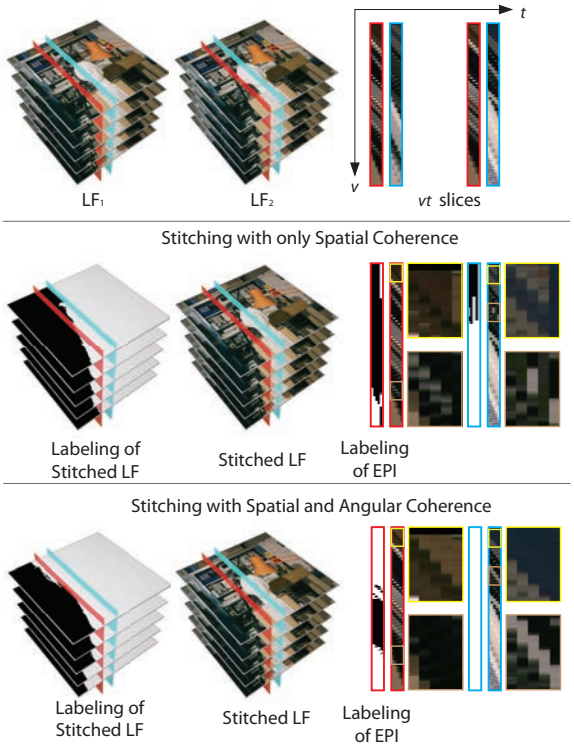


Fig. 5. The importance of enforcing angular coherence. Top row: Two 3D LFs from the Tsukuba dataset. The red and blue slices are the EPIs (v, t slices) of each LF. Second row: The labeling of the stitched LF, the stitched LF by enforcing only spatial coherence, labeling of EPIs, and stitched EPIs. Third row: The labeling of the stitched LF, the stitched LF by enforcing both spatial and angular coherence, labeling of EPIs, and stitched EPIs.

two angularly adjacent stitched images may appear significantly different. Fig. 5 shows an example on stitching two 3D LFs (in this case, Π_{st} is a 1D line) from the Tsukuba dataset. Compared with our result, stitching with only spatial coherence introduces discontinuities along angular dimension (see stitched EPIs).

Since our graph construction requires adding edges in both angular and spatial dimensions, the resulting 4D graph using the raw resolution can be ultra-large. For example, a graph constructed by two LFs at a resolution of $10 \times 10 \times 800 \times 800$ with 50 percent overlap can result in 32 million nodes and 0.25 billion edges. Applying the max-flow/min-cut algorithm on such a graph would be computationally prohibitive. For efficiency reasons, we opt for a coarse-to-fine version of graph-cut [35]. Our strategy is to first construct a graph at a low resolution and compute the cut. We then go one level finer and use the interpolated cut to eliminate unnecessary nodes in the higher resolution graph and find a new cut. Specifically, we build a Gaussian pyramid of the LFs in the spatial domain (i.e., we only down sample uv but not st as the angular sampling is generally sparse). Starting from the top level, we find the optimal 4D boundary and map it to the initial boundary at the next level. All nodes within a radius of R ($R = 20$ in our experiments) are marked as new active nodes. Next we refine the boundary with a new cut and go one level further.

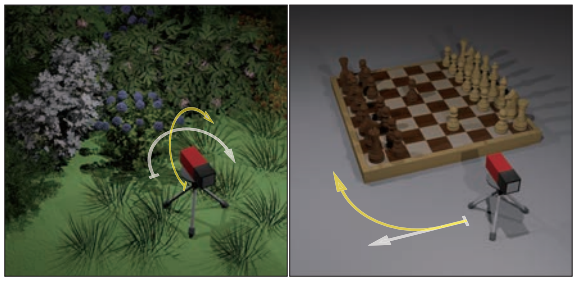


Fig. 6. Acquisition setups. (a) To increase the FoV, we pan and tilt the LF camera. (b) To enhance synthetic aperture and parallax, we translate the LF camera (white arrow); To enhance rotational parallax, we rotate the camera around the object.

6 Experimental Results

In this section, we show results of enhancing various light field rendering effects: extended horizontal field-of-view, larger synthetic aperture, and larger spatial/angular parallax. All experiments are conducted on a PC with Intel i7-3930 CPU and 64GB memory. We use the Lytro LF camera to capture LFs at a spatial resolution of 328×378 and an approximate angular resolution of 10×10. Since ray mapping is focal length dependent for the Lytro camera, we use a fixed focal length throughout the capture of a scene.

We use the toolkit by Dansereau et al. [42] to extract the raw LF data, calibrate each microlens, and perform vignetting correction and demosaicing. Since we know the intrinsics of the LF camera for each image (from the image profile), we can compute the ray equation for each pixel. To stitch the Lytro LFs, we find that using a three-level pyramid is sufficient. The average time taken to stitch a pair of LFs is about 20 minutes. For N LFs, the time is about $20 \times (N - 1)$ minutes.

6.1 Extending the Horizontal and Vertical FoV

LFs captured typically have narrow FoVs. For example, the Stanford LF array produces a horizontal FoV of 57° and the Lytro camera has an even narrower FoV of about 40°. Our first task is to apply the stitching technique to significantly increase the horizontal FoV. This is analogous to producing a panorama from a sequence of narrow FoV images. We therefore also call this application panoramic LFs. To capture panoramic LFs, we mount the Lytro camera on a tripod and rotate it horizontally to capture and then stitch consecutive LFs (Fig. 6 (a)).

Fig. 7. A panoramic LF generated from a 5×4 grid of LFs. Left: full view focusing at a plane close to the foreground flower. Right 2×2 images: close-up views of highlighted regions focused at the background (top row) and foreground (bottom row).

The garden scene shown in Fig. 1 (a) is challenging as it contains wide depth variations and complex occlusions. To capture an enhanced FoV of the scene, we shoot 7 horizontal LFs to cover about 150° FoV. Note that the fountain on the right is visible in one LF but not the others. Fig. 7 shows an example of stitching a 2D grid of LFs, where the FoV is extended in both horizontal and vertical directions. The capture is done by panning and tilting the Lytro camera to acquire a 5×4 array of LFs. For this dataset, we ensured that the fountain is visible in multiple LFs. The appearance of the water is inconsistent across the LFs due to its motion, but our graph-cut technique is still able to stitch them to produce a visually acceptable panoramic LF.

Given the Lytro's small FoV, using it to capture a people scene significantly constrains how people can be positioned and still appear in the image. With our LF stitching technique, we can allow the locations of people to be more spread out by acquiring and then stitching multiple LFs. Fig. 9 shows an example of 4 people standing apart at different depths. Notice that for such a scene, it would be hard for people to keep completely stationary between the shots. Our graph-cut technique is able to compensate for small motions and the resulting panoramic LFs exhibit minimal ghosting artifacts. The green curves show the cuts and the bottom row shows an autostereoscopic rendering of the stitched LF which can be directly used as input to autostereoscopic 3D displays. (Here we assume the display consists of vertical cylindrical lens array.)

Note that the panoramic LF can be viewed as a form of concentric mosaics [31]. The main difference is that each view captured is actually an LF consisting of 2D grid of views, while the inputs to the concentric mosaic are views from a rotating single camera.

6.2 Extending Rotational Parallax

To enhance the rotational parallax in LF rendering, we place the target object on a rotation table with a constant color background. The Lytro camera is used to capture the object at different rotated views. With a constant background, the acquisition process emulates rotating the camera around the object (Fig. 6 (b)).

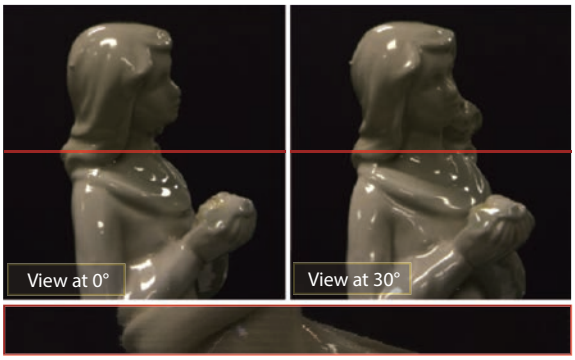


Fig. 8. A stitched LF with increased rotational parallax. The top row shows two views from the stitched LF. The horizontal strip at the bottom is a composite of the profile along the red line as the object is virtually rotated. Notice the coherency in the strip.

For the example shown in Fig. 8, we acquired 60 LFs to cover around 30° around the object. In this case, each LF is a piecewise linear approximation of the rays captured along a circle around the object. With the stitched LF, we can then synthesize any view point around the object within a range of about 30° (while only a small motion is permitted for a single LF).

Extending rotational parallax is particularly useful for visualizing view-dependent features such as transparency. In Fig. 1 (d), we captured 60 LFs to cover about 30° around a transparent glass containing specular and translucent dices. Since the scene is highly non-Lambertian, 3D reconstruction based methods are not expected to be reliable. In contrast, our RSMM based on ray-ray correspondences is (in principle) insensitive to view-dependency effects. In practice, however, the

Fig. 9. A panoramic LF of a people scene. Top row: reference (central) views of 4 LFs and the cuts (projected from 4D to 2D). Bottom row: an autostereoscopic rendering of the stitched LF.

acquired LFs are generally undersampled in the angular dimension. As a result, slight changes in ray directions can still lead to large appearance changes, e.g., on specular highlights. The issue is effectively addressed using RANSAC that helps pick more Lambertian features of the scene and remove most view-dependent outliers. The resulting RSMM is usually quite accurate, producing stitched LFs with satisfactory rotational parallax effects as shown in Fig. 1 (d) and in the supplementary video.

6.3 Increasing Synthetic Aperture and Parallax

A common problem in LF acquisition is the choice of baseline between view cameras. To reduce angular aliasing, Lytro uses a small baseline but the resulting virtual aperture is also small and synthetic defocus blurs are less obvious. In contrast, a large baseline (as used in an LF array) can synthesize more significant defocus blurs; however, it also introduces severe aliasing due to undersampling. Our LF stitching provides a natural solution to resolve this problem by covering more space and therefore increase the synthetic aperture.

In our setup, we mount the Lytro camera on a manual translation stage and move it horizontally at an interval of roughly 0.5 mm, as shown in Fig. 6 (b). Fig. 1 (b) shows a chess set scene captured by 20 LFs using this setup. The lower triangle of (b) shows the refocusing result with a single LF. Due to the small virtual aperture, the Bokeh effect appears rather small and the parallax of the chess pieces is barely noticeable. In contrast, applying refocusing on the stitched LF produces much larger (horizontal) Bokeh and parallax effects, as shown in the upper triangle of (b) and supplementary video. In fact, using the stitched LF, we can dynamically control the magnitude of refocusing by integrating over different angular spans of rays, as shown in (c).

As with the case of increased translational parallax, the extended virtual aperture also helps to better visualize view-dependent effects (Fig. 10). Here, 20 LFs were also captured. More results can be found in the supplementary video. Notice that the our current setup only increases the virtual aperture in the horizontal direction and can be easily extended to both horizontal and vertical direction.

6.4 Comparisons to Other Approaches

Recently, Birklbauer et al. [39] have presented a LF stitching technique that requires the LF camera to rotate precisely around an axis parallel to the sensor. The rotation axis should also intersect with the camera's optical axis and the sequence should be acquired with an approximately identical rotation angle. In contrast, our technique supports much more flexible camera movement. Specifically, we can handle rotation sequences that do not share a common rotation axis, i.e., the sequence can exhibit combinations of rotational and translational motions. This is because we set out to automatically obtain the general LF registration matrix RSMM that can simultaneously handle rotation and translation.

For comparison, we apply the source code [39] provided by the authors to stitch our garden LF data.

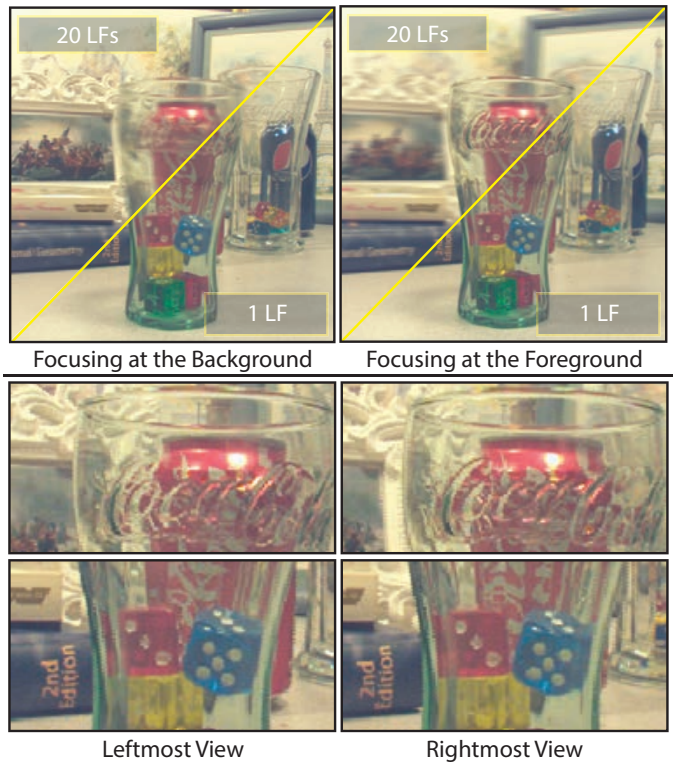


Fig. 10. A stitched LF with increased translational parallax and Bokeh. The top row shows shallow DoF rendering at the background (left) and the foreground (right). In both cases, the upper triangle uses the stitched LF and the lower uses a single LF. The Bokeh is significantly increased using the stitched version. The middle and bottom rows show close-up views of synthetically changing the view points around the glass.

Figure 11 compares the refocused rendering results on the stitched LFs produced by our method vs. [39]. Recall that this LF dataset was captured by manually rotating the LF camera where the rotation axis exhibits slight translations across the LFs. Such translational motions, even though very small, violate the assumptions in [39]. As a result, the refocused results exhibit strong ghosting (aliasing) artifacts due to misalignment. Our solution accounts for both rotation and translation and it further corrects potential registration errors via a high-dimensional graph-cut. Therefore, our refocused rendering results are nearly aliasing free and preserve sharp details. More importantly, our approach frees the user from precise LF acquisition.

6.5 Failure Cases

The main limitation of our technique is that the adjacent LFs have large overlaps, to ensure reliable RSMM estimations. If the estimated RSMM contains small errors, graph-cut can still effectively eliminate inconsistency and produce visually pleasing results. In this case, however, the stitched result is not a "real" LF: corresponding rays are not guaranteed to intersect at common 3D points. Such artifacts are best illustrated in synthetical aperture rendering. Top row of Fig. 12 uses the rotational parallax setup to capture 20 LFs of a human eye. Due to errors in RSMM, the shallow DoF rendering result exhibits blurs when focusing at the pupil even though each view in the stitched LF is sharp. If the motion between adjacent LFs are too large, there will not be sufficient corresponding rays for computing the RSMM. Fig. 12 shows a typical example. Although graph-cut can partially eliminate inconsistencies, the resulting stitched LF exhibits large errors, e.g., part of the highlighted flower is missing in the final stitched result.

7 Concluding Remarks

We have presented a simple but effective LF stitching technique to enhance a number of LF tasks. We derived a linear method to align the LFs using what we call the *ray-space motion matrix (RSMM)*. The use of RSMM obviates the need for explicit camera motion and scene depth estimation. To account for imprecise alignment and possible scene changes across the input LFs, we refine the output LF by stitching using multi-resolution, high-dimensional graph-cut. Using the commodity Lytro camera, we showed how we can enhance LFs through extended horizontal and/or vertical FoV, larger synthetic aperture and defocus blur, and larger parallax.

As discussed in Section 6.5, our approach requires significant overlap between adjacent LFs. To handle larger motions, we plan to explore the joint use of RSMM and SfM to compute LF camera motion. Here, the RSMMs may be estimated to initialize SfM so as to reduce the possibility of bad local minima. Although our RSMM estimation can, in principle, be applied to any LF sampling (regular or irregular), in practice, dense sampling of each LF is required for reliable results.

Another interesting possibility is to reconstruct the ray space without warping and stitching by making use of the recent simplex-based ray space approach [17] (which generates a triangulated ray space from ray samples). Our RSMM could potentially be used as the ray-ray correspondence constraints. However, currently the 4D Constrained Delaunay Triangulation is still an open problem in computational geometry and the 3D approximation might fail to interpolate complex ray geometry.

Like any local registration technique, our current pairwise LF registration method is subject to drift. While this is less of an issue for the relatively short LF sequences in our work, we plan to investigate global alignment approaches for much longer sequences. In addition, by using the principles of plenoptic sampling [43], it may be possible to plan a minimal capture configuration while minimizing aliasing effects due to undersampling.

8 Acknowledgement

This project was partially supported by the National Science Foundation under grants IIS-CAREER-0845268 and

Fig. 11. Comparisons of refocused rendering using our method vs. [39] on the garden LF data. Our method is able to stitch the input into a ghosting free panoramic light field while [39] produces strong ghosting artifacts due to translational motions across the input LFs.

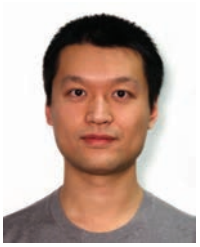
Fig. 12. Failure cases. Top row: the depth-of-field rendering on the in-focus region (the pupil) exhibits blurs due to errors in RSMM estimation. Bottom row: the flower is incorrectly stitched due to large displacement between the two LFs.

IIS-1218156. We'd like to thank Clemens Birklbauer and Oliver Bimber for providing the source code of [39] and useful suggestions. We'd also like to thank anonymous TVCG reviewers for their insightful comments.

REFERENCES

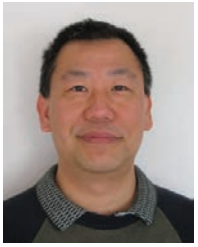
[1] B. Wilburn, N. Joshi, V. Vaish, M. Levoy, and M. Horowitz, "High-speed videography using a dense camera array," in *CVPR*, 2004.
[2] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen, "The lumigraph," in *SIGGRAPH*, 1996.
[3] E. H. Adelson and J. Y. A. Wang, "Single lens stereo with a plenoptic camera," *TPAMI*, 1992.
[4] T. Georgiev, K. C. Zheng, B. Curless, D. Salesin, S. Nayar, and C. Intwala, "Spatio-angular resolution tradeoff in integral photography," in *EGSR*, 2006.
[5] T. E. Bishop, S. Zanetti, and P. Favaro, "Light field superresolution," in *ICCP*, 2009.
[6] T. Georgiev, G. Chunev, and A. Lumsdaine, "Superresolution with the focused plenoptic camera," in *Proc. SPIE 7873*, 2011.
[7] S. Wanner and B. Goldluecke, "Spatial and angular variational super-resolution of 4D light fields," in *ECCV*, 2012.
[8] Y. Xiong and K. Pulli, "Fast image stitching and editing for panorama painting on mobile phones," in *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2010.
[9] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *SIGGRAPH*, 2001.
[10] M. Levoy and P. Hanrahan, "Light field rendering," in *SIGGRAPH*, 1996.
[11] R. Ng, M. Levoy, M. Brdif, G. Duval, M. Horowitz, and P. Hanrahan, "Light field photography with a hand-held plenoptic camera," *Stanford University Computer Science Tech Report*, 2005.
[12] M. Zwicker, W. Matusik, F. Durand, H. Pfister, and C. Forlines, "Antialiasing for automultiscopic 3D displays," in *SIGGRAPH*, 2006.
[13] A. Davis, M. Levoy, and F. Durand, "Unstructured light fields," *Comp. Graph. Forum*, 2012.
[14] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen, "Unstructured lumigraph rendering," in *SIGGRAPH*, 2001.
[15] B. G. S. Wanner, "Globally consistent depth labeling of 4D light fields," in *CVPR*, 2012.
[16] T. Basha, S. Avidan, A. Hornung, and W. Matusik, "Structure and motion from scene registration," in *CVPR*, 2012.
[17] Z. Yu, X. Guo, H. Ling, A. Lumsdaine, and J. Yu, "Line assisted light field triangulation and stereo matching," in *ICCV*, 2013.
[18] C. Kim, H. Zimmer, Y. Pritch, A. Sorkine-Hornung, and M. Gross,

"Scene reconstruction from high spatio-angular resolution light fields," *ACM Trans. Graph.*, 2013.
[19] D. Aliaga and I. Carlbom, "Plenoptic stitching: A scalable method for reconstructing 3D interactive walkthroughs," in *SIGGRAPH*, 2001.
[20] S. B. Kang, M. Wu, Y. Li, and H.-Y. Shum, "Large environment rendering using plenoptic primitives," *IEEE Trans. On Circuits and Systems for Video Technology*, 2003.
[21] J. Lehtinen, T. Aila, J. Chen, S. Laine, and F. Durand, "Temporal light field reconstruction for rendering distribution effects," *ACM Trans. Graph.*, 2011.
[22] R. Szeliski, "Image alignment and stitching: A tutorial," *Found. Trends. Comput. Graph. Vis.*, 2006.
[23] R. Szeliski and H.-Y. Shum, "Creating full view panoramic image mosaics and environment maps," in *SIGGRAPH*, 1997.
[24] K. He, H. Chang, and J. Sun, "Rectangling panoramic images via warping," *ACM Trans. Graph.*, 2013.
[25] B. Summa, J. Tierny, and V. Pascucci, "Panorama weaving: fast and flexible seam processing," *ACM Trans. Graph.*, 2012.
[26] A. Agarwala, M. Agrawala, M. Cohen, D. Salesin, and R. Szeliski, "Photographing long scenes with multi-viewpoint panoramas," in *SIGGRAPH*, 2006.
[27] S. M. Seitz and J. Kim, "The space of all stereo images," *IJCV*, 2002.
[28] A. Zomet, D. Feldman, S. Peleg, and D. Weinshall, "Mosaicing new views: the crossed-slits projection," *TPAMI*, 2003.
[29] S. Pirk, M. F. Cohen, O. Deussen, M. Uyttendaele, and J. Kopf, "Video enhanced gigapixel panoramas," in *SIGGRAPH Asia Technical Briefs*, 2012.
[30] S. Peleg and M. Ben-Ezra, "Stereo panorama with a single camera," in *CVPR*, 1999.
[31] H.-Y. Shum and L.-W. He, "Rendering with concentric mosaics," in *SIGGRAPH*, 1999.
[32] C. Kim, A. Hornung, S. Heinzle, W. Matusik, and M. Gross, "Multi-perspective stereoscopy from light fields," *ACM Trans. Graph.*, 2011.
[33] C. Richardt, Y. Pritch, H. Zimmer, and A. Sorkine-Hornung, "Megastereo: Constructing high-resolution stereo panoramas," in *CVPR*, 2013.
[34] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," in *SIGGRAPH*, 2003.
[35] A. Agarwala, K. C. Zheng, C. Pal, M. Agrawala, M. Cohen, B. Curless, D. Salesin, and R. Szeliski, "Panoramic video textures," in *SIGGRAPH*, 2005.
[36] A. Rav-Acha, Y. Pritch, D. Lischinski, and S. Peleg, "Dynamosaics: video mosaics with non-chronological time," in *CVPR*, 2005.
[37] V. Couture, M. S. Langer, and S. Roy, "Panoramic stereo video textures," in *ICCV*, 2011.
[38] C. Birklbauer, S. Opelt, and O. Bimber, "Rendering gigaray light fields," in *Eurographics*, 2013.
[39] C. Birklbauer and O. Bimber, "Panorama light-field imaging," in *Eurographics*, 2014.
[40] J. Yu and L. McMillan, "Modelling reflections via multiperspective imaging," in *CVPR*, 2005.
[41] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen, "Interactive digital photomontage," in *SIGGRAPH*, 2004.
[42] D. G. Dansereau, O. Pizarro, and S. B. Williams, "Decoding, calibration and rectification for lenselet-based plenoptic cameras," in *CVPR*, 2013.
[43] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, "Plenoptic sampling," in *SIGGRAPH*, 2000.



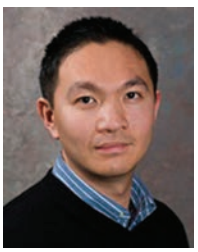
Xinqing Guo is now a Ph.D. student at the Department of Computer and Information Sciences, University of Delaware. He received his M.S. degree in Electrical Engineering from University of Delaware in 2011, and B.S. degree in Electrical Engineering from Southeast University, China, in 2008. His research interests include computational photography, computer graphics and computer vision.

Zhan Yu is a Research Scientist at Adobe Systems Inc. since 12/2013. Before that, he received his Ph.D. of Computer Science from the University of Delaware in 2013 and B.S. of Software Engineering from Xiamen University in 2008. His research interests are computer graphics, computer vision, and computational photography.



Sing Bing Kang is a principal researcher at Microsoft Research. He received his Ph.D.in robotics from Carnegie Mellon University, Pittsburgh in 1994. His areas of interest are computer vision and computer graphics, more specically image-based modeling as well as image and video enhancement. Sing Bing has coedited two books ("Panoramic Vision" and "Emerging Topics in Computer Vision") and coauthored two books ("Image-Based Rendering" and "Image-Based Modeling of Plants and Trees"). He has served as area chair and member of technical committee for the major computer vision conferences (ICCV, CVPR, ECCV). In addition, he has served as papers committee member for SIGGRAPH and SIGGRAPH Asia. Sing Bing was program chair for ACCV 2007 and CVPR 2009, and is currently Associate Editor-in-Chief for IEEE Transactions on Pattern Recognition and Machine Intelligence. He is an IEEE Fellow.

Haiting Lin obtained his PhD in Computer Science from the National University of Singapore in 2013 and his B. E. degree from the Renmin University of China in 2008. His research interests include image processing, computer vision. He is a member of the IEEE.



Jingyi Yu is a professor at Computer and Information Science Department at the University of Delaware. He received his B.S. from Caltech in 2000 and M.S. and Ph.D. degree in EECS from MIT in 2005. His research interests span a range of topics in computer graphics, computer vision, and image processing, including computational photography, medical imaging, nonconventional optics and camera design, tracking and surveillance, and graphics hardware. He is an associate editor for IEEE Transactions on Pattern Recognition and Machine Intelligence, Springer The Visual Computer Journal and Springer Machine Vision and Application. He is a member of the IEEE.