Light Field Reconstruction Using Sparsity in the Continuous Fourier Domain

LIXIN SHI, HAITHAM HASSANIEH, ABE DAVIS, DINA KATABI, and FREDO DURAND
Massachusetts Institute of Technology

Sparsity in the Fourier domain is an important property that enables the dense reconstruction of signals, such as 4D light fields, from a small set of samples. The sparsity of natural spectra is often derived from continuous arguments, but reconstruction algorithms typically work in the discrete Fourier domain. These algorithms usually assume that sparsity derived from continuous principles will hold under discrete sampling. This article makes the critical observation that sparsity is much greater in the *continuous* Fourier spectrum than in the *discrete* spectrum. This difference is caused by a windowing effect. When we sample a signal over a finite window, we convolve its spectrum by an infinite sinc, which destroys much of the sparsity that was in the continuous domain. Based on this observation, we propose an approach to reconstruction that optimizes for sparsity in the continuous Fourier spectrum. We describe the theory behind our approach and discuss how it can be used to reduce sampling requirements and improve reconstruction quality. Finally, we demonstrate the power of our approach by showing how it can be applied to the task of recovering non-Lambertian light fields from a small number of 1D viewpoint trajectories.

Categories and Subject Descriptors: I.4.1 [**Image Processing and Computer Vision**]: Digitization and Image Capture—*Imaging geometry, Sampling*; I.3.3 [**Computer Graphics**]: Picture/Image Generation—*Digitizing and Scanning, Viewing Algorithms*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Light fields, computational photography, Fourier transform, sparse FFT, continuous spectrum

ACM Reference Format:

Lixin Shi, Haitham Hassanieh, Abe Davis, Dina Katabi, and Fredo Durand. 2014. Light field reconstruction using sparsity in the continuous Fourier domain. ACM Trans. Graph. 34, 1, Article 12 (November 2014), 13 pages. DOI: http://dx.doi.org/10.1145/2682631

The work is funded by NSF and CGV (no. 1115242).

Authors' addresses: L. Shi, H. Hassanieh, A. Davis (corresponding author), D. Katabi, and F. Durand, Computer Science and Artificial Intelligence Lab, Electrical Engineering and Computer Science Department, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139-4307; email: abedavis@mit.edu.

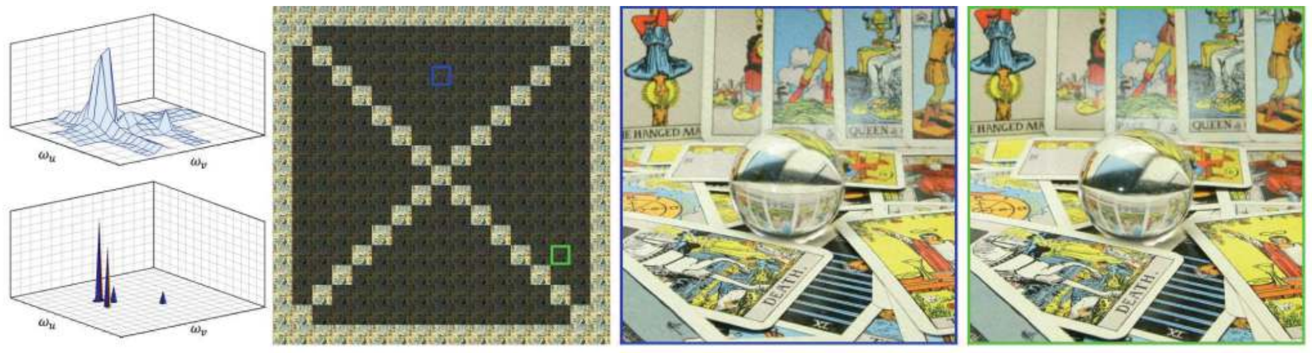
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
© 2014 ACM 0730-0301/2014/11-ART12 $15.00
DOI: http://dx.doi.org/10.1145/2682631

1. INTRODUCTION

Fourier analysis is a critical tool in rendering and computational photography. It tells us how to choose sampling rates (e.g., Heckbert [1989], Mitchell [1991], Durand et al. [2005], and Egan et al. [2009]), predict upper bounds on sharpness (e.g., Ng [2005], Levin et al. [2008b, 2009]), do fast calculations (e.g., Soler and Sillion [1998], model wave optics) (e.g., Goodman [1996], Zhang and Levoy [2009]), perform light field multiplexing [Veeraraghavan et al. 2007], and do compressive sensing (e.g., Candes et al. [2006a]. In particular, the sparsity of natural spectra such as those of light fields makes it possible to reconstruct them from smaller sets of samples (e.g., Levin and Durand [2010] and Veeraraghavan et al. [2007]). This sparsity derives naturally from the continuous Fourier transform, where continuous-valued depth in a scene translates to 2D subspaces in the Fourier domain. However, practical algorithms for reconstruction usually operate on the Discrete Fourier Transform (DFT). Unfortunately, little attention is usually paid to the impact of going from the continuous Fourier domain to the discrete one, and it is often assumed that the sparsity derived from continuous principles holds for discrete sampling and computation. In this article, we make the critical observation that much of the sparsity in continuous spectra is lost in the discrete domain and that this loss of sparsity can severely limit reconstruction quality. We propose a new approach to reconstruction that recovers a discrete signal by optimizing for sparsity in the continuous domain. We first describe our approach in general terms, then demonstrate its application in the context of 4D light field acquisition and reconstruction, where we show that it enables high-quality reconstruction of dense light fields from fewer samples without requiring extra priors or assumptions such as Lambertian scenes.

The difference between continuous and discrete sparsity is due to the windowing effect. Sampling a signal, such as a light field, inside some finite window is analogous to multiplying this signal by a box function. In the frequency domain, this multiplication becomes convolution by an infinite sinc. If the non-zero frequencies of the spectrum are not perfectly aligned with the resulting discretization of the frequency domain (and therefore the zero crossings of the sinc), this convolution destroys much of the sparsity that existed in the continuous domain. This effect is shown in Figure 1(a) which plots a 2D angular slice of the 4D light field spectrum of the Stanford crystal ball. In practice, natural spectra, including those of light fields, are never so conveniently aligned, and this loss of sparsity is always observed.

We introduce an approach to recover the sparsity of the original continuous spectrum based on nonlinear gradient descent. Starting with some initial approximation of the spectrum, we optimize for sparsity in the continuous frequency domain through careful modeling of the projection of continuous sparse spectra into the discrete domain. The output of this process is an approximation of the continuous spectrum. In the case of a light field, this approximation can be used to reconstruct high-quality views that were never captured



(a) sparsity in the discrete vs. continuous spectrum (b) sampled viewpoint images (c) reconstruction of unsampled viewpoint image: blue in (b) (d) reconstruction of unsampled viewpoint image: green in (b)

Fig. 1. Sparsity in the discrete vs. continuous Fourier domain, and our reconstruction results: (a) The discrete Fourier transform (top) of a particular 2D angular slice ω_u, ω_v of the crystal ball's light field, and its reconstructed continuous version (bottom); (b) a grid showing the original images from the Stanford light field archive. The images used by our algorithm are highlighted (courtesy of Stanford [2008]); (c) and (d) two examples of reconstructed viewpoints showing successful reconstruction of this highly non-Lambertian scene that exhibits caustics, specularities, and nonlinear parallax. The uv locations of (c) and (d) are shown as blue and green boxes in (b).

and even extrapolate to new images outside the aperture of recorded samples.

Our approach effectively reduces the sampling requirements of 4D light fields by recovering the sparsity of the original continuous spectrum. We show that it enables the reconstruction of full 4D light fields from only a 1D trajectory of viewpoints, which could greatly simplify light field capture. We demonstrate a prototype of our algorithm on multiple datasets to show that it is able to accurately reconstruct even highly non-Lambertian scenes. Figures 1(b), 1(c), and 1(d) show our reconstruction of a highly non-Lambertian scene and the 1D trajectory of viewpoints used by our implementation.

We believe our observations on continuous versus discrete sparsity and careful handling of sampling effects when going from a sparse continuous Fourier transform into the discrete Fourier domain can also have important applications for computational photography beyond light field reconstruction.

2. RELATED WORK

2.1 Light Field Capture and Priors

Light field capture is challenging because of the 4D nature of light fields and the high sampling rate they require. Capture can be done with a microlens array at the cost of spatial resolution (e.g., Adelson and Wang [1992], Ng et al. [2005], and Georgeiv et al. [2006]), using robotic gantries [Levoy and Hanrahan 1996], using camera arrays [Wilburn et al. 2005], or with a handheld camera moved over time around the scene [Gortler et al. 1996; Buehler et al. 2001; Davis et al. 2012]. All these solutions require extra hardware or time, which has motivated the development of techniques that can reconstruct dense light fields from fewer samples.

Levin and Durand [2010] and Levin et al. [2008a] argue that the fundamental differences between reconstruction strategies can be seen as a difference in prior assumptions made about the light field. Such priors usually assume a particular structure of sparsity in the frequency domain.

Perhaps the most common prior on light fields assumes that a captured scene is made up of Lambertian objects at known depths. Conditioning on depth, the energy corresponding to a Lambertian surface is restricted to a plane in the frequency domain. Intuitively, this means that, given a single image and its corresponding depth map, we could reconstruct all 4 dimensions of the light field (as done in many image-based rendering techniques). The problems with this approach are that the Lambertian assumption does not always hold and that depth estimation usually involves fragile nonlinear inference that depends on angular information, meaning that sampling requirements are not reduced to 2D in practice. However, paired with a coded aperture [Levin et al. 2007; Veeraraghavan et al. 2007] or plenoptic camera [Bishop et al. 2009], this prior can be used to recover superresolution for Lambertian scenes in the spatial or angular domain.

Levin and Durand [2010] use a Lambertian prior, but do not assume that depth is known. This corresponds to a prior that puts energy in a 3D subspace of the light field spectrum, and reduces reconstruction to a linear inference problem. As a result, they require only 3 dimensions of sampling, typically in the form of a focal stack. Like our example application, their technique can also reconstruct a light field from a 1D set of viewpoints. However, they still rely on the Lambertian assumption and the views they reconstruct are limited to the aperture of input views. In contrast, we show how our approach can be used to synthesize higher-quality views, both inside and outside the convex hull of input images, without making the Lambertian assumption. For a comparison, see Section 6.

The work of Marwah et al. [2013] assumes a different kind of structure to the sparsity of light fields. This structure is learned from training data. Specifically, they use sparse coding techniques to learn a dictionary of basis vectors for representing light fields. The dictionary is chosen so that training light fields may be represented as sparse vectors, and their underlying assumption is that new light fields will have similar structure to those in their training data.

2.2 Sparse Fourier Transform

We build on recent work on the sparse Fourier transform, which shows that it is possible to compute the Fourier representation of a sparse signal using only a subset of its samples [Gilbert et al. 2005; Hassanieh et al. 2012a, 2012b; Ghazi et al. 2013]. Since the light field is sparse in the angular domain, it should be possible to leverage this sparsity to recover the signal from a subset of the viewpoints without sampling the whole 2D angular space. The existing sparse Fourier algorithms, however, assume the signal is very sparse in the discrete Fourier domain, that is, the nonzero angular frequencies should be less than 2% to 3% (see Figure 3 in Hassanieh et al. [2012b]). Due to the windowing effect, the discrete angular spectrum of the light field is not sufficiently sparse, as shown in Figure 1 (upper right). Thus, simply applying one of the existing sparse Fourier algorithms to light field reconstruction would only recover the large frequencies and miss many of the small ones, producing poor results. We handle this by optimizing for sparse coefficients in the continuous domain.

3. SPARSITY IN THE DISCRETE VS. CONTINUOUS FOURIER DOMAIN

In this section, we show how the discretization of a signal that is sparse in the continuous Fourier domain results in a loss of sparsity. We then give an overview of our approach for recovering sparse continuous spectra. In subsequent sections, we will describe in detail one application of this theory, namely reconstructing full 4D light fields from a few 1D viewpoint segments. We will also show results of this application on real light field data.

A signal $x(t)$ of length N is k-sparse in the continuous Fourier domain if it can be represented as a combination of k nonzero continuous frequency coefficients:

$$x(t) = \frac{1}{N}\sum_{i=0}^{k} a_i \exp\left(\frac{2\pi j t \omega_i}{N}\right), \tag{1}$$

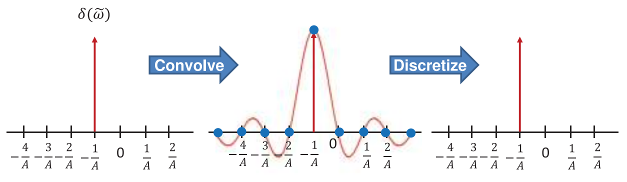
where $\{\omega_i\}_{i=0}^{k}$ are the continuous positions of frequencies (i.e., each ω_i is not necessarily an integer), and $\{a_i\}_{i=0}^{k}$ are the coefficients or values corresponding to these frequencies. The same signal is sparse in the discrete Fourier domain only if all of the ω_i's happen to be integers. In this case, the output of its N-point DFT has only k nonzero coefficients. Consequently, any signal that is k-sparse in the discrete Fourier domain is also k-sparse in the continuous Fourier domain; however, as we will show next, a signal that is sparse in the continuous Fourier domain is not necessarily sparse in the discrete Fourier domain.

3.1 The Windowing Effect

The windowing effect is a general phenomenon that occurs when one computes the discrete Fourier transform (DFT) of a signal using a finite window of samples. Since it is not limited to the light field, we will explain the concept using 1D signals. It naturally extends to higher dimensions.

Consider computing the discrete Fourier transform of a time signal $\mathbf{y}(t)$. To do so, we would sample the signal over a time window $[-\frac{A}{2}, \frac{A}{2}]$, then compute the DFT of the samples. Since the samples come from a limited window, it is as if we multiplied the original signal $\mathbf{y}(t)$ by a box function that is zero everywhere outside of this acquisition window. Multiplication in the time domain translates into convolution in the Fourier domain. Since acquisition multiplies the signal by a box, the resulting DFT returns the spectrum of the original signal $\mathbf{y}(t)$ convolved with a sinc function.

Convolution with a sinc, in most cases, significantly reduces the sparsity of the original signal. To see how, consider a simple example where the signal $\mathbf{y}(t)$ is one sinusoid, namely, $\mathbf{y}(t) = \exp(-2j\pi\tilde{\omega}t)$. The frequency domain of this signal has a single impulse at $\tilde{\omega}$. Say we sample the signal over a window $[-\frac{A}{2}, \frac{A}{2}]$, and take its DFT. The spectrum will be convolved with a sinc, as explained earlier. The DFT will discretize this spectrum to the DFT grid points located at integer multiples of $\frac{1}{A}$. Because a sinc function of width $\frac{1}{A}$ has zero


$\delta(\tilde{\omega})$
Convolve
Discretize
$-\frac{4}{A}$ $-\frac{3}{A}$ $-\frac{2}{A}$ $-\frac{1}{A}$ 0 $\frac{1}{A}$ $\frac{2}{A}$
(a) frequency lies on a DFT grid point

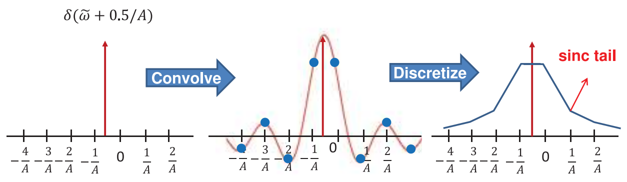

$\delta(\tilde{\omega} + 0.5/A)$
Convolve
Discretize
sinc tail
$-\frac{4}{A}$ $-\frac{3}{A}$ $-\frac{2}{A}$ $-\frac{1}{A}$ 0 $\frac{1}{A}$ $\frac{2}{A}$
(b) frequency does not lie on a DFT grid point

Fig. 2. The windowing effect: limiting samples to an aperture A is equivalent to convolving the spectrum with a sinc function. (a) If a frequency spike lies on a DFT grid point, then the sinc disappears when it is discretized and the original sparsity of the spectrum is preserved; (b) if the frequency spike is not on the DFT grid, once we discretize we get a sinc tail and the spectrum is no longer as sparse as in the continuous domain.

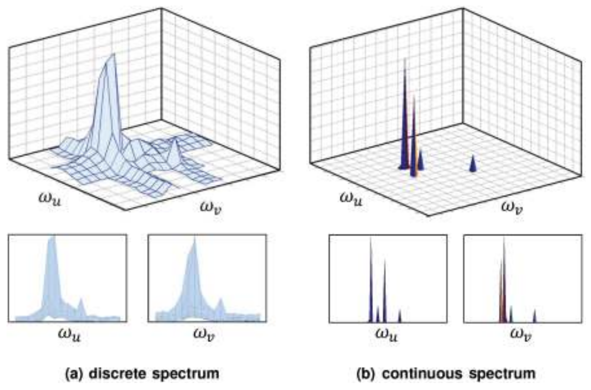


Fig. 3. A 2D angular slice of the 4D light field spectrum of the Stanford crystal ball for $(\omega_x, \omega_y) = (50, 50)$. (a) In the discrete Fourier domain, we have sinc tails and the spectrum is not very sparse; (b) in the continuous Fourier domain, as reconstructed by our algorithm, the spectrum is much sparser. It is formed of four peaks that do not fall on the grid points of the DFT.

crossings at multiples of $\frac{1}{A}$ (as can be seen in Figure 2(a)), if $\tilde{\omega}$ is an integer multiple of $\frac{1}{A}$ then the grid points of the DFT will lie on the zeros of the sinc($\cdot$) function and we will get a single spike in the output of the DFT. However, if $\tilde{\omega}$ is a not an integer multiple of $\frac{1}{A}$, then the output of the DFT will have a sinc tail as shown in Figure 2(b).

Like most natural signals, the sparsity of natural light fields is not generally aligned with any sampling grid. Thus, the windowing effect is almost always observed in the DFT of light fields along spatial and angular dimensions. Consider the effect of windowing in the angular domain (which tends to be more limited in the number of samples and consequently exhibits a stronger windowing effect). Light fields are sampled within a limited 2D window of $\mathbf{uv}$ coordinates. As a result, the DFT of each 2D angular slice, $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(\omega_u, \omega_v)$, is convolved with a 2D sinc function, reducing sparsity. Figure 3(a) shows the DFT of an angular slice from the crystal ball light field. As can be seen in the figure, the slice shows a sparse number of peaks but these peaks exhibit tails that decay very slowly. These tails ruin sparsity and make reconstruction more difficult. We propose an approach to light field reconstruction that removes the windowing effect by optimizing for sparsity in the continuous spectrum. Figure 3(b) shows a continuous Fourier transform of the same crystal ball slice, recovered using our approach. Note that the peak tails caused by windowing have been removed and the underlying sparsity of light fields has been recovered.

3.2 Recovering the Sparse Continuous Fourier Spectrum

From sparse recovery theory we know that signals with sparse Fourier spectra can be reconstructed from a number of time samples proportional to sparsity in the Fourier domain [Candès et al. 2006b; Donoho 2006; Hassanieh et al. 2012a]. Most practical sparse recovery algorithms work in the discrete Fourier domain, however, as described previously, the nonzero frequency coefficients of most light fields are not integers. As a result, the windowing effect ruins sparsity in the discrete Fourier domain and can cause existing sparse recovery algorithms to fail. Our approach is based on the same principle of sparse recovery, but operates in the continuous Fourier domain where the sparsity of light fields is preserved.

Recall our model in Eq. (1) of a signal that is sparse in the continuous Fourier domain. Given a set of discrete time samples of $x(t)$, our goal is to recover the unknown positions $\{\omega_i\}_{i=0}^k$ and values $\{a_i\}_{i=0}^k$ of the nonzero frequency coefficients. From Eq. (1), we see that this problem is linear in the values $\{a_i\}_{i=0}^k$ and nonlinear in the positions $\{\omega_i\}_{i=0}^k$ of the nonzero coefficients. Thus, to recover the values and positions, we use a combination of a linear and nonlinear solver.

Recovering Coefficient Values $\{a_i\}_{i=0}^k$. If we know the positions of nonzero coefficients (i.e., each ω_i) then Eq. (1) becomes a system of linear equations with unknowns $\{a_i\}_{i=0}^k$, and, given $> k$ discrete samples of $x(t)$, we can form an overdetermined system allowing us to solve for each a_i.

Recovering Continuous Positions $\{\omega_i\}_{i=0}^k$. We use nonlinear gradient descent to find the continuous positions $\{\omega_i\}_{i=0}^k$ that minimize the square error between observed discrete samples of $x(t)$ and the reconstruction of these samples given by our current coefficient positions and values. Thus, the error function we wish to minimize can be written as

$$e = \sum_t \left\| x(t) - \frac{1}{N} \sum_{i=0}^k \tilde{a}_i \exp\left(\frac{2\pi j t \tilde{\omega}_i}{N}\right) \right\|^2, \tag{2}$$

where $\tilde{a}_i$ and $\tilde{\omega}_i$ are our estimates of a_i and ω_i and where the preceding summation is taken over all the observed discrete samples.

As with any gradient descent algorithm, in practice, we begin with some initial guess of discrete integer positions $\{\omega_i^{(0)}\}_{i=0}^k$. In Section 5.2 we describe the initialization used to generate the results in this article, but other initializations are also possible. From this initial guess, we use gradient descent on $\{\omega_i\}_{i=0}^k$ to minimize our error function. In practice, the gradient is approximated using finite differences. In other words, we calculate error for perturbed peak locations $\{\omega_i^{(j)} + \delta_i\}$ and update our $\{\omega_i^{(j+1)}\}$ with the ϵ_i that result in the smallest error. We keep updating until the error converges.

Once we have recovered both a_i and ω_i, we can reconstruct the signal $x(t)$ for any sample t using Eq. (1).

4. LIGHT FIELD NOTATION

A 4D light field $\mathbf{L}(x, y, u, v)$ characterizes the light rays between two parallel planes, namely, the **uv** camera plane and the **xy** image plane, which we refer to as angular and spatial dimensions, respectively. Each (u, v) coordinate corresponds to the location of the viewpoint of a camera and each (x, y) coordinate corresponds to a pixel location. $\widehat{\mathbf{L}}(\omega_x, \omega_y, \omega_u, \omega_v)$ characterizes the 4D spectrum of this light field. We will use $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(\omega_u, \omega_v)$ to denote a 2D angular slice of this 4D spectrum for fixed spatial frequencies (ω_x, ω_y). Similarly, $\mathbf{L}_{u,v}(x, y)$ denotes the 2D image captured by a camera with its center of projection at location (u, v). Table I presents a list of terms used throughout this article.

Table I. Notation

Term	Definition
u, v	angular/camera plane coordinates
x, y	spatial plane coordinates
ω_u, ω_v	angular frequencies
ω_x, ω_y	spatial frequencies
$\mathbf{L}(x, y, u, v)$	4D light field kernel
$\widehat{\mathbf{L}}(\omega_x, \omega_y, \omega_u, \omega_v)$	4D light spectrum
$\widehat{\mathbf{L}}_{\omega_x,\omega_y}(\omega_u, \omega_v)$	a 2D angular slice of the 4D light spectrum
$\widehat{\mathbf{L}}_{\omega_x,\omega_y}(u, v)$	a 2D slice for fixed spatial frequencies
$\mathbf{X}$	2D slice $= \widehat{\mathbf{L}}_{\omega_x,\omega_y}(u, v)$
S	set of samples (u, v)
$\mathbf{X}_{\|S}$	2D $\mathbf{X}$ with only samples in S
$\mathbf{x}_S$	$\mathbf{X}_{\|S}$ reordered as $1 \times \|S\|$ vector
P	set of frequency positions (ω_u, ω_v)
$\widehat{\mathbf{x}}_P$	$1 \times \|P\|$ vector of frequency coefficients
F	set of positions and coefficients (ω_u, ω_v, a)
$[N]$	the set $\{0, 1, \ldots N-1\}$
$\mathbf{y}$	1D signal or line segment
$M \times M$	number of image pixels in spatial domain
$N \times N$	number of camera locations

5. LIGHT FIELD RECONSTRUCTION FROM 1D VIEWPOINT TRAJECTORIES

To demonstrate the power of sparse recovery in the continuous Fourier domain, we show how it can be used to reconstruct light fields from 1D viewpoint trajectories. We choose to work with 1D trajectories because it simplifies the initialization of our gradient descent. However, the continuous Fourier recovery described in the previous section is general and does not require this assumption.

In this section we describe the initialization used for our experiments as well as our implementation of the continuous Fourier recovery. The high-level structure of the algorithm described in this section is given by the pseudocode in Algorithm 5.1. More pseudocode describing the subroutines used in Algorithm 5.1 can be found in Appendix A.

5.1 Input

Our input is restricted to 1D viewpoint trajectories that consist of discrete lines. A discrete line in the angular domain is defined by the set of (u, v) points such that

$$\begin{cases} u = \alpha_u t + \tau_u \mod N \\ v = \alpha_v t + \tau_v \mod N \end{cases}, \quad \text{for } t \in [N], \tag{3}$$

where $0 \leq \alpha_u, \alpha_v, \tau_u, \tau_v < N$ and $\mathrm{GCD}(\alpha_u, \alpha_v) = 1$.

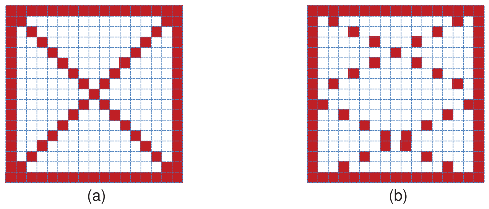


Fig. 4. Sampling patterns: Our algorithm samples the (u, v) angular domain along discrete lines. (a) Box and 2 diagonals; (b) box and 2 lines with slopes $= \pm 2$. Note that in this case the discrete line wraps around.

ALGORITHM 5.1: Light Field Reconstruction Algorithm

```
procedure SparseLightField($\mathbf{L}_{|S}$)
    $\widehat{\mathbf{L}}_{u,v}(\omega_x, \omega_y) = \text{FFT}(\mathbf{L}_{u,v}(x, y))$ for $u, v \in S$
    for $\omega_x, \omega_y \in [M]$ do
        $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(\omega_u, \omega_v) = \text{2DSparseFFT}(\widehat{\mathbf{L}}_{\omega_x,\omega_y}(u, v)_{|S})$
    $\mathbf{L}(x, y, u, v) = \text{IFFT}(\widehat{\mathbf{L}}(\omega_x, \omega_y, \omega_u, \omega_v))$
    return L
procedure 2DSparseFFT($\mathbf{X}_{|S}$)
    $P = \text{SparseDiscreteRecovery}(\mathbf{X}_{|S})$
    $F, e = \text{SparseContinousRecovery}(\mathbf{X}_{|S}, P)$
    $\mathbf{X}(u, v) = \sum_F a \cdot \exp\left(2j\pi \frac{u\omega_u + v\omega_v}{N}\right)$ for $u, v \in [N]$
    $\widehat{\mathbf{X}} = \text{FFT}(\mathbf{X})$
    return $\widehat{\mathbf{X}}$
```

This lets us use the Fourier projection slice theorem to recover a sparse discrete spectrum that we use to initialize our gradient descent. Figure 4 shows the specific sampling patterns used in our experiments.

For a light field $\mathbf{L}(x, y, u, v)$, our algorithm operates in the intermediate domain $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(u, v)$ that describes spatial frequencies as a function of viewpoint. We start by taking the 2D DFT of each input image, which gives us the spatial frequencies (ω_x, ω_y) at a set of viewpoints S consisting of our 1D input lines. We call this set of known samples $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(u, v)_{|S}$. Our task is to recover the 2D angular spectrum $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(\omega_u, \omega_v)$ for each spatial frequency (ω_x, ω_y) from the known samples $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(u, v)_{|S}$. For generality and parallelism we do this at each spatial frequency independently, but one could possibly use a prior on the relationship between different (ω_x, ω_y) to improve our current implementation.

5.2 Initialization

The goal of our initialization is to calculate some initial guess for the positions $\{\omega_i^{(0)}\}_{i=0}^k$ of our nonzero frequency coefficients. We do this by using the Fourier projection slice theorem in a voting scheme similar to a Hough transform. By the projection slice theorem, taking the DFT of an input discrete line gives us the projection of our light field spectrum onto the line. Each projection gives the sum of several coefficients in our spectrum. Different projections provide us with different sums, and each sum above a given threshold votes for the discrete positions of coefficients that it sums. Our initialization then selects the discrete positions that receive a vote from every input projection and returns these as its initial guess. We refer to this initialization as sparse discrete recovery.

5.2.1 *Computing Projections.* To simplify our discussion of slices, we will use $\mathbf{X}$ to denote the 2D slice $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(u, v)$ in our intermediate domain and $\widehat{\mathbf{X}}$ to denote its DFT, $\widehat{\mathbf{L}}_{\omega_x,\omega_y}(\omega_u, \omega_v)$. Thus, our input is given by a subset of sample slices $\mathbf{X}_{|S}$, where the set S gives the coordinates of our input samples ((u, v) viewpoint positions).

For each slice $\mathbf{X}$ in our input $\mathbf{X}_{|S}$, the views in $\mathbf{X}$ lie on a discrete line. We perform a 1D DFT for each of these discrete lines, which yields the projection of our 2D spectrum onto a corresponding line in the Fourier domain. Specifically, let y be the 1D discrete line corresponding to a 2D slice $\mathbf{X}$ (parameterized by $t \in [N]$)

$$\mathbf{y}(t) = \mathbf{X}(\alpha_u t + \tau_u \bmod N, \alpha_v t + \tau_v \bmod N), \tag{4}$$

where $0 \le \alpha_u, \alpha_v, \tau_u, \tau_v < N$ and $\text{GCD}(\alpha_u, \alpha_v) = 1$.

Then, $\widehat{\mathbf{y}}$, the DFT of $\mathbf{y}$, is a projection of $\widehat{\mathbf{X}}$ onto this line, that is, each point in $\widehat{\mathbf{y}}$ is a summation of the N frequencies that lie on a discrete line orthogonal to $\mathbf{y}$, as shown in Figure 6. Specifically, the frequencies (ω_u, ω_v) that satisfy $\alpha_u\omega_u + \alpha_v\omega_v = \omega \mod N$ project together onto $\widehat{\mathbf{y}}(\omega)$ (recall that discrete lines may "wrap around" the input window).

5.2.2 *Voting.* To recover the discrete positions of the non-zero frequency coefficients, we use a voting approach. For each line projection, the projected sums that are above some threshold vote for the frequencies that map to them (similar to a Hough transform). Since the spectrum is sparse, most projected values are very small and only the coefficients of large frequencies receive votes from every line projection. Thus, by selecting frequencies that receive votes from every projection, we get an estimate of the discrete positions of non-zero coefficients.

To better illustrate how voting works, consider the simple example shown in Figure 7(a). The 2D spectrum has only 3 large frequencies at $(5, 5)$, $(5, 9)$, and $(9, 5)$. When we project along the rows of our grid, the $5th$ and $9th$ entries of the projection will be large and this projection will vote for all frequencies in the $5th$ and $9th$ columns. Similarly, when we project along columns, the projection will vote for all frequencies in the $5th$ and $9th$ rows. At this point, frequencies $(5, 5)$, $(5, 9)$, $(9, 5)$, $(9, 9)$ have two votes. However, when we project on the diagonal, frequency $(9, 9)$ will not get a vote. After 3 projections, only the 3 correct frequencies get 3 votes. Another example is shown in Figure 7(b).

5.3 Optimization in the Continuous Fourier Domain

Recall from Section 3.2 that our optimization takes the initial positions $\{\omega_i^{(0)}\}_{i=0}^k$ and a subset of discrete samples as input. With both provided by the input and initialization described earlier, we now minimize the error function of our reconstruction using the gradient descent approach outlined in Section 3.2.

5.3.1 *Recovering Frequency Coefficients.* As we discussed in Section 3.2, when we fix the coefficient positions $\{\omega_i^{(0)}\}_{i=0}^k$, Eq. (1) becomes linear in the coefficient values $\{a_i\}_{i=0}^k$. To solve for the full light field spectrum at each iteration of our gradient descent, we express each of our known discrete input samples as a linear combination of the complex exponentials given by our current choice of $\{\omega_i\}_{i=0}^k$.

With the appropriate system of linear equations, we can solve for the coefficient values that minimize the error function described in Eq. (2). To construct our system of linear equations, we concatenate the discrete input (u, v) samples from $\mathbf{X}_{|S}$ into an $|S| \times 1$ vector that we denote as $\mathbf{x}_S$. Given the set P of frequency positions, we let $\widehat{\mathbf{x}}_P$ be the $|P| \times 1$ vector of the frequency coefficients we want to recover (with each coefficient in $\widehat{\mathbf{x}}_P$ corresponding to a frequency position in P). Finally, let $\mathbf{A}_P$ be a matrix of $|S| \times |P|$ entries. Each row of $\mathbf{A}_P$ corresponds to a (u, v) sample, each column corresponds to an (ω_u, ω_v) frequency, and the value of each entry is given by a

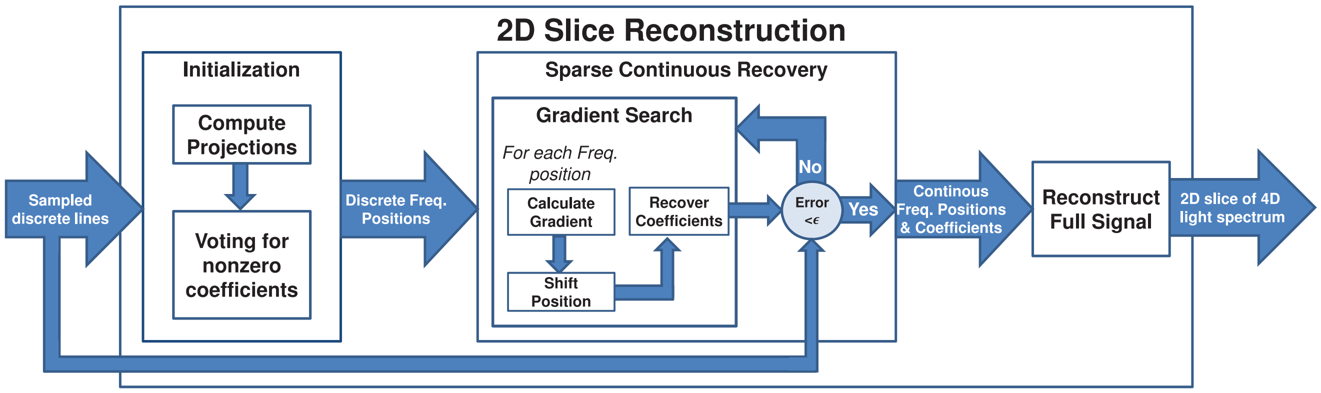


Fig. 5. Flowchart of the 2D sparse FFT reconstruction algorithm. The algorithm takes a set of sampled discrete lines. The initialization, Discrete Recovery, has 2 steps: computing the projections and recovering the discrete positions of the large frequency coefficients. In the sparse continuous recovery, the gradient search tries to shift the positions of the frequencies to noninteger locations and recover their coefficients. We keep repeating this gradient search until we get a small enough error. This stage will output a list of continuous frequency positions and their coefficients, which can then be used to reconstruct the full 2D slice.

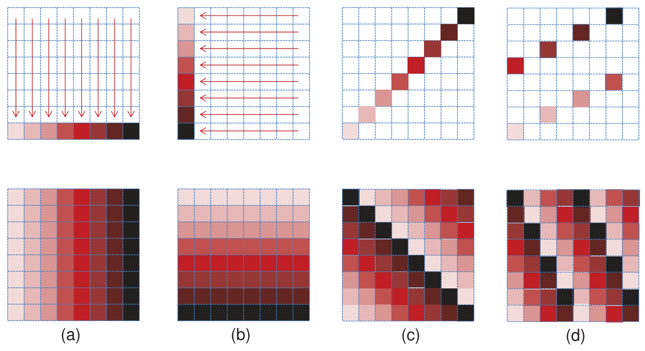


Fig. 6. Computing the DFT of a discrete line of a 2D signal is equivalent to projecting the 2D spectrum onto the line. The top row of figures shows the sampled lines and the bottom row of figures shows how the spectrum is projected. Frequencies of the same color are projected onto the same point. (a) Row projection; (b) column projection; (c) diagonal projection; (d) line with slope = 2.

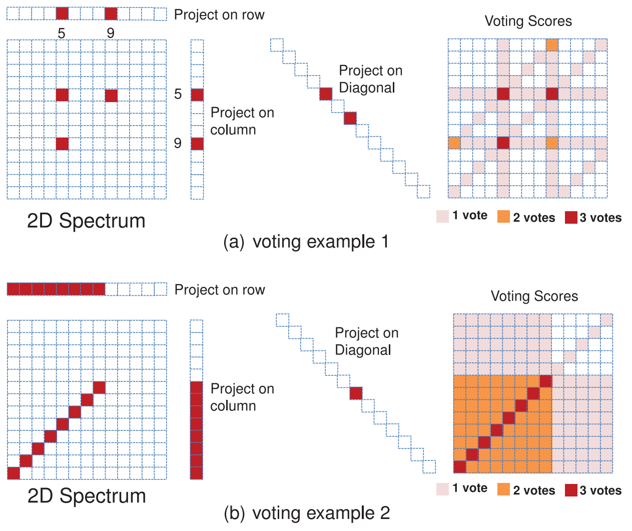


Fig. 7. Two examples of the voting approach used to recover the discrete positions of the large frequencies from projections on discrete lines. The 2D spectrum is projected on a row, a column, and a diagonal. Each large projection votes for the frequencies that map to it. Using only projections on a row and column, many frequencies get two votes. By adding a 3rd projection on the diagonal, only the large frequencies get 3 votes. (a) Frequencies (5,5), (5,9), and (9,5) are large and only they get 3 votes; (b) some frequencies on the diagonal are large and only these frequencies get 3 votes.

complex exponential.

$$\mathbf{A}_P((u,v),(\omega_u,\omega_v)) = \exp\left(2j\pi\frac{u\omega_u + v\omega_v}{N}\right). \qquad (5)$$

Now our system of linear equations becomes

$$\mathbf{x}_S = \mathbf{A}_P\widehat{\mathbf{x}}_P, \qquad (6)$$

and we use the pseudo-inverse of $\mathbf{A}_P$ to calculate the vector $\widehat{\mathbf{x}}^*_P$ of coefficient values that minimize our error function (i.e., $\widehat{\mathbf{x}}^*_P = \mathbf{A}^\dagger_P\mathbf{x}_S$).

Recall that we did not specify the threshold used to determine a "vote" in our initialization. Rather than using a fixed threshold, we choose the smallest threshold such that the preceding system of equations becomes well determined.

5.3.2 *Gradient Descent.* Recall from Section 3.2 that our gradient descent algorithm minimizes the error function, which we can now rewrite as

$$\text{minimize } \; e(P) = ||\mathbf{x}_S - \mathbf{A}_P\mathbf{A}^\dagger_P\mathbf{x}_S||^2. \qquad (7)$$

In the previous equation, the frequency positions in the list P are continuous, but the input samples $\mathbf{x}_S$ that we use to compute our error are discrete. Thus, our optimization minimizes error in the *discrete* reconstruction of our light field by finding optimal *continuous* frequency positions.

In our gradient descent, each iteration of the algorithm updates the list of frequency positions P. For each recovered frequency position in P, we fix all other frequencies and shift the position of this frequency by a small fractional step $\delta \ll 1$. We shift it in all eight directions as shown in Figure 8 and compute the new error $e(P)$ given the new position. We then pick the direction that best minimizes the error $e(P)$ and change the position of the frequency

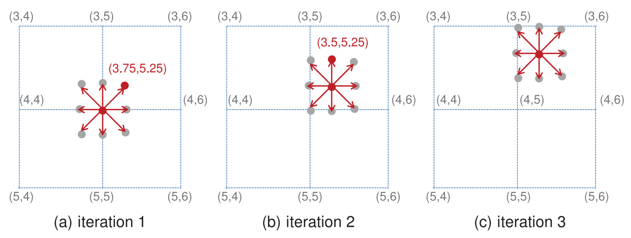


Fig. 8. The gradient descent algorithm shifts the frequency by a small step in every iteration. The frequency is shifted in the direction that minimizes the error. (a) Frequency (4,5) is shifted to a noninteger position that best minimizes the error; (b) the frequency is shifted again to minimize the error; (c) the frequency position converges since shifting in any direction will increase the error.

in this direction. If none of the directions minimizes the error, we do not change the position of this frequency. We repeat this for every frequency position in P.

Our gradient descent ensures that, from iteration i to iteration $(i + 1)$, we always reduce error, namely, $e(P^{(i+1)}) < e(P^{(i)})$. The algorithm keeps iterating over the frequencies until the error $e(P)$ falls below a minimum acceptable error ϵ. Once we have a final list of continuous frequency positions, we can recover their coefficients as described in Section 5.3.1. Pseudocode of this gradient search and sparse continuous recovery is provided in Appendix A.

5.4 Reconstructing the Viewpoints

As explained in Section 3.2, once we have the continuous positions and values of our nonzero frequency coefficients, we can reconstruct the missing viewpoints by expressing Eq. (1) in terms of our data.

$$L_{\omega_x,\omega_y}(u,v) = \sum_{(a,\omega_u,\omega_v)\in F} a \cdot \frac{1}{N} \exp\left(2j\pi \frac{u\omega_u + v\omega_v}{N}\right) \quad (8)$$

By setting (u, v) to the missing viewpoints, we are able to reconstruct the full light fields. Figure 5 shows a flowchart of the entire reconstruction.

Note that the previous equation lets us reconstruct any (u, v) position. We can interpolate between input views and even extend our reconstruction to images that are outside the convex hull of our input. This would not be possible if our sparse coefficients were limited to the discrete Fourier domain, since the preceding equation would be periodic modulo N. This would create a wrapping effect, and attempting to reconstruct views outside the span of our input would simply replicate views inside the span of our input. In other words, the discrete spectrum assumes that our signal repeats itself outside of the sampling window, but, by recovering the continuous spectrum, we can relax this assumption and extend our reconstruction to new views.

6. RESULTS

We experimented with several datasets where full 4D coverage of the light field was available for comparison. For each of these datasets we extracted a small number of 1D segments that we then used to reconstruct the full 2D set of viewpoints. We compare our reconstructed light fields against the complete original datasets in our accompanying videos.

Three of our datasets, the Stanford Bunny, the Amethyst, and the Crystal Ball datasets, were taken from the Stanford light field archive [Stanford 2008]. Each of the Stanford datasets consists of a 17×17 grid of viewpoints and was reconstructed using the box-and-X pattern shown in Figure 4(a). On these datasets, we performed our reconstruction in the YUV color space. The U and V channels were reconstructed at half the resolution of the Y channel.

To show how our method scales with the number of input images we are given, we captured a larger dataset (the Gnome) consisting of 51×51 viewpoints. This dataset was captured using a robotic gantry similar to the one from Stanford [2008], and the double X pattern in Figure 4(b) was used to select our input. The total number of input images is the same for both the single X pattern and the double X pattern, as the effective spacing between input views along diagonals is changed. For this dataset our input consists of less than 12% of the original images.

Our code was designed for flexible experimentation and is currently slow. However, the algorithm is highly parallelizable and we run it on a PC cluster. The code is written in C++ using the Eigen library. The ω_u, ω_v slices of a light field are divided among different machines, and the results are collected once all of the machines have finished.

Load balancing, variability in the number of machines used, and different convergence characteristics for different inputs make it difficult to estimate exact runtimes. Using a cluster of up to 14 machines at a time (averaging 5–6 cores each), typical runtimes ranged from 2 to 3 hours for a colored dataset (three channels). There are several ways to accelerate the method - for example, one could leverage the coherence across slices or replace finite differences with a method that converges faster, but we leave this for future work.

6.1 Viewing Our Results

Our results are best experienced by watching the accompanying videos, or by using our interactive light field viewer. Code for this interactive viewer was downloaded from the Stanford light field archive [Stanford 2008], but the datasets we provide are our own reconstructions.

6.2 The Stanford Bunny

The Stanford Bunny dataset is our simplest test case. The scene is Lambertian and therefore especially sparse in the frequency domain. The spacing between input views is also very narrow, so there is little aliasing. Each image is 512×512 pixels.

Our reconstruction of the Bunny is difficult to distinguish from the full light field captured by Stanford [2008], as shown in Figure 9. Figure 11 shows that the reconstruction error is small.

6.3 Amethyst

The Amethyst dataset is highly non-Lambertian. It exhibits both specular reflections and refraction. Again, it is difficult to distinguish our reconstruction from the full captured light field, as shown in Figure 10. We reconstruct most of the reflected details, with the exception of some undersampled features that move so fast they do not appear at all in the input. Figure 15 gives an example.

6.4 Crystal Ball

The Crystal Ball scene is extremely non-Lambertian, exhibiting caustics, reflections, specularities, and nonlinear parallax. We are able to reproduce most of the complex properties that make this scene shown in Figure 12 so challenging, as can be seen in our accompanying video or interactive viewer.

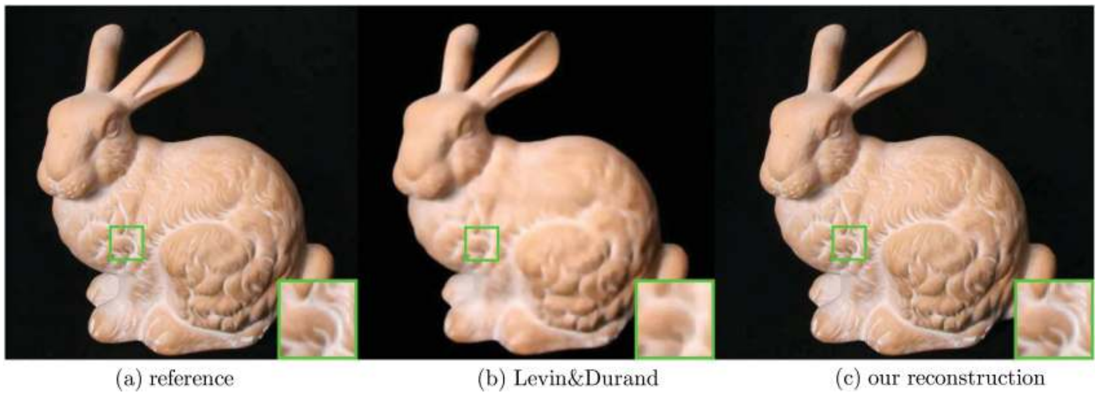


Fig. 9. The reconstruction of Stanford Bunny dataset from one (u, v) viewpoint. On the left and in the middle are the reference figure and the reconstruction from Levin and Durand [2010] (courtesy of Levin). The sampling pattern we used is the box-and-X pattern shown in Figure 4(a). Though we used more samples than Levin and Durand [2010] used, we did a much better job in terms of less blurring, preserving the textures, and having less noise.

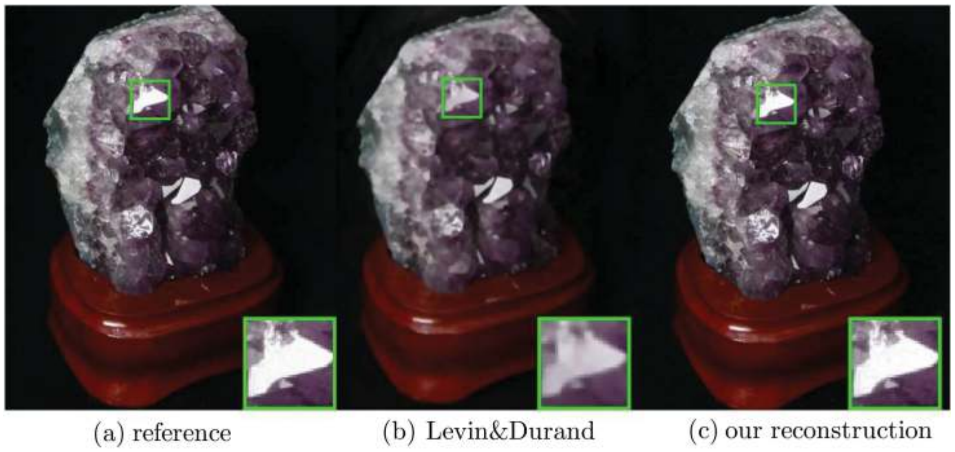


Fig. 10. The reconstruction of Amethyst dataset (right), the reference figure (left), and the reconstruction from Levin and Durand [2010] (middle, courtesy of Levin). We are using the box-and-X sampling pattern shown in Figure 4(a), which is more than the number of samples Levin and Durand used. However, we are able to reconstruct this highly non-Lambertian view and it is hard to distinguish our reconstruction from the full captured light field.

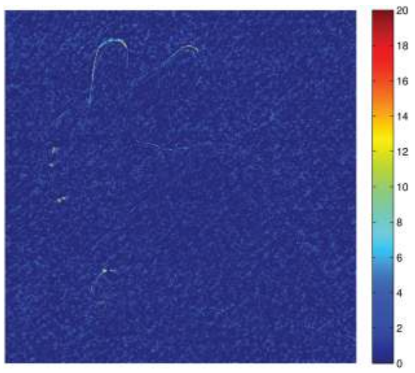


Fig. 11. A color map of the difference in the Y channel between a reference view and a reconstructed view from the Stanford Bunny dataset. In about half of the pixels, the difference is zero. There is some unstructured noise, but it is hard to tell whether this comes from the reference figure or our reconstruction. There is also some structured noise on the edge of the bunny, but again, it is difficult to tell whether this comes from reconstruction error or an error in the pose estimate of the reference image.

If one looks closely, our reconstruction of this light field contains a small amount of structured noise. We believe this happens because the underlying spectrum is less sparse, which we discuss more in Section 7.

6.5 Gnome

We acquired a new dataset consisting of 52×52 viewpoints. The resolution of each image is 640×480, and we reconstructed all channels at full resolution.

The Gnome scene is mostly Lambertian with a few specular highlights. In terms of the subject being captured, the difficulty of this scene sits somewhere between the Stanford Bunny and the Amethyst datasets. However, what makes this data more challenging is the level of noise in our input. The captured images of the Gnome have noticeable shot noise, flickering artifacts, and registration errors ("camera jitter"). Since these artifacts are not sparse in the frequency domain, our algorithm does not reproduce them in the output shown in Figure 13. For most of these artifacts, the result is a kind of denoising, making our output arguably better than the reference images available for comparison. This is especially clear in the case of camera jitter, where the effect of denoising can be seen

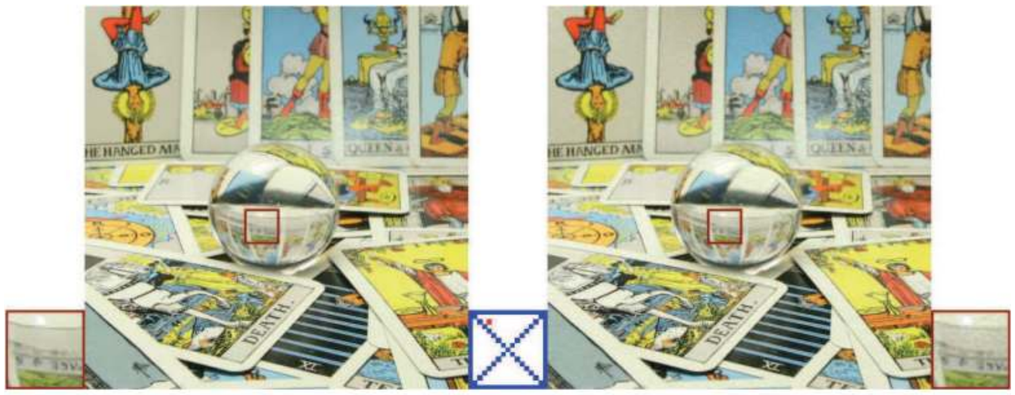


Fig. 12. One (u, v) view from the reconstruction of the Crystal Ball dataset. We are using the box plus diagonals sampling pattern (as shown in the blue box in the center). The red dot shows the position of reconstructed view in the angular domain. Despite the fact that the scene is extremely non-Lambertian and has complex structures, we are still able to reconstruct most details of the light field.

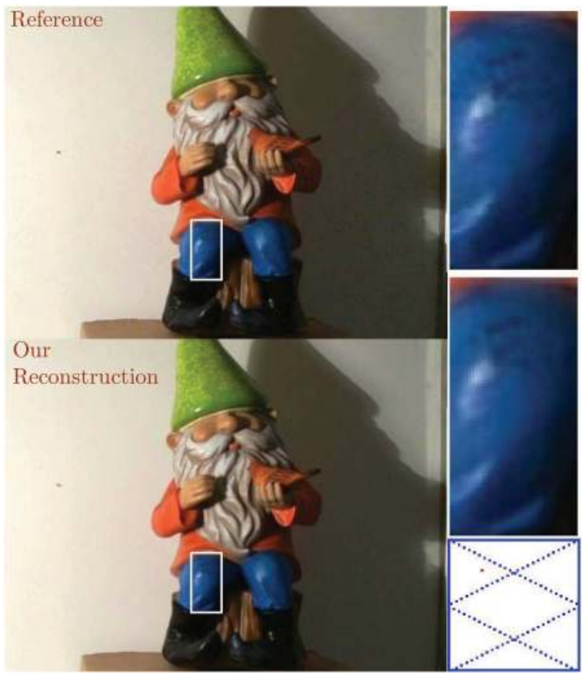


Fig. 13. One (u, v) view from our reconstruction of the Gnome dataset. We use the sample pattern from Figure 4(b), as shown by the blue box in the bottom right. The red marker shows where the view is in the angular domain. Although the captured dataset is noisy, we are still able to reconstruct it in good detail.

clearly in an epipolar image shown in Figure 16. However, some of the shot noise in our input is reconstructed with greater structure. We have a more general discussion of noise in Section 7.

6.6 Extending Views

Reversing the windowing effect in the second step of our algorithm makes it possible to reconstruct views outside the original window of

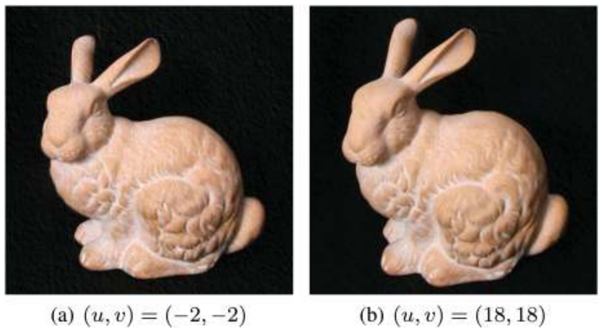


Fig. 14. Extending views: We extend our reconstruction of the Stanford Bunny dataset (Figure 9) and extend the camera views. The original view is $0 \leq u \leq 16$ and $0 \leq u \leq 16$, and here we show our extension to $(-2, -2)$ and $(18, 18)$.

our input. To demonstrate, we extend each of the u and v dimensions in our Bunny dataset by an additional 4 views, increasing the size of our reconstructed aperture by 53% (see Figure 14). These results are best appreciated in our accompanying video.

6.7 Informal Comparison with Levin and Durand [2010]

Like us, Levin and Durand [2010] reconstruct light fields from a 1D set of input images. Their technique is based on a Lambertian prior with unknown depth. We provide an informal comparison with their approach, but the different sampling patterns of the two techniques make it difficult to hold constant the number of input views used by each technique. Levin and Durand's reconstruction uses fewer images but is restricted to synthesizing views within the convex hull of input viewpoints. Our sampling patterns use slightly more images, but let us synthesize views outside the convex hull of our input. Small differences in input aside, the comparison in Figure 9 and Figure 10 shows that our reconstruction is less blurry and does not have some of the ringing artifacts that appear in their results.

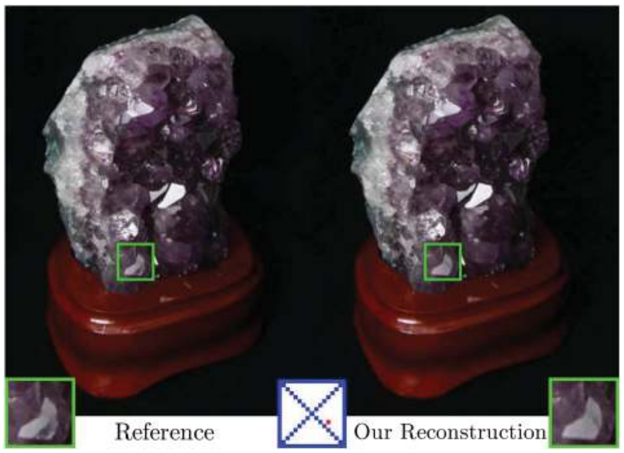


Fig. 15. One example of a reconstructed view from the Amethyst dataset where we lose some reflection details. The missing specular reflection does not appear in any of our input views, so it cannot be recovered.

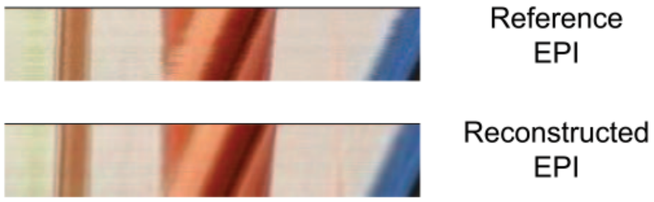


Fig. 16. Top: We see noise in the u, v dimensions of our reference data caused by registration errors. This error shows up as camera shake in the reference images. Bottom: Our algorithm effectively removes this noise in the reconstruction, essentially performing camera stabilization.

7. DISCUSSION

7.1 Viewpoint Denoising

One advantage of reconstruction based on a sparse prior is the potential for denoising. Noisy input tends to create low-power high frequencies that are not part of our scene. These frequencies make the spectrum less sparse and are usually zeroed out by our algorithm.

Since our reconstruction is based on sparsity in the ω_u, ω_v domain, we remove noise in u, v. This noise corresponds to "jitter" usually caused by registration errors or camera shake. We can see the effect of this denoising by examining a v, y slice of our light field, like the one in Figure 16. These slices are often referred to as Epipolar Plane Images (EPIs) in computer vision. To observe the visual effect of this denoising, the reader should watch our accompanying Gnome video. The reconstructed camera motion in this video is much smoother than the reference camera motion. One way to think of this effect is as a kind of video stabilization.

Our ability to denoise in u, v is limited by the number of input slices we have and the sparsity of the spectrum we are reconstructing. If the noise affects the sparsity of our scene too much, some of its power might be projected onto existing spikes from our signal, changing their estimated power. We can see some of this in the Gnome dataset, where some of the shot noise in our input is reconstructed with slight structure along the dominant orientation of our scene.

7.2 Importance of *Continuous* Fourier Recovery

To better understand how operating in the continuous Fourier domain affects our reconstruction, we examine the impact of our continuous recovery on the reconstructed Bunny light field. We choose this light field because our results are almost indistinguishable from the reference data, so we can reasonably assume that the sparsity estimated by our full algorithm reflects the true sparsity of the captured scene.

We first compare our sparse continuous recovery in Figure 17(c) with the sparse discrete recovery used to initialize our gradient descent (shown in Figure 17(a)). The error in Figure 17(a) shows how existing sparse recovery theory is limited by the lack of sparsity in the discrete light field spectrum. However, this result does not necessarily isolate the effects of working in the discrete domain. To better isolate these limits, we generate a third reconstruction in Figure 17(b) by rounding the coefficients of our final reconstruction in Figure 17(c) to the nearest discrete frequency positions and removing the sinc tails that result from this rounding. This reconstruction approximates the discrete spectrum that is closest to our continuous spectrum while exhibiting the same sparsity.

As we see in Figure 17, the effect of discretization predicted by our experiment is a kind of ghosting. To understand why, recall that the discrete Fourier transform assumes that signals are periodic in the primal domain and that, given a finite number of frequencies, our reconstruction will be band limited. As a result, the IDFT will attempt to smooth between images at opposite ends of the primal domain. If we look at Figure 18 we can see this effect across the set of viewpoints in our light field. When viewpoints near the center are averaged (smoothed) with their neighbors, the artifact is less noticeable because their neighbors are very similar. However, when this smoothing wraps around the edges of our aperture, we average between more dissimilar images and the ghosting becomes more severe.

7.3 Potential Applications

Our reconstruction from 1D viewpoint trajectories is directly applicable to capture techniques that seek to acquire a dense 2D sampling of viewpoints on a grid. One could, for instance, use it to significantly reduce the number of cameras needed by a camera array. Alternatively, for applications where light fields are captured by a single moving camera (such as Gortler et al. [1996], Buehler et al. [2001], and Davis et al. [2012]), the algorithm could be used to greatly increase the speed of capture. In both of these cases, the sparse continuous spectrum we recover could also be used as a highly compressed representation of the light field.

The theory of continuous recovery has many potential applications beyond our reconstruction from 1D viewpoint segments. Sparsity in the continuous Fourier domain is a powerful prior more general than Lambertianality, making it an exciting new direction for research. While our choice of initialization uses viewpoint sampling patterns that consist of discrete lines, one can imagine different initialization strategies that work with different input. This input could come from plenoptic or mask-based light field cameras, or even some combination of multiview stereo and image-based rendering algorithms. However, continuous recovery is not necessarily convex, so proper initialization strategies will be an important and possibly nontrivial part of applying our continuous recovery approach to different types of data. We believe this will be an exciting area of future work.

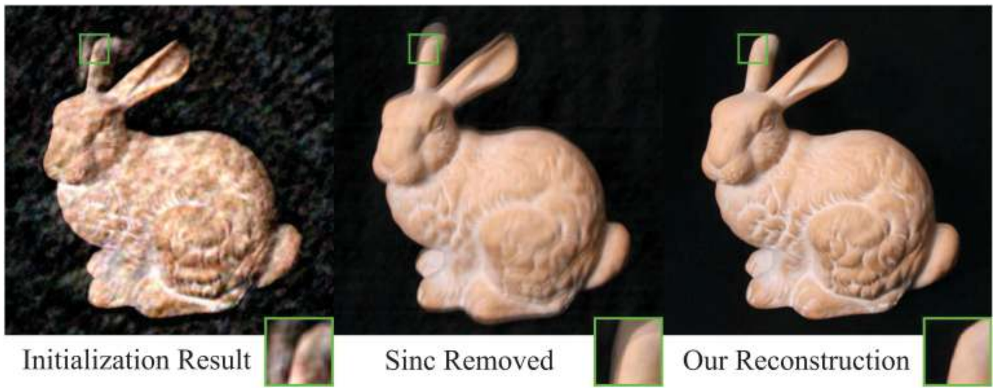


Fig. 17. Comparison of our final reconstruction in the continuous domain to two alternative reconstructions in the discrete domain. We compare our result (right) with the output of only our initialization (left), as well as the discrete approximation of our result with sinc tails removed (middle).

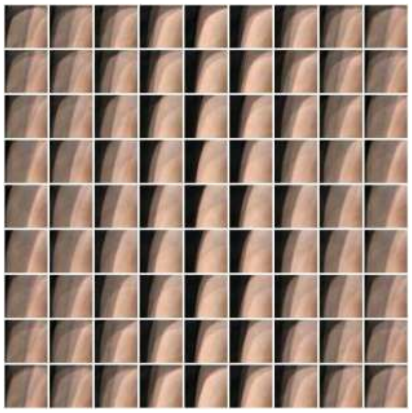


Fig. 18. A demonstration of the ghosting that happens when we simply remove sinc tails in the frequency domain. We removed the sinc tails from the spectrum of the Stanford Bunny dataset and selected the same inset from each u, v image (we chose the same inset as in Figure 17). This figure shows how the inset changes across the (u, v) aperture (note that we subsampled the 17×17 aperture by 2). Ghosting gets worse closer to the edge of the input views.

8. CONCLUSION

We have made the important observation that natural signals like light fields are much sparser in the continuous Fourier domain than in the discrete Fourier domain, and we have shown how this difference in sparsity is the result of a windowing effect. Based on our observations, we presented an approach to light field reconstruction that optimizes for sparsity in the continuous Fourier domain. We then showed how to use this approach to reduce sampling requirements and improve reconstruction quality by applying it to the task of recovering high-quality non-Lambertian light fields from a small number of 1D viewpoint trajectories. We believe our strategy of optimizing for sparsity in the discrete spectrum will lead to exciting new research in light field capture and reconstruction. Furthermore, we hope that our observations on sparsity in the discrete versus continuous domain will have an impact on areas of computational photography beyond light field reconstruction.

APPENDIX

A. PSEUDOCODE

A.1. Initialization: Sparse Discrete Recovery

```
procedure SparseDiscreteRecovery($\mathbf{X}_{|S}$)
    $\widehat{\mathbf{y}}_1 = $ ProjectLine($\mathbf{X}_{|S}, 0, 1, 0, 0$)
    $\widehat{\mathbf{y}}_2 = $ ProjectLine($\mathbf{X}_{|S}, 1, 0, 0, 0$)
    $\widehat{\mathbf{y}}_3 = $ ProjectLine($\mathbf{X}_{|S}, 1, 1, 0, 0$)
    $\widehat{\mathbf{y}}_4 = $ ProjectLine($\mathbf{X}_{|S}, 0, 1, N-1, 0$)
    $\widehat{\mathbf{y}}_5 = $ ProjectLine($\mathbf{X}_{|S}, 1, 0, 0, N-1$)
    $\widehat{\mathbf{y}}_6 = $ ProjectLine($\mathbf{X}_{|S}, 1, -1, 0, N-1$)
    $P = $ RecoverPositions($\widehat{\mathbf{y}}_1, \widehat{\mathbf{y}}_2, \widehat{\mathbf{y}}_3, \widehat{\mathbf{y}}_3, \widehat{\mathbf{y}}_4, \widehat{\mathbf{y}}_5, \widehat{\mathbf{y}}_6$)
    return $P$
procedure ProjectLine($\mathbf{X}_{|S}, \alpha_u, \alpha_v, \tau_u, \tau_v$)
    $\mathbf{y}(i) = \mathbf{X}(i\alpha_u + \tau_u, i\alpha_v + \tau_v)$ for $i \in [N]\}$
    $\widehat{\mathbf{y}} = $ FFT($\mathbf{y}$)
    return $\widehat{\mathbf{y}}$
procedure RecoverPositions($\widehat{\mathbf{y}}_1, \widehat{\mathbf{y}}_2, \widehat{\mathbf{y}}_3, \widehat{\mathbf{y}}_4, \widehat{\mathbf{y}}_5, \widehat{\mathbf{y}}_6$)
    $V_1 = $ Vote($\widehat{\mathbf{y}}_1, 0, 1, 0, 0, \theta$)          ▷ $\theta$:Power threshold
    $V_2 = $ Vote($\widehat{\mathbf{y}}_2, 1, 0, 0, 0, \theta$)
    $V_3 = $ Vote($\widehat{\mathbf{y}}_3, 1, 1, 0, 0, \theta$)
    $V_4 = $ Vote($\widehat{\mathbf{y}}_4, 0, 1, N-1, 0, \theta$)
    $V_5 = $ Vote($\widehat{\mathbf{y}}_5, 1, 0, 0, N-1, \theta$)
    $V_6 = $ Vote($\widehat{\mathbf{y}}_6, 1, -1, 0, N-1, \theta$)
    $P = V_1 \bigcap V_2 \bigcap V_3 \bigcap V_4 \bigcap V_5 \bigcap V_6$
    return $P$
procedure Vote($\widehat{\mathbf{y}}, \alpha_u, \alpha_v, \theta$)
    $I = \{i : ||\widehat{\mathbf{y}}(i)|| > \theta\}$
    $V = \{(\omega_u, \omega_v) : \alpha_u\omega_u + \alpha_v\omega_v = i$ where $i \in I\}$
    return $V$
```

A.2. Sparse Continuous Recovery Algorithm Using Gradient Search

```
procedure SparseContinuousRecovery($\mathbf{X}_{|S}$, $P$)
    $F^{(0)}, e^{(0)} = $ RecoverCoefficient($\mathbf{X}_{|S}$, $P$)
    $i = 0$
    while $e > \epsilon$ do
        $F^{(i+1)}, e^{(i+1)} = $ GradientSearch($\mathbf{X}_{|S}$, $F^{(i)}$, $e^{(i)}$)
        $i++$
    return $F^{(i)}, e^{(i)}$
procedure GradientSearch($\mathbf{X}_{|S}$, $F$, $e$)
    $P = \{(\omega_u, \omega_v) : (a, \omega_u, \omega_v) \in F\}$
    for $(\omega_u, \omega_v) \in P$ do
        $(\Delta u, \Delta v) = $ GetGradient($\mathbf{X}_{|S}$, $P, e, \omega_u, \omega_v$)
        $(\omega_u, \omega_v) = (\omega_u, \omega_v) + (\delta \Delta u, \delta \Delta v)$
    $F', e' = $ RecoverCoefficient($\mathbf{X}_{|S}$, $P$)
    return $F', e'$
procedure GetGradient($\mathbf{X}_{|S}$, $P, e, \omega_u, \omega_v$)
    $\Delta = \{(-1,-1), (-1,0), (-1,1), (0,-1), (0,1), (1,-1), (1,0), (1,1)\}$
    for $(du, dv) \in \Delta$ do
        $P' = P - \{(\omega_u, \omega_v)\}$
        $P' = P \bigcup \{(\omega_u + \delta du, \omega_v + \delta dv)\}$
        $F, e' = $ RecoverCoefficient($\mathbf{X}_{|S}$, $P'$)
        $\mathrm{d}e_{du,dv} = (e - e')/||(du, dv)||$
    $(du^*, dv^*) = \mathrm{argmax}_{(du,dv) \in \Delta} \mathrm{d}e_{du,dv}$
    return $(du^*, dv^*)$
procedure RecoverCoefficient($\mathbf{X}_{|S}$, $P$)
    $\mathbf{A} = 0_{|S| \times |P|}$
    $\mathbf{x}_S = 0_{|S| \times 1}$
    for $i \in \{0, \ldots, |S| - 1\}$ do
        $(u, v) = S_i$
        $\mathbf{x}_S(i) = \mathbf{X}(u, v)$
        for $k \in \{0, \ldots, |P| - 1\}$ do
            $(\omega_u, \omega_v) = P_k$
            $\mathbf{A}(i, k) = \exp\left(2j\pi \frac{u\omega_u + v\omega_v}{N}\right)$
    $\widehat{\mathbf{x}}_P = \mathbf{A}^\dagger \mathbf{x}_S$          ▷ $\mathbf{A}^\dagger$ is the pseudo-inverse of $\mathbf{A}$
    $e = ||\mathbf{x}_S - \mathbf{A}\widehat{\mathbf{x}}_P||^2$
    $F = \{(a, \omega_u, \omega_v) : a = \widehat{\mathbf{x}}_P(k), (\omega_u, \omega_v) = P_k\}_{k=0}^{|P|}$
    return $F, e$
```

ACKNOWLEDGMENTS

We thank the editors and the anonymous reviewers for their constructive comments. We are grateful to Piotr Indyk, Justin Zhang, and Andrew Adams for their help, support and insightful suggestions. We also thank the NETMIT group and MIT wireless center for the support.

REFERENCES

E. H. Adelson and J. Y. A. Wang. 1992. Single lens stereo with a plenoptic camera. *IEEE Trans. Pattern Anal. Mach. Intell.* 14, 2, 99–106.

T. E. Bishop, S. Zanetti, and P. Favaro. 2009. Light field superesolution. In *Proceedings of the IEEE International Conference on Computational Photography (ICCP'09).*

C. Buehler, M. Bosse, L. Mcmillan, S. Gortler, and M. Cohen. 2001. Unstructured lumigraph rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01).* ACM Press, New York, 425–432.

E. Candes, J. Romberg, and T. Tao. 2006a. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inf. Theory* 52, 2, 489–509.

E. Candes, J. Romberg, and T. Tao. 2006b. Stable signal recovery incomplete and inaccurate measurements. *Comm. Pure Appl. Math.* 59, 1207–1223.

A. Davis, M. Levoy, and F. Durand. 2012. Unstructured light fields. *Comput. Graph. Forum* 31, 2.1, 305–314.

D. Donoho. 2006. Compressed sensing. *IEEE Trans. Inf. Theory* 52, 4, 1289–1306.

F. Durand, N. Holzschuch, C. Soler, E. Chan, and F. X. Sillion. 2005. A frequency analysis of light transport. *ACM Trans. Graph.* 24, 3, 1115–1126.

K. Egan, Y.-T. Tseng, N. Holzschuch, F. Durand, and R. Ramamoorthi. 2009. Frequency analysis and sheared reconstruction for rendering motion blur. *ACM Trans. Graph.* 28, 3, 93:1–93:13.

T. Georgeiv, K. C. Zheng, B. Curless, D. Salesin, S. Nayar, and C. Intwala. 2006. Spatio-angular resolution tradeoff in integral photography. In *Proceedings of the Eurographics Symposium on Rendering (EGSR'06).* 263–272.

B. Ghazi, H. Hassanieh, P. Indyk, D. Katabi, E. Price, and L. Shi. 2013. Sample-optimal average-case sparse fourier transform in two dimensions. http://people.csail.mit.edu/lixin/files/ALLER13_0252_FI.pdf.

A. Gilbert, M. Muthukrishnan, and M. Strauss. 2005. Improved time bounds for near-optimal sparse fourier representations. *Proc. SPIE 5914.*

J. W. Goodman. 1996. *Introduction to Fourier Optics*. McGraw-Hill.

S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. 1996. The lumigraph. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'96).* ACM Press, New York, 43–54.

H. Hassanieh, P. Indyk, D. Katabi, and E. Price. 2012a. Nearly optimal sparse fourier transform. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC'12).* 563–578.

H. Hassanieh, P. Indyk, D. Katabi, and E. Price. 2012b. Simple and practical algorithm for sparse fft. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12).* 1183–1194.

P. Heckbert. 1989. Fundamentals of texture mapping and image warping. https://www.cs.cmu.edu/∼ph/texfund/texfund.pdf.

A. Levin and F. Durand. 2010. Linear view synthesis using a dimensionality gap light field prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10).* 1831–1838.

A. Levin, R. Fergus, F. Durand, and W. T. Freeman. 2007. Image and depth from a conventional camera with a coded aperture. *ACM Trans. Graph.* 26, 3.

A. Levin, W. T. Freeman, and F. Durand. 2008a. Understanding camera trade-offs through a bayesian analysis of light field projections. http://people.csail.mit.edu/alevin/papers/lightfields-Levin-Freeman-Durand-ECCV08.pdf.

A. Levin, P. Sand, T. S. Cho, F. Durand, and W. T. Freeman. 2008b. Motion-invariant photography. *ACM Trans. Graph.* 27, 3.

A. Levin, S. W. Hasinoff, P. Green, F. Durand, and W. T. Freeman. 2009. 4d frequency analysis of computational cameras for depth of field extension. *ACM Trans. Graph.* 28, 3.

M. Levoy and P. Hanrahan. 1996. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'96).* ACM Press, New York, 31–42.

K. Marwah, G. Wetzstein, Y. Bando, and R. Raskar. 2013. Compressive light field photography using overcomplete dictionaries and optimized projections. *ACM Trans. Graph.* 32, 4, 1–11.

D. P. Mitchell. 1991. Spectrally optimal sampling for distributed ray tracing. *ACM Comput. Graph.* 25, 4, 157–164.

R. Ng. 2005. Fourier slice photography. *ACM Trans. Graph.* 24, 735–744.

R. Ng, M. Levoy, M. Bredif, G. Duval, M. Horowitz, and P. Hanrahan. 2005. Light field photography with a hand-held plenoptic camera. https://graphics.stanford.edu/papers/lfcamera/lfcamera-150dpi.pdf.

C. Soler and F. X. Sillion. 1998. Fast calculation of soft shadow textures using convolution. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'98).* 321–332.

Stanford. 2008. Stanford light field archive. http://lightfield.stanford.edu/.

A. Veeraraghavan, R. Raskar, A. Agrawal, A. Mohan, and J. Tumblin. 2007a. Dappled photography: Mask enhanced cameras for heterodyned light fields and coded aperture refocusing. *ACM Trans. Graph.* 26, 3, 69.

B. Wilburn, N. Joshi, V. Vaish, E.-V. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, and M. Levoy. 2005. High performance imaging using large camera arrays. *ACM Trans. Graph.* 24, 3, 765–776.

Z. Zhang and M. Levoy. 2009. Wigner distributions and how they relate to the light field. In *Proceedings of the IEEE International Conference on Computational Photography (ICCP'09).* 1–10.

Received February 2014; accepted May 2014