Supplementary Material:
Compressive Light Field Photography using Overcomplete Dictionaries and Optimized Projections

Kshitij Marwah[1]   Gordon Wetzstein[1]   Yosuke Bando[2,1]   Ramesh Raskar[1]
[1]MIT Media Lab   [2]Toshiba Corporation

This document provides additional discussion and results in support of the primary text. Our work builds on recent advances in the signal processing community; we outline and evaluate relevant mathematical tools in supplementary section A. One of the key challenges in compressive light field photography is the choice of a dictionary in which natural light fields are sparse. In supplementary section B, we discuss approaches to learn light field atoms—essential building blocks of natural light fields that sparsely represent such high-dimensional signals. In supplementary sections C and D, we discuss additional details on hardware and software implementation, including calibration procedures and training sets used for all experiments in the primary texts. Finally, we discuss more details of the coupled dictionary learning method used for "undappeling" coded 2D projections (sec. 7.1) in supplementary section E.

A A Primer on Sparse Coding

This section reviews the mathematical background of sparse coding. Much progress on this topic has been made in the signal and information theory community throughout the last decade. This document serves merely as a concise overview of mathematical tools relevant for compressive light field photography. We refer the interested reader to relevant literature whenever appropriate. Sparse reconstruction approaches are evaluated based on the following criteria: speed, quality, and ease of implementation or available code.

Let us start with the problem statement. Given a vectorized sensor image $\mathbf{i} \in \mathbb{R}^m$ that contains a coded projection of the incident vectorized light field $\mathbf{l} \in \mathbb{R}^n$, we wish to recover a set of sparse coefficients $\boldsymbol{\alpha} \in \mathbb{R}^d$ that form an accurate representation of the light field in some dictionary $\boldsymbol{\mathcal{D}} \in \mathbb{R}^{n \times d}$

$$\mathbf{i} = \mathbf{\Phi} \mathbf{l} = \mathbf{\Phi} \boldsymbol{\mathcal{D}} \boldsymbol{\alpha}. \tag{S.1}$$

The challenges here are twofold. First, the number of measurements (sensor pixels) m is significantly lower than the number of unknowns (light field rays) n. Second, assuming that tools from the signal processing community can be employed to solve this problem, which basis or dictionary $\boldsymbol{\mathcal{D}}$ provides a sufficiently sparse representation for natural light fields? While supplementary section B is mainly concerned with answering the latter question, we outline tools that are at our disposal to tackle the first challenge in this section.

Solving an underdetermined linear system, such as Equation S.1, is challenging because it has infinitely many solutions. Our work builds on recent advances in compressed sensing (see, e.g. [Donoho 2006a; Candès and Wakin 2008]) to solve such equation systems. The general idea is that, under certain conditions, underdetermined systems can be solved if the unknowns are either sparse or can be represented in a sparse basis or overcomplete dictionary. A signal is said to be k-sparse, if it has at most k nonzero coefficients. Mathematically, sparsity is expressed by the ℓ_0 pseudo-norm $\|\cdot\|_0$ that simply counts the number of nonzero elements in a vector.

The answer to the following problem is at the core of finding a robust solution to Equation S.1:

$$\begin{aligned} \underset{\{\boldsymbol{\alpha}\}}{\text{minimize}} \quad & \|\boldsymbol{\alpha}\|_0 \\ \text{subject to} \quad & \|\mathbf{i} - \mathbf{\Phi} \boldsymbol{\mathcal{D}} \boldsymbol{\alpha}\|_2 \leq \epsilon \end{aligned} \tag{S.2}$$

This formulation seeks the sparsest representation of a signal that achieves a prescribed approximation error. Equation S.2 can similarly be stated with equality constrains; we focus on bounded errors as the measurements taken by computational cameras usually contain sensor noise. A different, but closely related problem is that of sparsity-constrained approximation:

$$\begin{array}{ll} \underset{\{\boldsymbol{\alpha}\}}{\text{minimize}} & \|\mathbf{i} - \boldsymbol{\Phi}\boldsymbol{\mathcal{D}}\boldsymbol{\alpha}\|_2 \\ \text{subject to} & \|\boldsymbol{\alpha}\|_0 \leq \kappa \end{array} \tag{S.3}$$

Here, the objective is to find a signal that has at most κ nonzero coefficients and minimizes the residual. Problems S.2 and S.3 can be expressed using a Lagrangian function that balances the twin objectives of minimizing both error and sparsity as

$$\underset{\{\boldsymbol{\alpha}\}}{\text{minimize}} \; \|\mathbf{i} - \boldsymbol{\Phi}\boldsymbol{\mathcal{D}}\boldsymbol{\alpha}\|_2 + \lambda \|\boldsymbol{\alpha}\|_0 . \tag{S.4}$$

The above problems (Eqs. S.2– S.4) are combinatorial in nature. Finding optimal solutions is NP-hard and, therefore, intractable for high resolutions [Natarajan 1995]. The two most common approaches to tackle these problems are greedy methods and convex relaxation methods (see, e.g. [Tropp and Wright 2010; Tropp 2004]). The remainder of this section outlines methods for both.

Please note that Equations S.2– S.4 and supplementary section A.1 model the ℓ_0 and ℓ_1 synthesis problem. We discuss the analysis problem in supplementary section A.2 and show comparisons of the two formulations in Figure S.1.

A.1 Approximate Solutions

In this section, we briefly review the two most common approximate solutions to the problems outlined in the previous section: greedy algorithms and convex relaxation methods.

A.1.1 Greedy Methods

Greedy methods for sparse approximation are simple and fast. In these iterative approaches, the dictionary atom that is most strongly correlated to the residual part of the signal is chosen and the corresponding coefficient added to the sparse coefficient vector (MP). In addition, a least-squares minimization can be added to each iteration so as to significantly improve convergence (OMP).

Matching Pursuit (MP) is a greedy method for sparse approximation that constructs a sparse approximant one step at a time by selecting the atom most strongly correlated with the residual part of the signal and uses it to update the current approximation. One way to accomplish this is to choose the atom most strongly correlated with the residual by computing inner products between residual and all atoms. Then, the coefficient vector is updated for the coefficient corresponding to the chosen atom with the inner product of that atom and the residual. This is done iteratively. For a more detailed discussion of MP and its variants, the reader is referred to [Tropp 2004].

Orthogonal Matching Pursuit (OMP) has been used for decades and is one of the earliest approaches to sparse approximation. While independently discovered by several researchers, a detailed treatise can be found in [Tropp and Gilbert 2007].

Just as MP, OMP is a greedy algorithm and therefore extremely fast. The improvement over MP comes from the fact that OMP adds a additional least-squares minimization in each iteration. Similarly to MP, in each iteration OMP picks the atom that contributes most to the overall residual and adds the corresponding coefficient to the sparse representation. In addition to picking that coefficient, OMP runs a least-squares minimization over all coefficients picked until the current iteration to obtain best approximation over the atoms that have already been chosen.

One of the disadvantages of OMP is that exact recovery of the signal is only guaranteed for a very low coherence value and that the sparsity of the signal has to be known. The latter is rarely the case in practice, however.

A.1.2 Convex Relaxation Methods

Convex relaxation methods follow a different strategy than greedy methods. Rather than trying to solve a difficult problem, these types of methods solve a slightly different problem that can be solved much easier in hope that the solutions will be close enough to those of the original problems. In particular, the ℓ_0-norm that makes Equations S.2–S.4 NP-hard is replaced by the ℓ_1-norm, which is convex yet non-smooth. This approach replaces a combinatorial sparse approximation problem with a related convex problem, which can be solved in polynomial time. Although there are no theoretical guarantees that these numerical methods actually solve sparse approximation problems, it has been shown that, under certain conditions, the solutions to the convex relaxation problems converge to the combinatorial problems with a very high probability [Candès et al. 2006; Donoho 2006b].

One of the most important achievements of recent literature on compressive sensing is the derivation of bounds on recoverability and required number of measurements for the convex relaxation methods described in this section. A lower bound can, for instance, be placed on the number of measurements m for a k-sparse d-dimensional signal [Candès and Wakin 2008]:

$$m \geq \text{const}\ k\ \log\left(\frac{d}{k}\right). \tag{S.5}$$

Solutions to the convexified problems can be found with a variety of approaches, including linear programming or nonlinear programming, such as interior point methods [Boyd and Vandenberghe 2004]. In the following, we discuss several different formulations for convex relaxations of Equations S.2– S.4.

Basis Pursuit (BP) replaces the ℓ_0-norm of Equation S.2 with an ℓ_1-norm, but also uses equality constraints for the measurements [Chen et al. 1998]:

$$\begin{array}{ll} \underset{\{\boldsymbol{\alpha}\}}{\text{minimize}} & \|\boldsymbol{\alpha}\|_1 \\ \text{subject to} & \boldsymbol{\Phi}\boldsymbol{\mathcal{D}}\boldsymbol{\alpha} = \mathbf{i} \end{array} \tag{S.6}$$

Efficient implementations of this problem can, for instance, be found in the SPGL1 solver package [van den Berg and Friedlander 2008]. Basis pursuit works well so long as the number of measurements $m \geq \text{const}\, k \log{(d/k)}$ and the measurement matrix is sufficiently incoherent. While this problem is important for many applications, in the context of light field cameras one usually has to deal with sensor noise which makes it difficult to use the equality constraints in the above formulation.

Basis Pursuit Denoise (BPDN) is very similar to basis pursuit; the main difference is that the equality constraints are replaced by inequality constraints [Chen et al. 1998]:

$$\begin{array}{ll} \underset{\{\boldsymbol{\alpha}\}}{\text{minimize}} & \|\boldsymbol{\alpha}\|_1 \\ \text{subject to} & \|\mathbf{i} - \boldsymbol{\Phi}\boldsymbol{\mathcal{D}}\boldsymbol{\alpha}\|_2 \leq \epsilon \end{array} \tag{S.7}$$

The parameter ϵ is basically the noise level of a recorded sensor image in our application. Both, BP and BPND can be summarized as minimizing the sparsity of a coefficient vector while complying with the observations, at least up to some threshold ϵ. For practical implementation, the constraints of Equation S.7 can be directly included in the objective function using a Lagrangian formulation

$$\underset{\{\boldsymbol{\alpha}\}}{\text{minimize}} \ \|\mathbf{i} - \boldsymbol{\Phi}\boldsymbol{\mathcal{D}}\boldsymbol{\alpha}\|_2 + \lambda \|\boldsymbol{\alpha}\|_1 \tag{S.8}$$

Equation S.8 is an unconstrained, convex, quadratic problem and can, therefore, be easily solved with existing solvers. Efficient implementations are readily available in the SPGL1 package [van den Berg and Friedlander 2008] and also in the NESTA solver [Becker et al. 2009]. Yang et al. [Yang et al. 2010] provide an excellent overview of recent and especially *fast* ℓ_1 minimization algorithms; upon request the authors also provide source code—we found their implementation of a homotopy method to solve the BPDN problem most efficient (see Fig. S.1).

Lasso is a problem closely related to BPDN and a convex relaxation of Equation S.4:

$$\begin{aligned} \underset{\{\boldsymbol{\alpha}\}}{\text{minimize}} \quad & \|\mathbf{i} - \boldsymbol{\Phi}\boldsymbol{\mathcal{D}}\boldsymbol{\alpha}\|_2 \\ \text{subject to} \quad & \|\boldsymbol{\alpha}\|_1 \leq \kappa \end{aligned} \tag{S.9}$$

Solving this problem provides the best approximation of a signal using a linear combination of (ideally) κ atoms or fewer from the dictionary [Tibshirani 1996]. Lasso can also be solved with the unconstrained, Lagrangian form of the BPDN problem (Eq. S.8); the parameter λ is simply chosen to give a different tradeoff between sparsity and error. An implementation of Lasso can be found in the SPGL1 package [van den Berg and Friedlander 2008].

A.2 The Analysis Problem

The last section gave a brief overview of approaches to sparse coding and approximate solutions. All problems are formulated as synthesis problems, however, which are theoretically only valid if the dictionary $\boldsymbol{\mathcal{D}}$ is an orthonormal basis. In case it is coherent, overcomplete, or redundant, Candès et al. [2011] have shown that solving the following ℓ_1-analysis problems theoretically results in superior reconstructions:

$$\begin{aligned} \underset{\{\mathbf{l}\}}{\text{minimize}} \quad & \|\boldsymbol{\mathcal{D}}^* \, \mathbf{l}\|_1 \\ \text{subject to} \quad & \|\mathbf{i} - \boldsymbol{\Phi}\mathbf{l}\|_2 \leq \epsilon \end{aligned} \tag{S.10}$$

[Candès et al. 2011] use the reweighted ℓ_1 method [Candès et al. 2008] to solve the Basis Pursuit Denoise analysis problem. Reweighted ℓ_1 is an iterative method; an efficient implementation is readily available in the NESTA solver package [Becker et al. 2009]. We compare NESTA's reweighted ℓ_1 solver with the BPND implementation of SPGL1 in Figure S.1 and conclude that both result in a comparable reconstruction quality for our problem.

A.3 Sliding Window Reconstructions

At this stage we would like to point out that reconstructions are performed independently for each pixel in the sensor image. That is, a window of patch size $p_x \times p_x$, $p_x^2 = m$ is centered around a sensor pixel and represents the measurement vector $\mathbf{i} \in \mathbb{R}^m$ (see Eq. S.1). A sparse set of 4D light field atoms, each of size $p_x \times p_x \times p_\nu \times p_\nu$, $p_x^2 p_\nu^2 = n$, is then reconstructed for each sensor pixel.

Following standard practice [Elad and Aharon 2006], we use this sliding window approach and reconstruct the 4D window for each sensor pixel separately in all results in the paper. As a post-processing step, overlapping patches are merged using a median function. We compare several different choices for the merging function, including average, median, and simply picking the spatial center of the 4D window in Figure S.1.

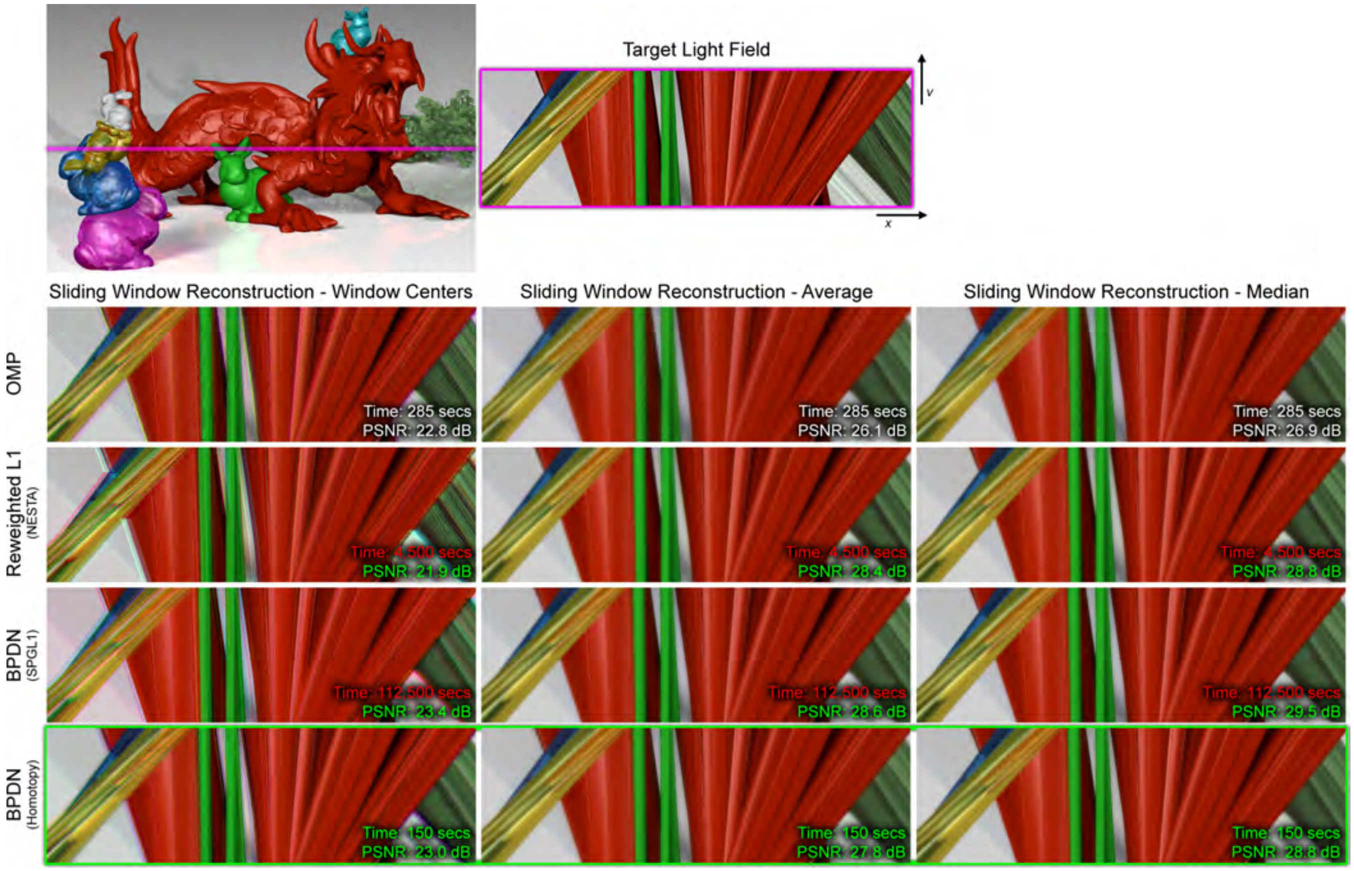

Figure S.1: *Evaluating sparse reconstruction methods. We reconstruct a 2D light field (top row) using the same dictionary and simulated sensor image with four different methods of sparse coding (rows 2–6) and different window merging functions. Although fast, orthogonal matching pursuit (OMP, second row) only achieves a low reconstruction quality. Reweighted ℓ_1, as implemented in the NESTA package, yields high-quality reconstructions but takes about 15 times as long as OMP (third row). Basis pursuit denoise, as implemented in the SPGL1 solver package, is also high-quality but almost 400 times slower than OMP (fourth row). Due to its immense time requirements, this implementation is not suitable for high-resolution 4D light field reconstructions. The homotopy approach to solving the basis pursuit denoise problem (fifth row) is even faster than OMP and results in a reconstruction quality that is comparable with the best, but much slower SPGL1 implementation.*

A.4 Evaluating Sparse Reconstruction Approaches

A comparison of several of the above discussed sparse coding algorithms is shown in Figure S.1. We evaluate these approaches based on reconstruction quality and speed. The choice of algorithm is based on publicly available code and ease of implementation; while there is a huge variety of different algorithms in the literature, we pick the most popular ones that can easily be implemented or are readily available. Figure S.1 shows that reweighted ℓ_1, as implemented in NESTA [Becker et al. 2009], basis pursuit denoise implemented in SPGL1 [van den Berg and Friedlander 2008], and basis pursuit denoise using the homotopy method [Yang et al. 2010] perform equally well in terms of reconstruction quality. However, the processing time for both reweighted ℓ_1 and BPDN (SPGL1) prohibits practical use for high-resolution 4D light field reconstructions. Hence, we chose the homotopy method described by Yang et al. [2010] with source code provided by the authors.

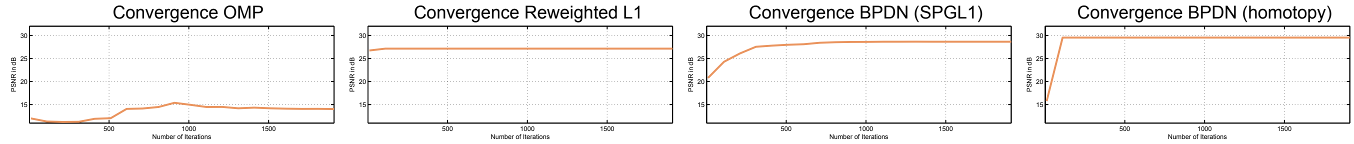

Figure S.2: *Evaluating convergence for all algorithms using a single patch. Please note that OMP is a greedy algorithm; while all other algorithms minimize the ℓ_1-norm of coefficients, OMP increments the number of used coefficients by one in each iteration.*

B Overview and Evaluation of Dictionary Learning Methods

One of the most critical parts of any sparse coding approach is the choice of basis or dictionary that sparsifies the signal appropriately. As shown in the primary text, natural light fields are poorly sparsified by standard bases such as the discrete cosine transform, the Fourier basis, or wavelets. Instead, we propose to learn the fundamental building blocks of natural light fields—light field atoms—in overcomplete, redundant, and possibly coherent dictionaries. Within the last few years, sparse coding with coherent and redundant dictionaries has gained a lot of interest in the signal processing community [Candès et al. 2011].

In this section, we review and evaluate approaches to learning dictionaries containing light field atoms from large training sets of natural light fields. The most significant criteria for evaluation are speed and memory footprint, quality of reconstruction, level of sparsity/compressibility, ease of implementation or available code, and interpretability. In this section, we evaluate several learning methods in flatland with straightforward extensions to the full, four-dimensional case. The flatland analysis simply allows for more intuitive interpretations.

B.1 Learning Light Field Atoms

For a comprehensive overview of dictionary learning methods and design criteria, the interested reader is referred to Michal Aharon's PhD thesis [Aharon 2006]. The general problem of learning an overcomplete dictionary $\mathcal{D}$ that sparsifies a class of signals can be expressed as

$$\begin{aligned} \underset{\{\mathcal{D},\mathcal{A}\}}{\text{minimize}} \quad & \|\mathbf{L} - \mathcal{D}\mathcal{A}\|_F \\ \text{subject to} \quad & \forall i, \|\mathcal{A}_i\|_0 \leq k \end{aligned} \tag{S.11}$$

Here, $\mathcal{D} \in \mathbb{R}^{n \times d}$ is the overcomplete dictionary, $\mathbf{L} \in \mathbb{R}^{n \times q}$ is the training set that contains q signals or patches that represent the desired class of signals well. An algorithm tackling Equation S.11 will not only compute $\mathcal{D}$ but also the sparse coefficients $\mathcal{A} \in \mathbb{R}^{d \times q}$ that approximate the training set. The Frobenius matrix norm in the above problem is defined as $\|\mathbf{X}\|_F = \sqrt{\sum_{ij} X_{ij}^2}$.

A variety of solutions to Equation S.11 exist, in the following we evaluate the most widely used methods.

B.1.1 K-SVD

The K-SVD algorithm [Aharon et al. 2006] is a simple, yet powerful method of learning overcomplete dictionaries from a training dataset. K-SVD applies a two-stage process to solve Equation S.11: given an initial estimate of $\mathcal{D}$ and $\mathcal{A}$, in the first stage (sparse coding stage) the coefficient vectors $\mathcal{A}_i$, $i = 1 \ldots q$ are updated independently using any pursuit method with the fixed dictionary. In the second stage (codebook update stage), $\mathcal{D}$ is updated by picking the singular vector of the residual matrix $\mathbf{E} = \mathbf{L} - \mathcal{D}\mathcal{A}$ that contributes most to the error; the corresponding coefficients in $\mathcal{A}$ and $\mathcal{D}$ are updated so as to minimize the residual with that singular vector. The vector can easily be found by applying a singular value decomposition (SVD) to the residual matrix $\mathbf{E}$ and picking the singular vector corresponding to the largest singular value. The K-SVD algorithm alternates between these two stages, sparse coding and codebook update, in an iterative fashion and is therefore similar in spirit to alternating least-squares methods.

B.1.2 Nonnegative K-SVD

The nonnegative variant of K-SVD [Aharon et al. 2005] (K-SVD NN) allows only positive values in the dictionary atoms as well as in the coefficients, i.e. $\boldsymbol{\mathcal{D}}_{ij} \geq 0$ and $\alpha_j \geq 0$. The K-SVD NN algorithm itself is a slight modification of K-SVD: in the sparse coding stage, any pursuit algorithm can be applied that only allows for positive coefficients; the SVD in the codebook update stage is replaced by finding the closest nonnegative rank-1 matrix (in Frobenius norm) that approximates $\mathbf{E}$. Algorithms for nonnegative matrix factorization (NMF) are well-explored (e.g., [Lee and Seung 1999]).

The advantage of nonnegative matrix factorizations and dictionary learning algorithms is that these often result in decompositions that carry physical meaning. In particular, a visual analysis of the extracted or decomposed features often allows for intuitive interpretations of its parts. For the application of learning light field atoms, nonnegative dictionaries allow for intuitive interpretations of what the basic building blocks of natural light fields are. Figure S.3 compares the atoms learned with K-SVD NN to alternative learning methods that allow negative atoms.

B.1.3 Online Sparse Coding

With online sparse coding, we refer to the online optimization algorithm proposed by Mairal et al. [2010]. The goal of this algorithm is to overcome the limited size of training datasets that can be handled by alternative dictionary learning methods, such as K-SVD. For this purpose, Marail et al. propose a method based on stochastic approximation that is specifically designed to handle large training sets of visual data consisting of small patches in an efficient manner. The proposed approach is implemented in the open source software package SPAMS (http://spams-devel.gforge.inria.fr).

Assume that a large training set of small patches is available. Conventional algorithms, such as K-SVD, randomly pick the maximum number of training patches that can be processed given a limited memory or time budget. Online sparse coding uses a batch approach, where the full training set is processed by picking a random patch at a time and updating the dictionary and coefficients accordingly. This is similar in spirit to matching pursuit algorithms.

While being able to handle very large training sets and being very fast, online sparse coding has the disadvantage of very slow converge rates. Fixing the number of iterations and comparing it to K-SVD, as illustrated in Figure S.3, shows that the resulting light field atoms are much noisier and lower quality. This is mainly due to the fact that it is unclear how to chose the next training patch at any given iteration of the algorithm. Ideally, one that would maximize the amount of new information should be chosen; it may be impossible to know which one that is, so a random sample is drawn that may not lead to any new information at all.

B.2 Generating "Good" Training Sets with Coresets

Coresets are means to preprocess large training sets so as to make subsequent dictionary learning methods more efficient. A comprehensive survey of coresets can be found in [Agarwal et al. 2004]. Given a training set $\mathbf{L} \in \mathbb{R}^{n \times q}$, a coreset $\boldsymbol{\mathcal{C}} \in \mathbb{R}^{n \times c}$ is extracted and directly used as a surrogate training set for the dictionary learning process (Eq. S.11). Coresets have two advantages: first, the size of the training set is significantly reduced (i.e., $c \ll q$) and, second, redundancies in the training set are removed, significantly improving convergence rates of batch-sequential algorithms such as online sparse coding. In a way, coresets can be interpreted as clustering algorithms specialized for the application of picking "good" training sets for dictionary learning [Feldman 2010].

A simple yet efficient approach to computing corsets is to select c patches of the training set that have a sufficiently high variance [Feigin et al. 2012]. Feigin et al. showed that the best dictionary learned from a coreset computed

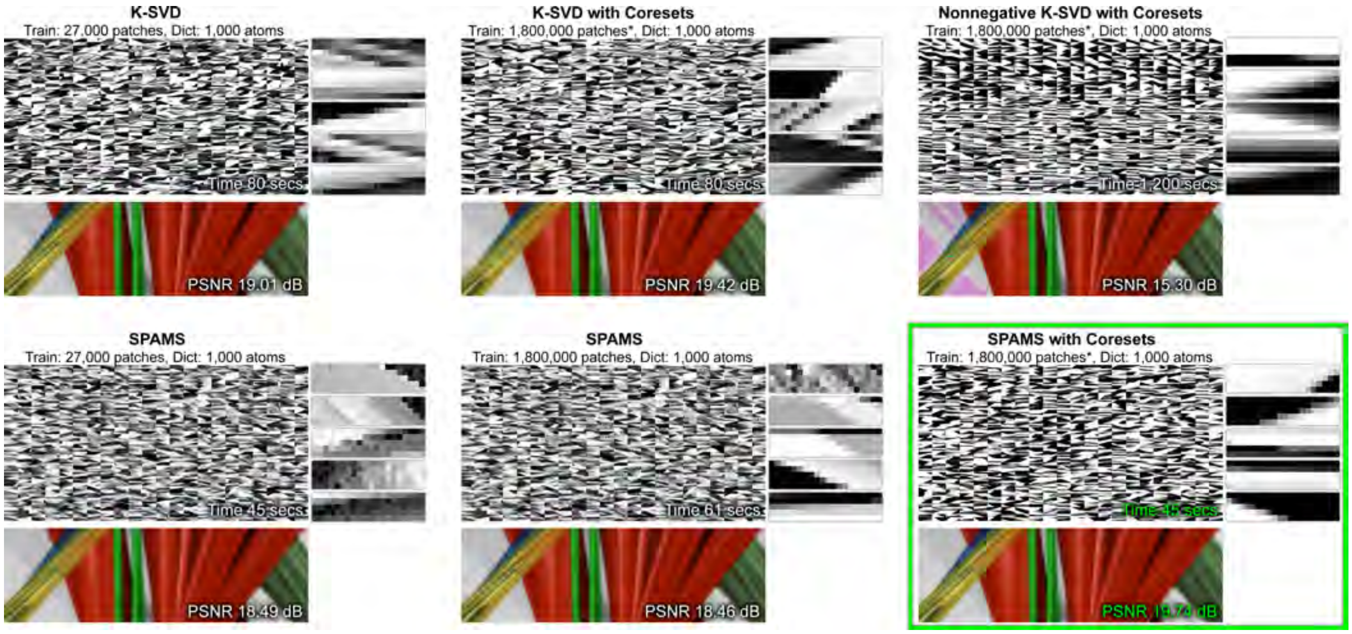
K-SVD
Train: 27,000 patches, Dict: 1,000 atoms
Time 80 secs
PSNR 19.01 dB
K-SVD with Coresets
Train: 1,800,000 patches*, Dict: 1,000 atoms
Time 80 secs
PSNR 19.42 dB
Nonnegative K-SVD with Coresets
Train: 1,800,000 patches*, Dict: 1,000 atoms
Time 1,200 secs
PSNR 15.30 dB
SPAMS
Train: 27,000 patches, Dict: 1,000 atoms
Time 45 secs
PSNR 18.49 dB
SPAMS
Train: 1,800,000 patches, Dict: 1,000 atoms
Time 61 secs
PSNR 18.46 dB
SPAMS with Coresets
Train: 1,800,000 patches*, Dict: 1,000 atoms
Time 45 secs
PSNR 19.74 dB

Figure S.3: *Evaluating various approaches to dictionary learning and choosing training sets. The two most important evaluation criteria are speed (and memory footprint) of the learning stage as well as quality of the learned light field atoms. The former is indicated by timings, whereas the latter is indicated by the PSNR and visual quality of reconstructions from the same measurements (2D light field projected onto a 1D sensor image with a random modulation mask). We evaluate the K-SVD algorithm and its nonnegative variant and online sparse coding as implemented in the SPAMS software package. K-SVD is a very slow algorithm that limits practical application to small training sets containing 27,000 training patches (top left); coresets are tested to reduce a much larger training set with 1,800,000 patches to the same, smaller size (27,000 patches, top center). In all cases, K-SVD learns high-quality light field atoms that capture different spatial features in the training set sheared with different amounts in the spatio-angular domain. Unfortunately, K-SVD becomes too slow for practical application with large high-dimensional training sets; nevertheless, we use it as a benchmark in these flatland experiments. Online sparse coding (bottom row) is much faster and capable of handling large training sets. Unfortunately, the learned atoms are very noisy and lead to low-quality reconstructions (bottom left and center). Using coresets, much of the redundancy in large training sets can be removed prior to learning the atoms (lower right), thereby removing noise in the atoms and achieving the highest-quality reconstructions in the least amount of time.*

with their algorithm is very close to the best dictionary that can be learned from the original training dataset that the coreset was learned from. We implement the method described in [Feigin et al. 2012] and evaluate it with K-SVD and online sparse coding in the following section.

B.3 Evaluating Dictionary Design Parameters

In this section, we evaluate various approaches to and design parameters of the learning stage in addition to those discussed in the primary text. We conclude that coresets applied to large training sets in combination with online sparse coding implemented in the SPAMS package gives the best results in the shortest amount of time.

Speed and Quality of Learning Stage We evaluate several different dictionary learning methods in Figure S.3. While the K-SVD algorithm results in high-quality light field atoms, online sparse coding usually extracts noisy atoms for a comparable number of iterations in both algorithms. Unfortunately, K-SVD becomes increasingly slow and unfeasible for large training sets of high-resolution and four-dimensional light field patches. Applying coresets

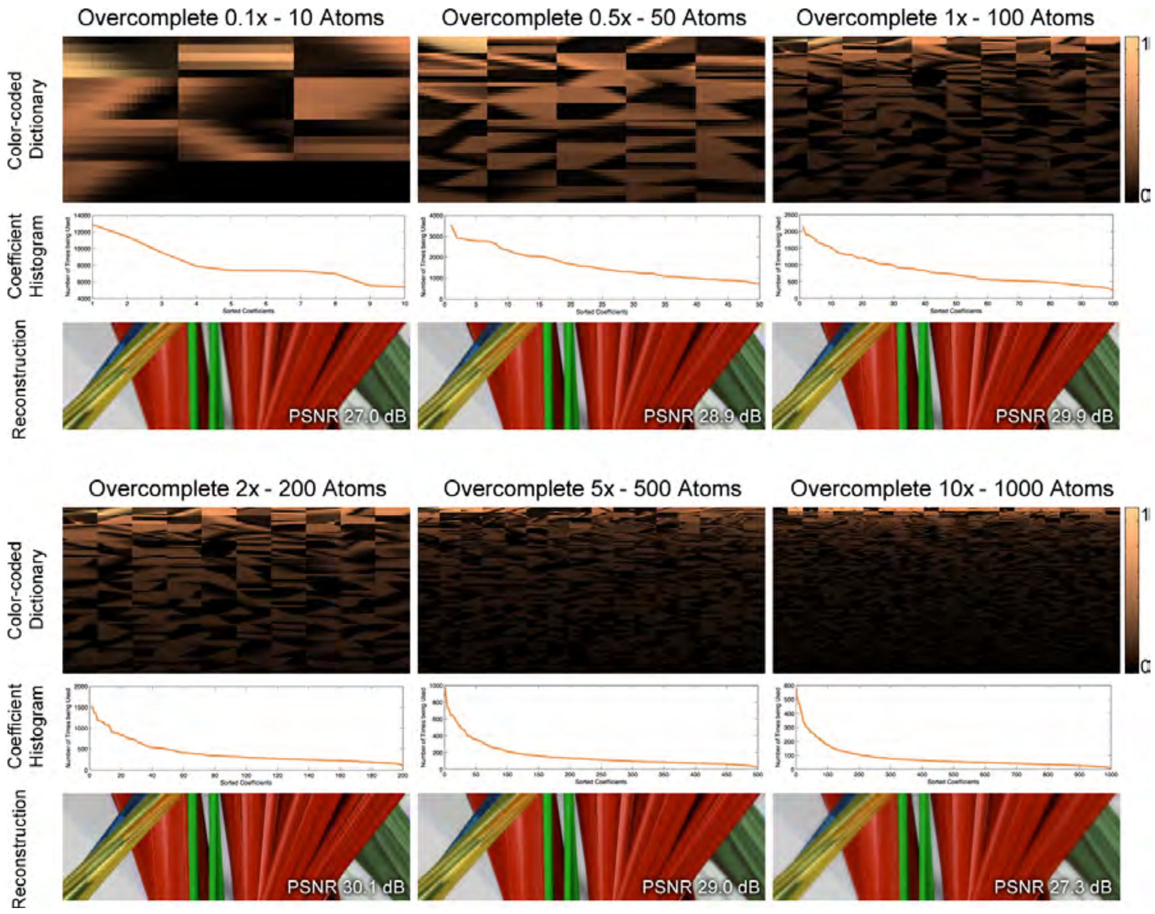

Figure S.4: *Evaluating dictionary overcompleteness. The color-coded visualizations of dictionaries (top row) and the histograms (center row) illustrate the intrinsic dimension of these dictionaries. For more than* $2\times$ *overcomplete dictionaries, most of the atoms are rarely used to adequately represent the training set. Reconstructions of a 2D light field that was not in the training set using the respective dictionaries (bottom row) show that the quality (PSNR) is best for* $1{-}2\times$ *overcomplete dictionaries and drops below and above. Dictionaries with* $0.1{-}0.5\times$ *overcompleteness do not perform well, because they simply do not contain enough atoms to sparsely represent the test light field. On the other hand, excessive overcompleteness (larger* d*) does not improve sparsity (smaller* k*) further, and* $k\log(d/k)$ *turns to increase, leaving the number of measurements insufficient.*

to large training sets prior to the learning stage, however, removes redundancies in the training data and allows SPAMS to learn atoms that result in higher-quality reconstructions than K-SVD in very little time (Fig. S.3, lower right).

We also evaluate the nonnegative variant of K-SVD for the purpose of improved interpretability of the learned atoms (Fig. S.3, upper right). As with all other approaches, nonnegative K-SVD atoms exhibit spatial features sheared with different amounts in the spatio-angular space. Edges in these atoms, however, are much smoother than in the approaches that allow for negative entries. This allows for improved blending of several nonnegative atoms to form light field “molecules”, such as T-junctions observed in occlusions. Atoms that allow for negative values can easily create such “molecules” when added. The reconstruction quality of nonnegative K-SVD is low, however; we conclude that this algorithm is great to analyze learned atoms but, considering low-quality and immense compute times, not fit for practical processing in this application.

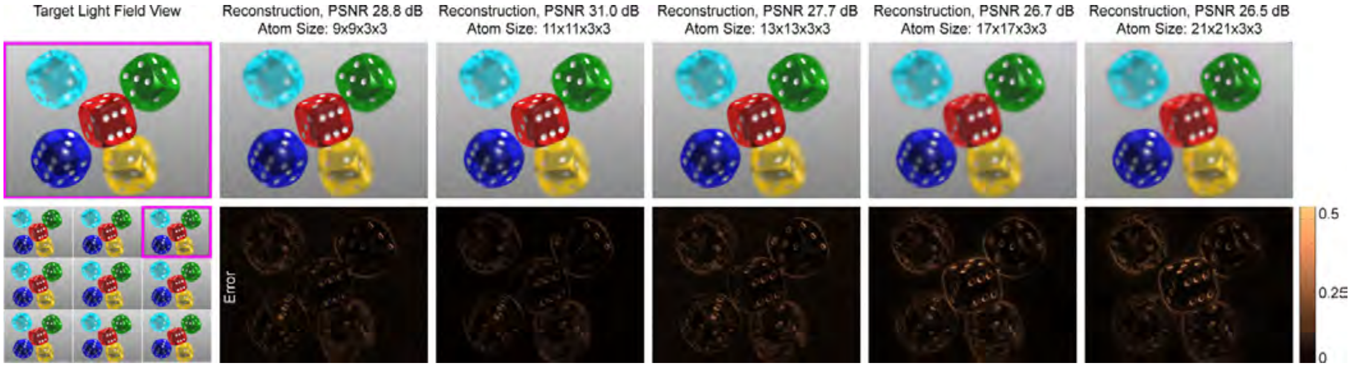

Figure S.5: *Evaluating light field atom sizes. A synthetic light field (lower left) is projected onto a sensor image with a random modulation mask and reconstructed with dictionaries comprised of varying atom sizes. The angular resolution of light field and atoms is 3×3, but the spatial resolution of the atoms ranges from 9×9 to 21×21. Dictionaries for all experiments are $5\times$ overcomplete. We observe best reconstruction quality for an atom size of 11×11. If chosen too small, the spatio-angular shears of objects at a distance to the focal plane will not be adequately captured by the light field atoms. Furthermore, the ratio between the number of measurements and number of unknowns is too low for a robust reconstruction with sparse coding approaches. For increasing atom resolutions, that ratio becomes more favorable for reconstruction—however, with an increasing spatial extend of an atom it also becomes increasingly difficult to represent the underlying light fields sparsely, leading to lower-quality reconstructions.*

Atom Size The size (or resolution) of light field atoms is an important design parameter. Consider an atom size of $n = p_x^2 \times p_\nu^2$—the number of measurements is always $m = p_x^2$. Assuming a constant sparseness k of the light field in the coefficient space, the minimum number of measurements should ideally follow Equation S.5, i.e. $m \geq \text{const}\, k \, \log{(d/k)}$. As the spatial atom size is increased for a given angular size, the recovery problem becomes more well-posed because m grows linearly with the atom size, whereas the right hand side only grows logarithmically. On the other hand, an increasing spatial light field atom size may decrease the compressibility of the light field expressed in terms of these atoms. Figure S.5 evaluates the sensitivity of the light field recovery process with respect to the spatial atom size. We conclude that there is an optimal tradeoff between the above arguments (number of measurements vs. sparsity). We heuristically determine the optimal atom size to be $p_x = 11$ for our application.

Overcompleteness We also evaluate how overcomplete dictionaries should ideally be, that is how many atoms should be learned from a given training set. Conventional orthonormal bases in this unit are "$1\times$" overcomplete—$\boldsymbol{\mathcal{D}}$ is square. The overcompleteness of dictionaries, however can be arbitrarily chosen in the learning process. We evaluate dictionaries that are $0.1\times, 0.5\times, 1\times, 2\times, 5\times, 10\times$ overcomplete in Figure S.4. The color-coded visualizations of the atoms in the respective dictionaries indicate how many times each of the atoms is actually being used for the training set (on a normalized scale). The histograms count how many times an atom was used to represent the training set is shown in the center row. We observe that for a growing dictionary size, the redundancy grows as well. While all coefficients in the $0.1\times$ and $0.5\times$ dictionaries are being used almost equally often, for $5\times$ and particularly $10\times$, most of the coefficients are rarely being used, hence overly redundant. We conclude that $1-2\times$ overcomplete dictionaries adequately represent this particular training set consisting of $27,000$ light field patches (reduced from 1,800,000 randomly chosen ones using coresets); all atoms have a resolution of 5×20 in angle and space.

At this point, we would like to remind the reader of the number of required measurements in convex relaxation methods, such as basis pursuit denoise used in this experiment, as outlined in supplementary Equation S.5. For a fixed sparsity, a linearly increasing dictionary size (or overcompleteness) requires an logarithmically growing number of measurements. As seen in the top and center rows of Figure S.4, the intrinsic dimension of these dictionaries, that

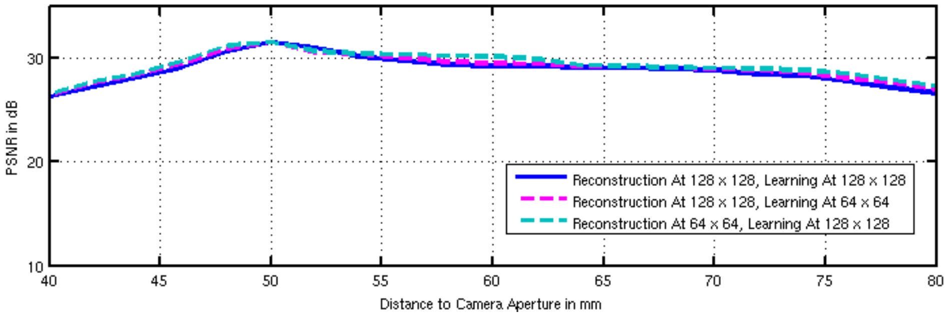

Figure S.6: *Evaluating dictionary learning at various resolutions. We learn two dictionaries from a training set consisting of black and white text at resolutions of* 128×128 *and* 64×64 *at different depths from a simulated camera aperture of 0.25 cm. Reconstructions of a resolution chart placed at the same depths but with different resolutions are then performed. As shown in the plot, the PSNRs do not vary significantly for dictionaries learned at* 64×64 *and reconstruction at* 128×128*, and vice versa. This can be attributed to our patch-by-patch reconstruction that captures features at a resolution of* 9×9 *in this case. At such a patch size, features are generally scale-invariant. Moreover, for the case of resolution charts, these features are generally edges (spatial or angular) that do not depend on resolutions.*

is the number of required coefficients to adequately represent the training set, is about $1 - 2\times$. As the overcompleteness grows ($5 - 10\times$), there are simply not enough measurements in the sensor image, hence the PSNR of the reconstructions drops (Fig. S.5). While this is only a heuristic experiment, we use it as an approximation for the optimal tradeoff in the primary text and the supplemental document.

Effect of Resolution We evaluate the effect of different resolutions between the learning and reconstruction phases. We learn two dictionaries from a plane of text placed at different depths at resolutions of 128×128 and 64×64, respectively. Given the 128×128 dictionary for benchmarking, we reconstruct a resolution chart at the same depths at the same resolution. To compare, we reconstruct an anti-aliased resolution chart downsampled to 64×64 with the dictionary learned at the resolution of 128×128. Similarly, we also reconstruct a resolution chart of 128×128 with the dictionary learned at the resolution of 64×64.

For this experiment, no significant difference was found in PSNR as shown in the plot in Figure S.6. Though the dictionary learning is not proven to be scale or rotation-invariant, it aims to capture natural building blocks of light fields. In the case of a textual planar scene at different depths, these blocks essentially contain spatial and angular edges that are scale-invariant. As shown in [Mairal et al. 2007], since dictionary learning is performed on patches at low resolutions 11×11 or 9×9 in our case, effects of multi-scale representations are less profound. A further evaluation on diverse set of light fields and patch sizes is needed to evaluate effects of scale on both learning and reconstruction. As computational resources increase and dictionaries can be efficiently learned at much higher patch resolutions, better insights may be drawn. As of now, this is outside of the scope of this paper and is left for future work.

Optimal Mask Patterns Traditionally in compressive sensing systems, random matrices have been used to project the high dimensional sparse signal to a low dimensional subspace. This is because random matrices are known to satisfy the well-known Restricted Isometry Property (RIP), that intuitively signifies preservation of energy for a sparse vector when projected down to a lower dimensional subspace. Recently, it has been shown that, for overcomplete dictionaries, there exists a class of matrices that can be optimized for a given basis [Duarte-Carvajalino and Sapiro 2009]. Furthermore, dictionaries and sensing matrices can be optimized together to be most orthonormal when operating on sparse vectors. Mathematically, this optimality criterion can be expressed as

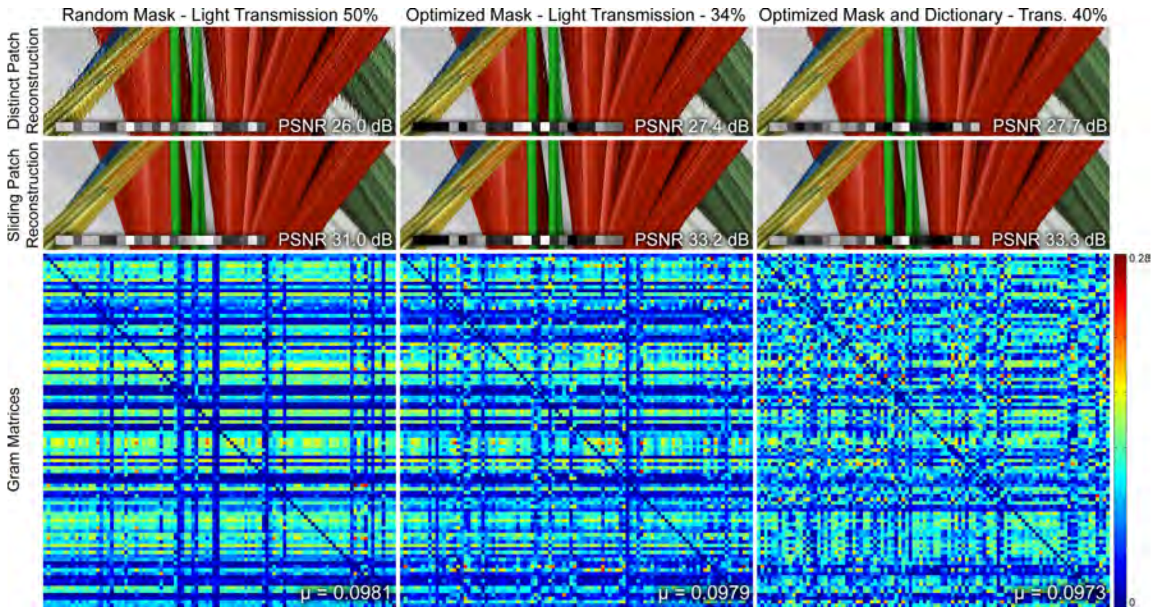

Figure S.7: *Evaluating optimal mask patterns. We evaluate three kinds of mask patterns for both sliding and distinct patch reconstructions for a flatland 2D light field. A light field is divided into patches of resolution 20 (space) × 5 (angle) and is projected down with the corresponding mask to a 20 pixel coded sensor image. As seen in the figure, given a dictionary, an optimized mask pattern performs much better than a traditional random mask. Jointly optimizing for both mask and dictionary performs slightly better than the optimized mask albeit at much higher computational times. The coherence value μ as shown in the figure decreases as we optimize for both the mask and the dictionary.*

$$\text{minimize} \quad \left\|\mathbf{I} - \mathbf{G}^T\mathbf{G}\right\|_F \tag{S.12}$$

where $\mathbf{G}$ is the product of the sensing matrix $\mathbf{\Phi}$ and the overcomplete dictionary $\boldsymbol{\mathcal{D}}$. This unconstrained problem is known to be convex, but the sensing matrices generated are generally dense and cannot be physically realized. To generate physical optimized code, we add physical constraints on our 1D mask patterns as

$$\begin{aligned} \underset{\{\mathbf{f}\}}{\text{minimize}} \quad & \left\|\mathbf{I} - \mathbf{G}^T\mathbf{G}\right\|_F \\ \text{subject to} \quad & 0 \leq f_i \leq 1, \, \forall i \end{aligned} \tag{S.13}$$

where $\mathbf{f} \in \mathbb{R}^m$ corresponds to the 1D mask pattern in the matrix and is zero otherwise. A further extension shown in [Duarte-Carvajalino and Sapiro 2009] is to perform coupled learning where, given a training set, one optimizes both for the dictionary and the sensing matrix.

$$\underset{\{\boldsymbol{\mathcal{D}},\mathbf{\Phi},\mathcal{A}\}}{\text{minimize}} \quad \lambda \left\|\mathbf{L} - \boldsymbol{\mathcal{D}}\mathcal{A}\right\|_F^2 + \left\|\mathbf{\Phi}\mathbf{L} - \mathbf{\Phi}\boldsymbol{\mathcal{D}}\mathcal{A}\right\|_F^2 \tag{S.14}$$

Here $\mathbf{L}$ corresponds to the light field training set, $\boldsymbol{\mathcal{D}}$ is the learned overcomplete dictionary, $\mathcal{A}$ is the sparse coefficient vector. and $\mathbf{\Phi}$ is the optimized sensing matrix. The constraints for the physically realizable mask pattern are then added as before. Figure S.7 compares reconstruction quality for all three mask patterns. As shown, given a fixed dictionary, the optimal mask pattern performs much better than the random projection. Joint optimization of dictionary and mask patterns performs marginally better than the stand-alone optimized mask albeit at an increased computational overhead that will not scale up to 4D light fields given current computational power.

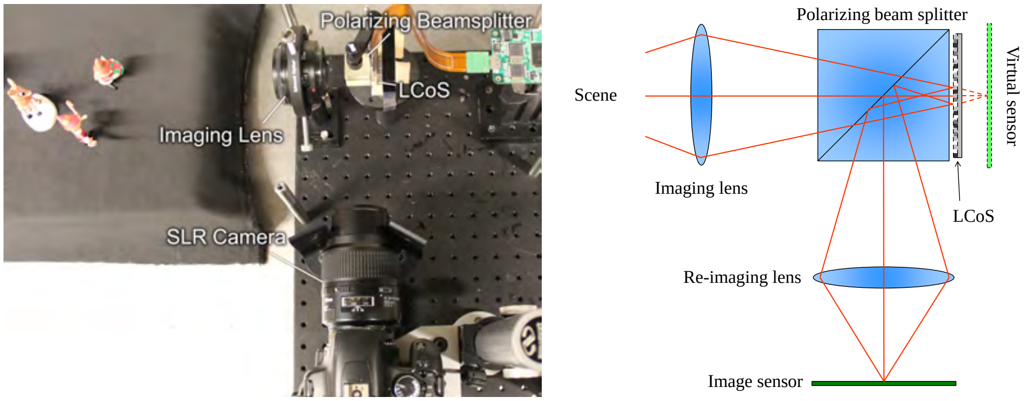

Figure S.8: *LCoS-based experimental optical setup (left) and a corresponding ray diagram (right).*

C Additional Hardware Implementation Details

This section provides additional information on the implementation issues for the LCoS-based prototype light field camera described in the main text. In addition, we also show another prototype using a printed transparency mask.

C.1 LCoS-based Prototype Light Field Camera

C.1.1 Hardware Setup

Figure S.8 shows a photo of our setup and a corresponding ray diagram. As described in the main text, the re-imaging lens of the SLR camera is focused in front of the LCoS mirror, thereby optically placing the (virtual) image sensor behind the LCoS plane. Therefore, the distance between the mask (LCoS plane) and the virtual image sensor can be adjusted by changing the focus of the re-imaging lens. To adjust the mask distance for capturing light fields with $p_\nu \times p_\nu$ angular resolution ($p_\nu = 5$ in our experiments), we display a pinhole array on the LCoS where the distances between adjacent pinholes are p_ν macropixels (Figure S.9(a)). We then place a piece of white cardboard in the scene and adjust the focus of the re-imaging lens so that disc-shaped blurred images under adjacent pinholes almost abut each other (Figure S.9(b)). In this way, the distance between the mask and the virtual sensor becomes such that the angular samples impinging on a sensor pixel pass through distinct macropixels on the LCoS with different attenuation values before getting integrated on the sensor.

C.1.2 Capturing Training Light Fields

To obtain training light fields for dictionary learning, we capture a variety of scenes using a traditional pinhole array-based method. However, as shown in Figure S.9(b), the discs under the pinholes (i.e., point-spread functions, or PSFs) have color-dependent nonuniform intensity distributions due to birefringence introduced by the LCoS pixels. To remove this effect from training light fields, we record images of a white cardboard scene also using shifting pinhole arrays to obtain a set of PSFs under each LCoS macropixels, and normalize training light fields by these images. Figure S.9(c) shows an example of a raw pinhole array image of a newspaper scene. By dividing Figure S.9(c) by the LCoS PSF image shown in Figure S.9(b), we obtain a normalized pinhole image shown in Figure S.9(d). The capture of the LCoS PSF is performed only once after the optical parameters have been determined according to Section C.1.1.

C.1.3 Calibration of the Projection Matrix

We measure the projection matrix $\mathbf{\Phi}$ by capturing the light field of a white cardboard scene modulated by the mask pattern. Again we use a shifting pinhole array, but rather than making each pinhole fully open (1.0) as in the case of

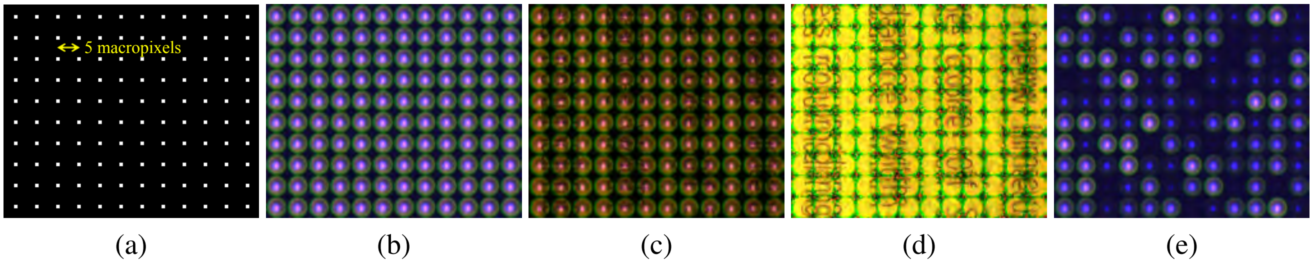

Figure S.9: *Pinhole array mask and captured images. For conciseness, only the region covering* 12×10 *pinholes are shown here. (a) Pinhole array displayed on the LCoS. (b) Image sensor recording of the LCoS pinhole array mask for a white cardboard scene. (c) Image sensor recording of the LCoS pinhole array mask for a newspaper scene. (d) Normalized image, i.e., (c) divided by (b). (e) Image sensor recording of the LCoS pinhole array mask for a white cardboard scene where each pinhole has a corresponding attenuation value according to the mask pattern.*

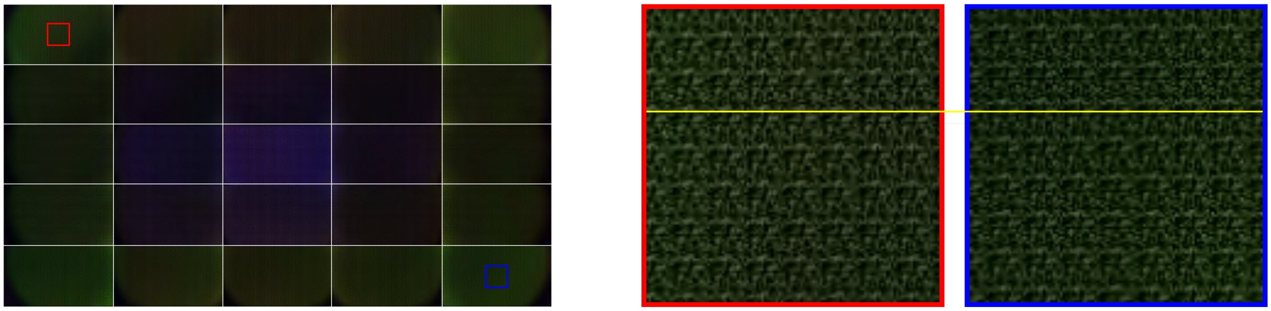

Figure S.10: *Calibrated projection matrix. Left: visualization of the matrix as a multi-view image. Right: magnified crops from the two views.*

training light field capture, we assign each pinhole a corresponding value ($\in [0,1]$) in the mask. An example of a mask-modulated pinhole array image is shown in Figure S.9(e).

We observed that the actual attenuation introduced by the LCoS was greater than the specified mask value. That is, the ratio of a captured pixel value in Figure S.9(e) to the corresponding pixel value in Figure S.9(b) was less than the LCoS macropixel value. To compensate for this non-linearity, we assumed gamma curve relationship between LCoS pixel values and actual attenuations, and performed linear search for the optimal gamma value for each color channel to obtain expected attenuation ratios.

Figure S.10 shows the calibrated projection matrix $\mathbf{\Phi}$, obtained from pinhole array images captured in the above-mentioned manner. Here, the projection "matrix" is shown as a multi-view image by rearranging the pixels of the pinhole images. Each view corresponds to the diagonal of each of the submatrices $\mathbf{\Phi}_j$. That is, the diagonal elements of $\mathbf{\Phi}_j$ are the lexicographically ordered pixel values of the j-th view. These views contain three effects: 1) shifted mask pattern (our optimized mask tiled over the LCoS area), 2) color-dependent nonuniform LCoS PSF effect, leading to color and intensity variations between the views, and 3) view-dependent vignetting from the imaging lens.

C.2 Printed Transparency-based Prototype Light Field Camera

Once the optical parameters are determined and dictionaries are learned, the proposed light field camera could be implemented in a more compact form factor with a static attenuation mask as shown in Figure S.11(a). We fabricate a mask holder that fits into the sensor housing of a Lumenera Lw11059 monochrome camera, and attach a film with a random mask pattern. As the printer guarantees 25 μm resolution, we conservatively pick a mask resolution of 50 μm, which roughly corresponds to 6×6 pixels on the sensor. We therefore downsample the sensor image by 6,

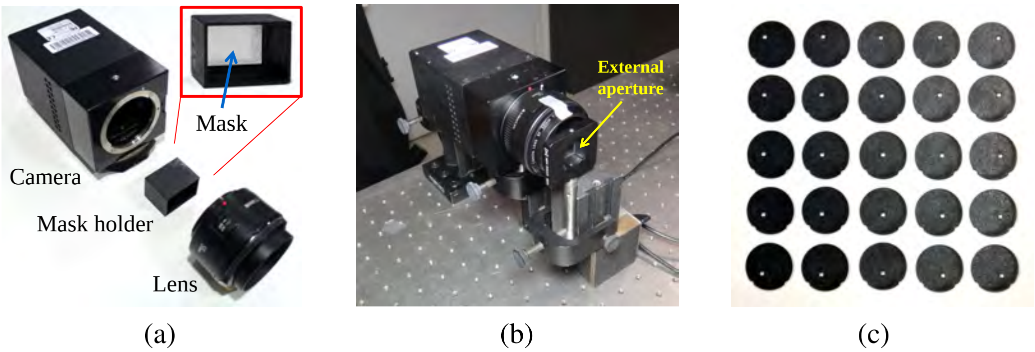

Figure S.11: *Printed transparency-based prototype camera (a) and calibration setups (b,c).*

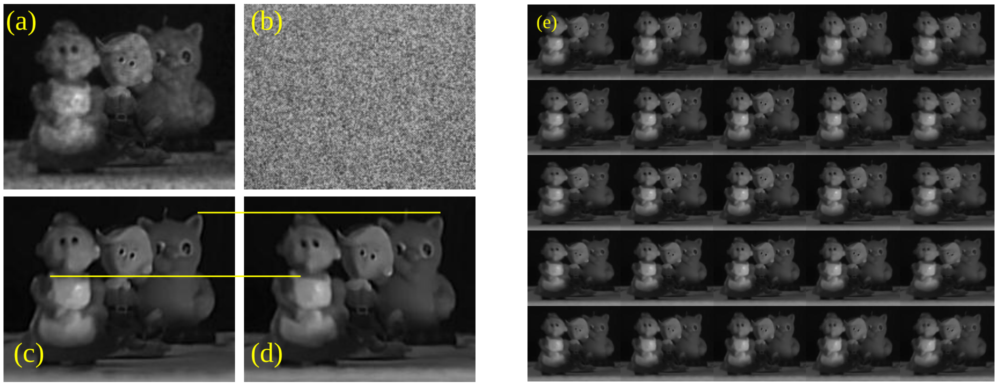

Figure S.12: *Light field reconstruction from an image modulated by a static film mask. (a) Captured image. (b) One view from the calibrated projection matrix* $\mathbf{\Phi}$*. (c,d) Two views from the reconstructed light field. Yellow lines are superimposed to show the parallax. (e) All of the* 5×5 *views from the reconstructed light field.*

and crop out the center 200×160 region for light field reconstruction in order to avoid mask holder reflection and vignetting. The distance between the mask and the sensor is 1.6 mm. A Canon EF 50mm f/1.8 II lens is used and focused at a distance of 50 cm.

As the mask is static, we use an aperture-based light field capture method to calibrate the projection matrix $\mathbf{\Phi}$. We place a $10 \times 10\,\text{mm}^2$ external aperture immediately in front of the lens as shown in Figure S.11(b), and capture a white cardboard scene with 5×5 sub-apertures each having an area of $2 \times 2\,\text{mm}^2$, as shown in Figure S.11(c).

Figure S.12 shows reconstruction results. The reconstruction quality is humble because we did not learn dictionaries or optimize mask patterns for this setup (we used the dictionary learned for the LCoS setup and used a random mask). Nevertheless, parallax was recovered.

D Additional Software Implementation Details

This section provides additional details on dictionary learning and our sparsity-exploiting reconstruction methods.

D.1 Overcomplete Dictionary Learning

For physical experiments we capture five training sets as shown in Figure S.13. Each light field in the training set has a resolution of 480×270 pixels in space and 5×5 views. We randomly extract about 360,000 overlapping patches from each of the training light fields, each patch has a spatial resolution of 11×11 pixels and an angular resolution of 5×5. To increase variability amongst these extracted patches we employ the coreset technique discussed in Section

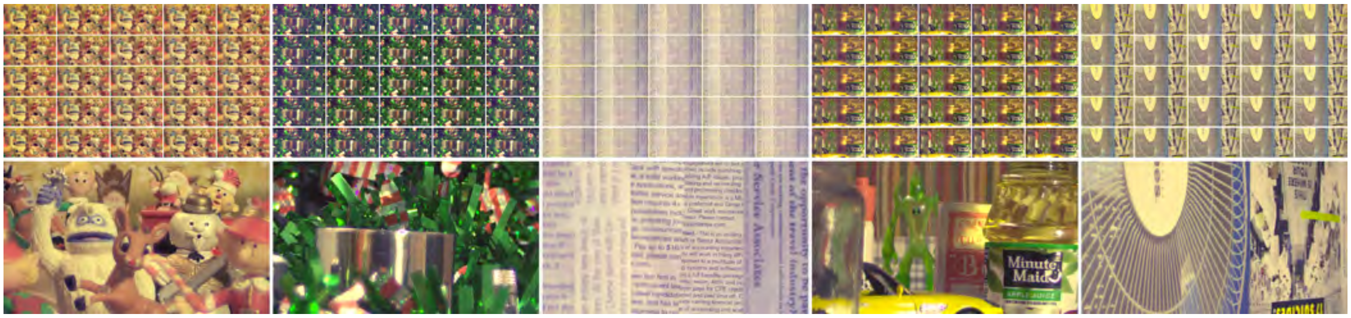

Figure S.13: *Training light fields and central views for physical experiments.*

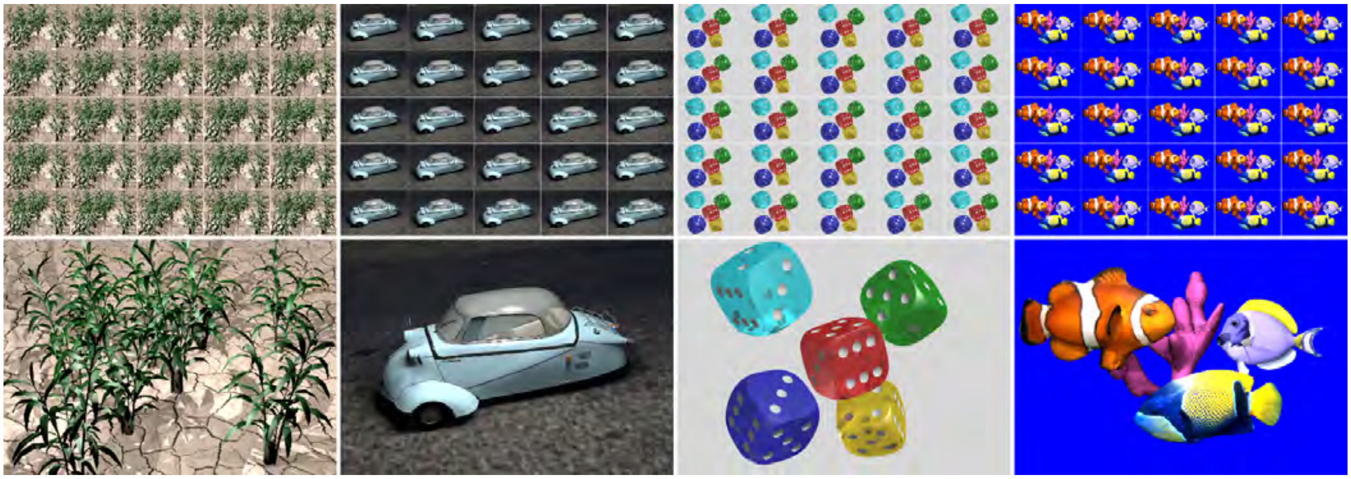

Figure S.14: *Training light fields and central views for simulated experiments.*

B.2 to reduce this set to tractable size of about 50,000 patches. This process is repeated for all the training light fields to generate a training set of about 250,000 patches. Coresets are again applied to reduce the final training set to about 50,000 patches.

The Sparse Modeling Software (SPAMS) [Mairal et al. 2010] is used to learn a $1.7\times$ overcomplete dictionary ($3,025$ rows, $5,000$ columns) with λ (error term) set to be 0.15 and number of iterations to be 10 in the toolbox parameters. Multi-threading was enabled for faster computations. It should be noted that a normalized vector of constants needs to be added to the computed dictionary to account for the DC component.

For simulated experiments, we use four synthetic training light fields, as shown in Figure S.14, to learn a dictionary. Each light field has a resolution of 840×593 pixels and 5×5 views. We apply hierarchical coresets, as described above, to learn a $1.2\times$ overcomplete dictionary with $2,025$ rows and $2,500$ columns. In these experiments, each light field patch has a resolution of 9×9 in space and 5×5 in angle.

D.1.1 Sparsity Exploiting Reconstruction

For smooth reconstructions, we divide an image captured with the prototype compressive light field camera into about 120,000 overlapping patches, each with a resolution of 11×11 pixels per color channel. Each of these patches forms a measurement vector for an independent optimization problem. Calibrated projection matrices for each patch are extracted from the full matrix (see Sec. C.1.3). L1-minimization is performed to recover a sparse set of coefficient vectors that represent the light field patches in the dictionary subspace. Such sparsity-exploiting algorithms perform convex optimization which makes them slow and intractable for problems of this magnitude. As discussed in Section A.4 we use a fast L1 minimizer [Yang et al. 2010]; in that solver, we set the sparsity penalizing parameter to 10, the tolerance to 0.001, and use 10,000 iterations.

It should be noted that at each step the product of the calibrated projection matrix and the dictionary needs to be normalized for correct minimization of the L1 norm. Each light field patch takes about 0.1 seconds to be recovered, resulting in runtime of about 18 hours for all there color channels on an 8-core Intel i7 machine with 16 GB of memory. Although the proposed method requires extensive computation times, we note that each patch is independent of all other patches. Hence, the reconstruction can be easily parallelized and significantly accelerated with modern high-end GPUs that have up to thousands of cores or cloud-based infrastructures.

E "Undappling" Sensor Images with Coupled Dictionaries

The proposed compressive camera setup uses a coded attenuation mask in the optical path. All captured images will contain the corresponding modulation or "dappling" pattern. Although this pattern can be removed optically, for instance using an SLM or mechanical movement of the mask, we discuss a purely computational approach in this section.

The most straightforward method to remove the mask pattern is to divide out the 2D projection of the mask pattern itself. That allows in-focus region to be free of mask dapples but renders the out-of-focus areas with high frequency artifacts. Another approach is to first recover a 4D light field and average out the views to get the image back, but that seems costly. To mitigate this problem, we use joint sparse coding to implicitly learn mappings from a modulated image to a demodulated or "undappled" image that still remains in the two-dimensional manifold of images in lieu of light field recovery. Let us consider two feature spaces $\mathbf{X}$ (consisting of mask modulated images of scenes as captured by our setup after dividing out the mask pattern) and $\mathbf{Y}$ (consisting of images of the same scenes as if there was no mask). Given these training sets, unlike traditional sparse coding that has been discussed so far, we employ joint sparse coding techniques to simultaneously learn two dictionaries $\boldsymbol{\mathcal{D}}_x$ and $\boldsymbol{\mathcal{D}}_y$ for the two feature sets such that the sparse representation of a mask modulated image $\mathbf{x}_i$ in terms of $\boldsymbol{\mathcal{D}}_x$ should be same as the representation of the corresponding demodulated image $\mathbf{y}_i$ in terms of $\boldsymbol{\mathcal{D}}_y$. Hence, if we know our measurement $\mathbf{x}_i$ we can recover its underlying signal $\mathbf{y}_i$. Mathematically, the joint sparse coding scheme [Yang et al. 2012] can be realized as:

$$\underset{\{\boldsymbol{\mathcal{D}}_x, \boldsymbol{\mathcal{D}}_y, \mathcal{A}\}}{\text{minimize}} \sum_i \|\mathbf{x}_i - \boldsymbol{\mathcal{D}}_x \boldsymbol{\alpha}_i\|_2 + \|\mathbf{y}_i - \boldsymbol{\mathcal{D}}_y \boldsymbol{\alpha}_i\|_2 + \lambda \|\boldsymbol{\alpha}_i\|_1 \tag{S.15}$$

The formulation requires a common sparse representation $\boldsymbol{\alpha}_i$ that should reconstruct both $\mathbf{x}_i$ and $\mathbf{y}_i$. As shown by Yang et al. [2012], we can convert the problem into standard sparse coding scheme with concatenated feature space of $\mathbf{X}$ and $\mathbf{Y}$. Now, given a new test modulated image $\mathbf{x}_t$, we find its representation in the learned dictionary $\boldsymbol{\mathcal{D}}_x$. The resulting coefficient vector is then multiplied to the dictionary $\boldsymbol{\mathcal{D}}_y$ to generate the demodulated image.

We captured modulated images from our setup and divide out the mean mask pattern. Each image is divided into overlapping patches of resolution 7×7, forming a training set for the first feature space. Correspondingly, training images without any modulation are also captured of the same scene and are divided into overlapping patches of resolution 7×7 to form the other training set. Joint sparse coding is then performed using software package described in [Zeyde et al. 2012]. Given a new modulated image, we first divide out the mean mask pattern and divide it into overlapping patches of resolution 7×7. Using our jointly trained dictionary, we reconstruct demodulated image patches which are then merged.

Supplementary References

AGARWAL, P. K., HAR-PELED, S., AND VARADARAJAN, K. R. 2004. Approximating Extent Measures of Points. *J. ACM 51*, 4, 606–635.

AHARON, M., ELAD, M., AND BRUCKSTEIN, A. M. 2005. K-SVD and its Non-negative Variant for Dictionary Design. In *Proc. SPIE Conference Wavelets*, 327–339.

AHARON, M., ELAD, M., AND BRUCKSTEIN, A. 2006. The K-SVD: An Algorithm for Designing of Overcomplete Dictionaries for Sparse Representation. *IEEE Trans. Signal Processing 54*, 11, 4311–4322.

AHARON, M. 2006. *Sparseness and Over-completeness in Signal Representation*. PhD thesis, Technion.

BECKER, S., BOBIN, J., AND CANDÈS, E. 2009. Nesta: A fast anad accurate first-order method for sparse recovery. In *Applied and Computational Mathematics*. http://www-stat.stanford.edu/ candes/nesta.

BOYD, S., AND VANDENBERGHE, L. 2004. *Convex Optimization*. Cambridge University Press.

CANDÈS, E., AND WAKIN, M. B. 2008. An Introduction to Compressive Sampling. *IEEE Signal Processing 25*, 2, 21–30.

CANDÈS, E., ROMBERG, J., AND TAO, T. 2006. Stable Signal Recovery from Incomplete and Inaccurate Measurements. *Comm. Pure Appl. Math. 59*, 1207–1223.

CANDÈS, E. J., WAKIN, M. B., AND BOYD, S. P. 2008. Enhancing Sparsity by Reweighted l1 Minimization. *Journal of Fourier Analysis and Applications 15*, 5, 877–905.

CANDÈS, E. J., ELDAR, Y. C., NEEDELL, D., AND RANDALL, P. 2011. Compressed Sensing with Coherent and Redundant Dictionaries. *Appl. and Comp. Harmonic Analysis 31*, 1, 59–73.

CHEN, S. S., DONOHO, D. L., MICHAEL, AND SAUNDERS, A. 1998. Atomic Decomposition by Basis Pursuit. *SIAM J. on Scientific Computing 20*, 33–61.

DONOHO, D. 2006. Compressed Sensing. *IEEE Trans. Inform. Theory 52*, 4, 1289–1306.

DONOHO, D. 2006. For Most Large Underdetermined Systems of Linear Equations, the Minimal ℓ_1-Norm Solution is also the Sparsest Solution. *Comm. Pure Appl. Math. 59*, 6, 797–829.

DUARTE-CARVAJALINO, J., AND SAPIRO, G. 2009. Learning to sense sparse signals: Simultaneous Sensing Matrix and Sparsifying Dictionary Optimization. *IEEE Trans. Im. Proc. 18*, 7, 1395–1408.

ELAD, M., AND AHARON, M. 2006. Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries. *IEEE Trans. Im. Proc. 15*, 12, 3736–3745.

FEIGIN, M., FELDMAN, D., AND SOCHEN, N. A. 2012. From High Definition Image to Low Space Optimization. In *Scale Space and Var. Methods in Comp. Vision*, vol. 6667, 459–470.

FELDMAN, D. 2010. *Coresets and Their Applications*. PhD thesis, Tel-Aviv University.

LEE, D. D., AND SEUNG, S. 1999. Learning the Parts of Objects by Non-negative Matrix Factorization. *Nature 401*, 788–791.

MAIRAL, J., SAPIRO, G., AND ELAD, M. 2007. Multiscale Sparse Representations with Learned Dictionaries. *Proc. ICIP*.

MAIRAL, J., PONCE, J., AND SAPIRO, G. 2010. Online Learning for Matrix Factorization and Sparse Coding. *Journal of Machine Learning Research 11*, 19–60.

NATARAJAN, B. K. 1995. Sparse Approximate Solutions to Linear Systems. *SIAM J. Computing 24*, 227–234.

TIBSHIRANI, R. 1996. Regression shrinkage and selection via the lasso. *J. Royal Statist. Soc B 58*, 1, 267–288.

TROPP, J. A., AND GILBERT, A. C. 2007. Signal Recovery From Random Measurements Via Orthogonal Matching Pursuit. *IEEE Trans. Information Theory 53*, 12, 4655–4666.

TROPP, J. A., AND WRIGHT, S. J. 2010. Computational Methods for Sparse Solution of Linear Inverse Problems. *Proc. IEEE 98*, 6, 948–958.

TROPP, J. A. 2004. *Topics in Sparse Approximation*. PhD thesis, University of Texas at Austin.

VAN DEN BERG, E., AND FRIEDLANDER, M. P. 2008. Probing the Pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing 31*, 2, 890–912. http://www.cs.ubc.ca/labs/scl/spgl1.

YANG, A., GANESH, A., SASTRY, S., AND MA, Y. 2010. Fast L1-Minimization Algorithms and An Application in Robust Face Recognition: A Review. Tech. rep., UC Berkeley.

YANG, J., WANG, Z., LIN, Z., COHEN, S., AND HUANG, T. 2012. Coupled Dictionary Training for Image Super-Resolution. *IEEE Trans. Im. Proc. 21*, 8, 3467–3478.

ZEYDE, R., ELAD, M., AND PROTTER, M. 2012. On single image scale-up using sparse-representations. In *Proc. Int. Conference on Curves and Surfaces*, 711–730.