EUROGRAPHICS 2015 / O. Sorkine-Hornung and M. Wimmer
(Guest Editors)

Volume 34 (2015), Number 2

Compressive Image Reconstruction in Reduced Union of Subspaces

Ehsan Miandji [†] and Joel Kronander and Jonas Unger

Linköping University, Sweden

Abstract

We present a new compressed sensing framework for reconstruction of incomplete and possibly noisy images and their higher dimensional variants, e.g. animations and light-fields. The algorithm relies on a learning-based basis representation. We train an ensemble of intrinsically two-dimensional (2D) dictionaries that operate locally on a set of 2D patches extracted from the input data. We show that one can convert the problem of 2D sparse signal recovery to an equivalent 1D form, enabling us to utilize a large family of sparse solvers. The proposed framework represents the input signals in a reduced union of subspaces model, while allowing sparsity in each subspace. Such a model leads to a much more sparse representation than widely used methods such as K-SVD. To evaluate our method, we apply it to three different scenarios where the signal dimensionality varies from 2D (images) to 3D (animations) and 4D (light-fields). We show that our method outperforms state-of-the-art algorithms in computer graphics and image processing literature.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction

Photo-realistic image synthesis is one of the long standing goals in computer graphics. The ever increasing demands on realism and visual fidelity have lead to the development of increasingly sophisticated algorithms for rendering and capture of visual data. However, this growth in realism and physical accuracy comes at the expense of increased computational complexity and long rendering times. The solution to this has traditionally been focused around research and development of new core rendering algorithms for efficient light transport. However, based on results from the field of *Compressed Sensing* (CS) [Don06] it can be shown that any sparse signal (e.g. visual data represented in a suitable basis) can be reconstructed with high quality from only a small number of measurements. This means that it is possible to accelerate existing techniques for rendering and capture of visual data by simply computing/measuring only a subset of the data, and apply CS-methods to compute an accurate estimate of the signal.

The performance of a CS algorithm is mainly dependent on the choice of the *dictionary*, a set of basis functions representing a signal. A signal exhibits different sparsity and sparsity patterns based on the dictionary. To minimize the reconstruction error, the representation of the signal in the dictionary should be as sparse as possible [Can08]. Most Previous CS algorithms for image synthesis have relied on analytical dictionaries, such as wavelets [SD11, PML*09] and Fourier bases [SD10]. However, due to the diversity of visual data, in e.g. natural images, a fixed analytical dictionary will not in general generate a sparse enough representation for accurate signal recovery. This leads to poor reconstruction quality and visual artifacts such as blurring, ringing, or speckle noise.

In this paper, we present a compressed sensing algorithm using a learning based dictionary to enable accurate reconstruction of incomplete and noisy visual data such as images, animations, and light-fields. The learning based approach enables the dictionary to exploit the sparsity patterns of the visual data and leads to a more sparse representation compared to fixed analytical dictionaries. In contrast to previous work, [MWBR13], who used a single 1D learned dictionary to enable CS for light fields, we train an

[†] e-mail:ehsan.miandji@liu.se

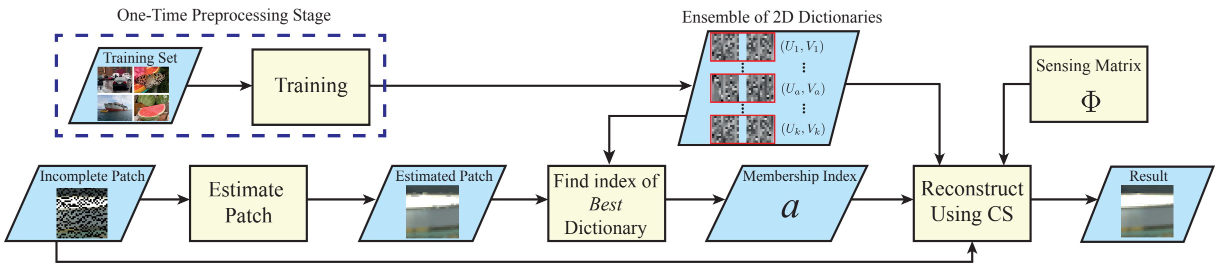


Figure 1: A flowchart of our method. The incomplete signal used here is extracted from an image, i.e. the 2D signal case.

ensemble of 2D dictionaries which admits a locally sparse representation. In other words, our method clusters the signal space while allowing sparse representation in each cluster. We show that signals represented by our method belong to a special case of the *reduced union-of-subspaces* signal model [ZMRE12, EKB10, BCW10], which admits a very sparse representation while keeping the size of dictionary minimal. A key property of our algorithm is that the sparsity pattern introduced by the dictionary ensemble exploits correlations in 2D. This enables near optimal recovery, and exhibits a very small memory footprint. The ensemble is trained only once for each specific signal type (images, videos, and light-fields).

The main contribution of this paper is a novel reconstruction method for noisy and incomplete visual data in 2D, 3D and 4D, combining compressed sensing with 2D dictionary ensembles enabling very sparse representations in reduced union of subspaces. To achieve this, we present a set of mathematical and algorithmic tools. In particular, we solve the problem of mapping the 2D sparse recovery in each subspace to an equivalent 1D form while preserving the properties inherent to the 2D dictionaries in the ensemble. The mapping of the problem from 2D to 1D form is an important contribution, as it enables the use of the highly efficient sparse solvers designed for 1D [WNF09,EHJT04,MBZJ09]. We also propose a greedy method for assignment of input data patches to a specific dictionary in the ensemble. The result is a robust method that, for the first time, achieves near optimal reconstruction of visual data. To the best of our knowledge, this is the first compressed sensing algorithm using the reduced union of subspaces model while exploiting 2D correlations and the sparsity induced by a dictionary ensemble.

To demonstrate the usefulness of our approach, we apply and evaluate our method in three user scenarios of different signal dimensionality: image plane (2D), animations (3D), and light field (4D) reconstruction. The evaluation shows that our method outperforms current state-of-the-art from the graphics and image processing literature. A key benefit of our method is that it is very robust to its parameters. This makes it possible to fix most of the parameters while providing control over the trade-off between quality and performance. Our numerical results are well supported by a theoretical analysis explaining the performance achieved by our method based on the sparsity induced using the ensemble of 2D dictionaries.

2. Algorithm Overview

In this Section, we provide a brief overview of our method as shown in Figure 1. Given an incomplete sampling of an *n*D visual dataset with noise (e.g. images, videos, or light-fields) and the learned ensemble of dictionaries, the goal is to accurately recover the original signal. Note that, the ensemble of dictionaries is computed in a one-time preprocessing stage (shown in a dashed box in Figure 1).

As a first step, we extract small 2D patches from the incomplete input dataset, where the locations of missing values are known. The reconstruction of a patch can be described as follows:

1. An initial, crude estimate of the patch is first computed by linear interpolation.
2. The estimated patch is then used to find the *best* dictionary using a greedy method. We call the index of such dictionary the *membership index*, denoted by a.
3. We use the location of missing values for the patch to form a binary sensing matrix.
4. The a:*th* dictionary, the incomplete patch, and the sensing matrix are then taken as input to the CS algorithm, which computes an accurate estimate of the sparse coefficients in the selected dictionary.
5. Using the coefficients obtained in step 4 and their corresponding dictionary, we reconstruct the patch.

We divide the detailed presentation of our method into three parts. The dictionary training algorithm is described in Section 3. The main part of the algorithm, sparse recovery, is presented in Section 4, followed by our proposed greedy algorithm for choosing a dictionary from the ensemble in Section 5. In Section 6 we provide an analysis of the parameters used by our algorithm. In Section 7 we then show that signals represented by our method belong to a special case of the *reduced union-of-subspaces* signal model. Finally, Section 8 provides an evaluation against current state-of-the-art methods in both computer graphics and image processing, and in Section 9 we discuss limitations of our approach and venues for future work.

Notation - Throughout the paper, lower case italic letters

	Dimensionality	Description
n	-	# of input patches
n_t	-	# of training patches
m_1	-	# of rows for input patch
m_2	-	# of columns for input patch
P_i	$m_1 \times m_2$	input patch
k	-	# of dictionaries
U_a	$m_1 \times m_1$	ath 2D dictionary in the
V_a	$m_2 \times m_2$	ensemble, $a = 1 \dots k$
X_{ia}	$m_1 \times m_2$	coefficients of patch i projected onto $(\mathrm{U}_a, \mathrm{V}_a)$
s_t	-	training sparsity
s	-	reconstruction sparsity
m	-	number of measurements
Φ_i	$m \times m_1 m_2$	sensing matrix for patch i

Table 1: Table of notations.

(s, k, ...) denote scalars, lower case letters (x, n, ...) denote vectors, and upper case letters (A, Φ, ...) denote matrices. The ℓ_2, ℓ_1 norms and the ℓ_0 pseudo-norm are denoted by $\|.\|_2$, $\|.\|_1$ and $\|.\|_0$, respectively. The notation vec(A), implies vectorization of A in a column-major fashion. The Kronecker product of matrices is denoted $\mathrm{A} \otimes \mathrm{B}$. Finally, the identity matrix is denoted by I. Table 1 summarizes the notations used in this paper.

3. Dictionary Training

We train an ensemble of 2D sparse dictionaries using an extended version of the Exemplar Orthogonal Bases (EOB) algorithm [GRBR10]. This method was introduced as an image compression approach to compete with JPEG, and has never been analyzed or used in the context of sparse signal representations or CS. Recently, Miandji et al. [MKU13] used EOB for compression of surface light-fields, enabling real-time reconstruction on the GPU. In this paper, we propose a novel CS framework that given incomplete and noisy data, achieves near-optimal recovery.

In this Section, we provide a brief overview of EOB and show that this method can be viewed as a manifold learning algorithm. Additionally, in Section 7 we present a novel interpretation of EOB in the context of sparse signal representation. The input data for the training phase is a set of complete training patches $\mathrm{P}_i \in \mathbb{R}^{m_1 \times m_2}$, $i = 1 \dots n_t$. The way we construct these patches is dependent on the application and will be described in Section 8.

The EOB algorithm aims at computing a set of $k \ll n_t$ dictionaries $\{\mathrm{U}_a \in \mathbb{R}^{m_1 \times m_1}, \mathrm{V}_a \in \mathbb{R}^{m_2 \times m_2}\}_{a=1}^{k}$, such that each P_i can be represented using sparse coefficients in *one* of these 2D dictionaries with minimal error. Formally, each signal can be represented as $\mathrm{P}_i = \mathrm{U}_a \mathrm{X}_{ia} \mathrm{V}_a^T$, where $\mathrm{X}_{ia} \in \mathbb{R}^{m_1 \times m_2}$ is the sparse coefficient matrix with at most s_t non-zero elements. From this definition, it is clear that $\|\mathrm{X}_{ia}\|_0 \leq s_t$ and $1 \leq s_t \leq m_1 m_2$. One can immediately notice the resemblance of this formula to Singular Value Decomposition (SVD). But there are two major differences. First, $\{\mathrm{U}_a, \mathrm{V}_a\}_{a=1}^{k}$ are trained in a way that X_i is arbitrarily sparse, rather than being diagonal. Second, the algorithm trains a set of $k \ll n_t$ basis pairs, as opposed to only one computed by SVD. Compared to 1D dictionaries [AEB06, MWBR13], a clear advantage of this representation is that the coherence along rows and columns of P_i is exploited simultaneously, allowing for a very sparse representation while keeping the size of dictionary minimal.

Given the parameters k and s_t, the EOB algorithm trains a set of 2D dictionaries $\{\mathrm{U}_a, \mathrm{V}_a\}_{a=1}^{k}$ by minimizing the following energy function:

$$E(\{\mathrm{U}_a, \mathrm{V}_a, \mathrm{X}_{ia}, \mathrm{M}_{ia}\}) = \sum_{i=1}^{n_t} \sum_{a=1}^{k} \mathrm{M}_{ia} \|\mathrm{P}_i - \mathrm{U}_a \mathrm{X}_{ia} \mathrm{V}_a^T\|_2^2 \quad (1)$$

$$\text{subject to} \quad \mathrm{U}_a^T \mathrm{U}_a = \mathrm{I},\ \mathrm{V}_a^T \mathrm{V}_a = \mathrm{I},\ \forall a,$$
$$\|\mathrm{X}_{ia}\|_0 \leq s_t,$$
$$\sum_{a=1}^{k} \mathrm{M}_{ia} = 1,\ \forall i,$$

where, M_{ia} is an element in a binary matrix, $\mathrm{M} \in \mathbb{R}^{n_t \times k}$, associating each signal to its corresponding dictionary. Since each P_i uses one dictionary in the ensemble, each row of M has a single non-zero entry. The last constraint in Eq. (1) enforces this. An iterative algorithm for solving Eq. (1) is presented in [GRBR10].

The EOB algorithm tries to find disjoint low-dimensional subspaces in $\mathbb{R}^{m_1 \times m_2}$. An illustrative example is given in Figure 2, where we consider patches in $\mathbb{R}^3$ and a subspace dimensionality (or sparsity) of 2. These subspaces are shown as planes. The goal of EOB is to find such subspaces and the set of patches that can be represented efficiently in each subspace. From this illustration and the discussion above, it is clear that these mappings are bijective and smooth. Therefore, one can interpret the EOB algorithm as a manifold learning technique, where a chart is defined as a local neighborhood of points that are projected onto a lower dimensional subspace (i.e. a low-rank approximation is performed in each chart). It is important to note that the charts are defined based on sparsity rather than proximity of points in signal space. In other words, two signals are in the same chart if they are sparse in the basis defined for that chart. Low-rank representations have previously been utilized for visual data representations, see e.g. [?]. These methods assume that the signal is low-rank in a single subspace. In contrast, EOB performs a clustering stage with the metric defined over sparsity, see Eq. (1), which leads to a collection of low dimensional subspaces.

A disadvantage of training-based methods is that the quality of reconstruction deteriorates when the training and testing sets are considerably different. For instance, consider the case when the dictionary ensemble is trained on natural images, but is used for reconstructing a binary image. Thanks to

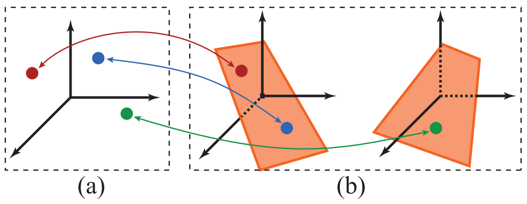
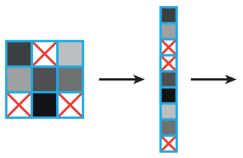


Figure 2: An illustration of 2-sparse points in $\mathbb{R}^3$. (a) Three points in $\mathbb{R}^3$, where each point is representative of a patch P_i. (b) Two two-dimensional subspaces (planes) embedded in $\mathbb{R}^3$. Each point in $\mathbb{R}^3$ is mapped to a 2D plane. The mappings are shown with arrows.



$$\begin{pmatrix} 1&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&1&0 \end{pmatrix}$$

Figure 3: A 3×3 patch is first vectorized and then the corresponding sensing matrix is constructed. The empty pixels are marked with a cross.

independence of the dictionaries in the ensemble, this issue can be solved easily by training a small set of dictionaries based on the new data (binary images). These new dictionaries can then be added to the set of existing ones that were computed using natural images. In addition, this update procedure is relatively fast, since we only train a small number of dictionaries. Results for this simple approach will be presented in Section 8.1. Note that other learning-based methods such as K-SVD [AEB06] do not share this property and cannot be expanded without re-training the dictionary.

4. Reconstruction From Incomplete Data

Given the ensemble of dictionaries and a set of 2D incomplete and noisy patches, P_i, $i = 1 \dots n$, our goal here is to recover the sparse coefficients with minimal error. To achieve this, we will cast the problem of 2D compressed sensing to an equivalent 1D form, enabling us to use conventional sparse coding algorithms, while preserving the benefits of the 2D ensemble.

Let us first present an overview of CS for 1D signals. Define $p_i = \mathrm{vec}(P_i) \in \mathbb{R}^{m_1 m_2}$. In compressed sensing, the measurement model for a signal is expressed as: $y_i = \Phi p_i + w$, where $y_i \in \mathbb{R}^m$ represents m measurements and $\Phi \in \mathbb{R}^{m \times m_1 m_2}$, $m < m_1 m_2$, is the measurement or sensing matrix. In addition, $w \in \mathbb{R}^m$ is the measurement noise, often assumed to be the zero mean white Gaussian noise with variance σ^2, i.e. $w \sim \mathcal{N}(0, \sigma^2 I)$. It is assumed that the signals are sparse or compressive in a certain orthogonal or overcomplete dictionary $\Psi \in \mathbb{R}^{m_1 m_2 \times k}$, $k \geq m_1 m_2$. Formally, $p_i = \Psi x_i$, where the sparse coefficient vector x_i has at most s non-zero elements, $\|x_i\|_0 \leq s \ll k$. Define $A = \Phi\Psi$, then given y_i, one can recover the sparse coefficients for p_i by solving the following problem, known as Basis Pursuit DeNoising (BPDN) [Tib94]:

$$\hat{x}_i = \min_{x} \|x\|_1 \quad \text{s.t.} \quad \|Ax - y_i\|_2 < \varepsilon, \tag{2}$$

Once the coefficients are recovered, the reconstructed patch can be computed as $\hat{p}_i = \Psi \hat{x}_i$.

We now show how the 2D compressed sensing problem can be cast into the form of Eq. (2). For a single 2D dictionary, (U_a, V_a), and a set of 2D patches, P_i, we can rewrite Eq. (2) as follows:

$$\hat{X}_{ia} = \min_{X} \|X\|_1 \quad \text{s.t.} \quad \|Y_i - \Phi_{m_1} U_a X V_a^T \Phi_{m_2}^T\|_2 < \varepsilon, \tag{3}$$

where $(\Phi_{m_1} \in \mathbb{R}^{m \times m_1}, \Phi_{m_2} \in \mathbb{R}^{m \times m_2})$ are two sensing matrices that operate on rows and columns of each P_i, respectively; i.e. we have $Y_i = \Phi_{m_1} P_i \Phi_{m_2}^T$. Equation (3) is solved independently for all $i = 1 \dots n$. Observe that (Φ_{m_1}, Φ_{m_2}) are different for each patch, but we drop the index i for notational brevity. Using the properties of the Kronecker product, we can rewrite the constraint of Eq. (3) as follows:

$$\begin{aligned} &\left\| \mathrm{vec}(Y_i) - \mathrm{vec}\left[(\Phi_{m_1} U_a) X (V_a^T \Phi_{m_2}^T)\right] \right\|_2 \\ &= \left\| \mathrm{vec}(Y_i) - \left[(V_a^T \Phi_{m_2}^T)^T \otimes (\Phi_{m_1} U_a)\right] \mathrm{vec}(X) \right\|_2 \\ &= \|\mathrm{vec}(Y_i) - [(\Phi_{m_2} V_a) \otimes (\Phi_{m_1} U_a)]\, \mathrm{vec}(X)\|_2 \\ &= \|\mathrm{vec}(Y_i) - [(\Phi_{m_2} \otimes \Phi_{m_1})(V_a \otimes U_a)]\, \mathrm{vec}(X)\|_2 . \end{aligned} \tag{4}$$

In general, if U_a is a basis in $\mathbb{R}^{m_1 \times m_1}$ and V_a is a basis in $\mathbb{R}^{m_2 \times m_2}$, then $V_a \otimes U_a$ forms a basis in $\mathbb{R}^{m_1 m_2 \times m_1 m_2}$ [DB12]. By substituting Eq. (4) into Eq. (3) and letting $x_{ia} = \mathrm{vec}(X_{ia})$, $y_i = \mathrm{vec}(Y_i)$, we have:

$$\begin{aligned} \hat{x}_{ia} = \min_{x} \|x\|_1 \qquad &\text{s.t.} \\ &\|y_i - (\Phi_{m_2} \otimes \Phi_{m_1})(V_a \otimes U_a) x\|_2 < \varepsilon, \end{aligned} \tag{5}$$

which is equivalent to Eq. (2) for a separable sensing matrix and a 2D dictionary in the ensemble. We refer to the ensemble of 1D dictionaries formed by $\{V_a \otimes U_a\}_{a=1}^{k}$ as the Kronecker-EOB ensemble, or K-EOB in short. A similar formulation was presented in [RS09], where the authors propose an imaging method with separable sensing matrices and a fixed analytical 2D dictionary. In contrast, our method utilizes an ensemble of trained 2D dictionaries.

For exact recovery, it has been shown in [RS09] that a separable sensing matrix requires $\sqrt{\log_{10}(\max(m_1, m_2))}$ more measurements compared to a single sensing matrix as defined in Eq. (2). Therefore, we define a single sensing matrix that operates on vectorized representation of the patch. Accordingly, Eq. (5) becomes:

$$\hat{x}_{ia} = \min_{x} \|x\|_1 \quad \text{s.t.} \quad \|y - \Phi_i (V_a \otimes U_a) x\|_2 < \varepsilon, \tag{6}$$

where $y_i = \Phi_i p_i$ and $\Phi_i \in \mathbb{R}^{m \times m_1 m_2}$ is a sensing matrix. Note that Equations (3) and (6) are equivalent; i.e. the changes introduced in Eq. (4) are merely a reformulation of Eq. (3), which does not affect the properties of the 2D dictionary ensemble presented in Section 3. Therefore, the 1D K-EOB ensemble possesses all the properties of the 2D EOB ensemble and exploits the coherence of a signal along its rows and columns simultaneously. As a result, once the coefficients are recovered by solving Eq. (6), the 1D and 2D patches can be computed as $\mathrm{p}_i = (\mathrm{V}_a \otimes \mathrm{U}_a)\hat{\mathrm{x}}_{ia}$ and $\mathrm{P}_i = \mathrm{U}_a\hat{\mathrm{X}}_{ia}\mathrm{V}_a^T$, respectively.

In this work we only show examples using binary sensing matrices that extract certain elements of the patch at random. For signals in $\mathbb{R}^{m_1 m_2}$ we construct an identity matrix of the same size. We then remove $m_1 m_2 - m$ rows at random. An Illustrative example is shown in Figure 3. An advantage of this simple construction is that we do not need to store sensing matrices, but just the indices of missing values. Note that our method is not limited to this choice. In fact, sensing matrices constructed from Gaussian and sub-Gaussian distributions, along with optimized sensing matrices [DCS09], can be applied directly.

5. Approximating Dictionary Membership Index

A key step in our algorithm is to, for each patch, find the index, a, of the best dictionary in the ensemble. A naïve approach would be to solve Eq. (6) for all dictionaries, $\{\mathrm{V}_a \otimes \mathrm{U}_a\}_{a=1}^k$, and pick the best result based on the reconstruction error. However, since we do not have the complete patch, it is impossible to define a metric for choosing the best result of Eq. (6). Moreover, this approach is computationally expensive which renders it impractical for the applications we considered in this paper. In this Section, we introduce a fast greedy algorithm that finds the index of a K-EOB dictionary in the ensemble which produces the most sparse coefficients with the least reconstruction error. We use the term *membership index* for the index of such a dictionary (denoted by a in Eq. (6)).

Our greedy method for computing the membership index is outlined in Algorithm 1. We project a patch, p_i, onto all K-EOB dictionaries in the ensemble, leading to a set of coefficient vectors, x_{ia}, see line 4. Given a threshold parameter, τ, and a value for maximum sparsity, s, we incrementally nullify $m_1 m_2 - t$, $t = 1 \dots s$, elements from x_{ia} with smallest absolute value, while the reconstruction error is above τ; i.e. we are gradually adding more coefficients until a maximum number of coefficients is reached or our reconstruction error is good enough. The index of the dictionary that produces the sparsest coefficient vector for p_i is stored at m_i, shown at line 11. For the case that we reach the maximum sparsity and the threshold constraint is not satisfied, the index of the dictionary with least reconstruction error is stored. Algorithm 1 is applied on all patches p_i, $i = 1 \dots n$.

If the incomplete patches are used directly in Algorithm 1, the probability of misclassification will be increased. Instead, we form an estimate of each patch by linear interpolation and feed the results to Algorithm 1. For this purpose, we use the Delaunay Linear Interpolation (DLI) algorithm, a fast and commonly used method for scattered data

Algorithm 1 Compute Dictionary Membership Index a

```
Input: A vectorized patch $p_i$, tolerance $\tau$, sparsity $s$ and the
    K-EOB ensemble $\{V_a \otimes U_a\}_{a=1}^k$
Output: The dictionary membership index $a$
 1: $e \in \mathbb{R}^k \leftarrow \infty$ and $z \in \mathbb{R}^k \leftarrow \mathbf{1}$
 2: for $a = 1 \dots k$ do
 3:     $x_{ia} \leftarrow (V_a \otimes U_a)^T p_i$
 4:     while $z_a \leq s$ and $e_a > \tau$ do
 5:         $y \leftarrow$ Nullify $m_1 m_2 - z_a$ smallest elements of $x_{ia}$
 6:         $e_k \leftarrow \|p_i - (V_a \otimes U_a) y\|_2^2$
 7:         $z_a = z_a + 1$
 8:     end while
 9: end for
10: $a \leftarrow$ index of min(z)
11: if min(z) $= s$ then
12:     $a \leftarrow$ index of min(e)
13: end if
```

interpolation. We found that performing DLI does not affect the computational complexity of the overall framework and improves our results by a relatively small margin. Figure 4 compares three types of input for Algorithm 1: complete, interpolated and incomplete patches. The dashed line represents DLI applied directly onto incomplete patches without compressed sensing. Although the approximation using DLI gives very poor results (dashed-line in Fig. 4), when we use this crude approximation in Algorithm 1, the performance of CS algorithm is slightly improved. These results also imply that the subspaces spanned by the ensemble are fairly distinct, which allows Algorithm 1 to detect a subspace close to the optimal one, even when the data is incomplete.

6. Parameters Analysis

Our method has the following parameters: training sparsity (s_t), number of dictionaries in the ensemble (k), Algorithm 1 threshold (τ), Algorithm 1 sparsity (s), patch size, and the number of measurements (m). In order to analyze how sensitive our method is to parameter changes, we performed an evaluation, where we varied one parameter at the time keeping the other three fixed and measured the reconstruction quality using PSNR. Figure 5 plots reconstruction quality given a large range of values for s_t, k, τ, and s. For these tests, we used the *castle* image (also used in Table 3). Relatively small changes in PSNR suggests that our method is not very sensitive to parameters. This is an important aspect which distinguishes our method from most of the previous work (see Section 8). The robustness of our algorithm allowed us to keep k and τ fixed for all of the applications (images, animations, light fields) described in Section 8. The parameters s_t and s were adjusted between applications, but kept fixed for all experiments within each application respectively. Keeping s_t, k, τ, and s fixed enables the parameter m to control the trade-off between quality and performance. Patch size does not have a significant affect on image quality

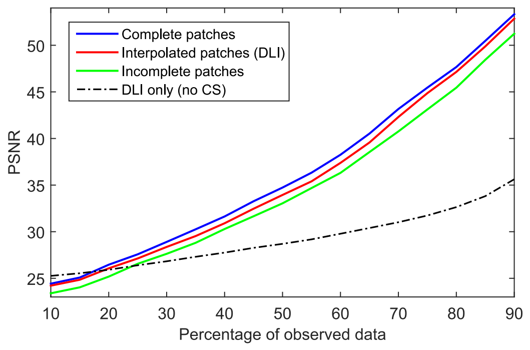


Figure 4: Percentage of observed data vs. reconstruction quality for three types of input for Algorithm 1, compared to naïve solution of linear interpolation (dashed line).

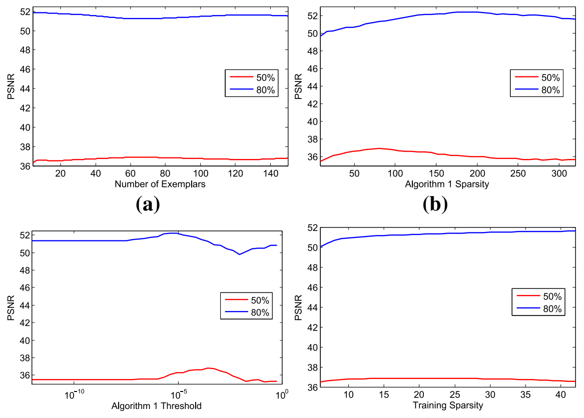


Figure 5: Reconstruction of the *Castle* image using 50% and 80% of pixels while changing the parameters. (a) $\tau = 10^{-4}$, $s = 18$, $s_t = 18$, and variable k. (b) $\tau = 10^{-16}$, $k = 64$, $s_t = 18$, and variable s. (c) $s = 432$, $k = 64$, $s_t = 18$, and variable τ. (d) $\tau = 10^{-4}$, $s = 18$, $k = 64$ and variable s_t. Patch size is 12×36, and therefore $s \leq 432$.

and was set heuristically based on computation time for each application.

Regardless of application, one can construct overlapping or non-overlapping patches from the input data. In the overlapping case, a patch is centered around each data point (e.g. each pixel). After reconstructing all patches, the final value for a data point is computed from patches that cover it [EA06]. Overlapping patches lead to a significant improvement in quality, but they are computationally expensive. However, one can define the distance between overlapping patches, e.g. every other or every third data point. The distance between overlapping patches provides a quality-performance trade-off.

7. Theoretical Analysis

The sparse signal model closest to K-EOB is the *reduced union-of-subspaces* model [BCW10] and its special case, the *one-block-sparse* model [EKB10]. In this model, the dictionary is composed of several blocks of atoms and each signal is dense in a single block (subspace) while it is sparse in the union of subspaces. Considering each K-EOB dictionary in the ensemble as a block, we can easily define a block-sparse dictionary by concatenating all $\mathrm{U}_a \otimes \mathrm{V}_a$ next to each other. The resulting block-sparse dictionary will be of size $m_1 m_2 \times k m_1 m_2$. Since U_a and V_a are both orthogonal, so is $\mathrm{U}_a \otimes \mathrm{V}_a$. Therefore, atoms in each block are linearly independent, conforming to the requirement of a block-sparse dictionary.

In fact, K-EOB can be considered as a special case of the one-block-sparse model, where in addition to the sparsity in the reduced union of subspaces, the signal is also sparse in the single subspace it belongs to. In other words, after selecting a subspace out of k disjoint subspaces (Algorithm 1), one has to construct a subspace of dimensionality s inside this subspace. This task involves searching for a subspace in $\binom{m_1 m_2}{s}$ subspaces, see Eq. (6). These types of nested sparsity patterns are termed *hierarchical sparsity* [SRSE11]. In case of K-EOB, we have two levels of sparsity: the inter-block sparsity and the intra-block sparsity. Unfortunately, natural images and their higher dimensional variants are too diverse to be analytically categorized based on their sparsity pattern. However, Our numerical results show that the model used here is well-suited for this class of signals. In addition, using a block-sparse representation, Zelnik-Manor et al. [ZMRE12] also report superior results compared to traditional overcomplete representations for natural images [AEB06].

By clustering images based on their sparsity, the K-EOB ensemble is able to exploit non-local coherencies in an image, or a light-field. This clustering is done via the membership index. Patches that are represented by one dictionary are not necessarily close to each other in the image space, yet they have minimal distance in the space spanned by the dictionary. This coherency, or sparsity, in the dictionary space is further exploited by projecting these patches to a lower dimensional space.

As mentioned before in Section 1, sparsity is an important factor in faithful reconstruction. To see why our method leads to better results, we compare the EOB and K-EOB ensembles with a conventional overcomplete dictionary trained with K-SVD and a one-block-sparse dictionary trained with the BK-SVD method [ZMRE12]. Our comparison is based on the sparsity induced by these dictionaries. To have a common ground for comparison, we assume that the size (storage cost) of these dictionaries are equal. We show that the K-EOB ensemble admits a more sparse signal representation, which in turn leads to a more accurate reconstruction. Assume that we have square patches of size $l = m_1^2$. An overcomplete dictionary with βl atoms is of size $l \times \beta l$, where β is a multiplier that defines overcompleteness. In order to have a same size ensemble for EOB, the number of dictionaries

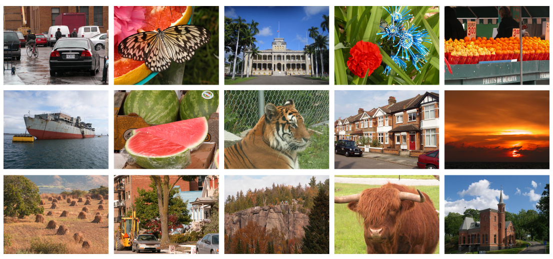


Figure 6: A subset of natural images we used for training

should be:

$$\frac{\text{size of EOB}}{\text{size of overcomplete}} = \frac{k(l+l)}{\beta l^2} = 1 \quad \Rightarrow \quad k = \frac{1}{2}\beta l \qquad (7)$$

If we form the equivalent K-EOB ensemble, and by using Eq. (7), we can see that the K-EOB dictionary has

$$\frac{\text{\# EOB atoms}}{\text{\# overcomplete atoms}} = \frac{kl}{\beta l} = \frac{0.5\beta l^2}{\beta l} = \frac{l}{2},$$

times more atoms than a conventional overcomplete dictionary with the same size. For instance, assuming signals of size 12×12, our dictionary is, implicitly, 72 times more overcomplete. Block-sparse dictionaries also possess this property. An advantage of using the K-EOB ensemble over a block sparse dictionary is that the former admits sparsity in each block, while the latter assumes dense coefficients for each block of the dictionary. Therefore, the K-EOB ensemble is more sparse than an equivalent one-block-sparse dictionary. In addition, the computational cost of training a block sparse dictionary that has the same size as the K-EOB ensemble is much higher.

8. Evaluation

We demonstrate the usefulness of our method in three different applications: 2D image reconstruction (Section 8.1), 3D image sequence reconstruction (Section 8.2), and 4D light-field reconstruction (Section 8.3). To solve Eq. (6), we used the Smoothed-ℓ_0 algorithm [MBZJ09]. Comparing Smoothed-ℓ_0 to different methods, including the reportedly best performing algorithm in [MWBR13], this method produced the best results with lowest computation time. We implemented our framework in MATLAB, while computationally expensive parts such as training, Algorithm 1, and the Smoothed-ℓ_0, were implemented in C++. All the results were obtained on a machine with 16 physical cores, running at 2.0GHz with 24GB of RAM. We used Peak Signal to Noise Ratio (PSNR) as our image quality metric. For our method, the timing results include the run-time of DLI, Algorithm 1 and Smoothed-ℓ_0 for solving Eq. (6).

In all our results, we set $\tau = 10^{-4}$ and $k = 32$. Since signal dimensionality changes per application, we have different values for training sparsity, s_t. The sparsity in Algorithm 1 is set to the same value, i.e. $s = s_t$. The threshold for Smoothed-ℓ_0, denoted as ε in Eq. (6), was fixed for all the tests to 10^{-4}.

8.1. Image Reconstruction

In this application, a photo-realistic renderer generates an image while randomly skipping a set of pixels. The locations of missing pixels are stored in a binary mask. From the incomplete image and its corresponding binary mask, we construct a set of overlapping patches centered around each pixel. The locations of missing pixels in the binary mask patches are used for constructing sensing matrices, as shown in Figure 3. The input to our reconstruction method is the set of sensing matrices and incomplete image patches. In the evaluation, we consider image reconstruction using 20%, 60%, 70%, and 80% of the original pixels. Our approach is purely image-based and independent of the scene information. It can therefore be applied as an addon to increase the efficiency of most existing rendering and image sampling algorithms [LAC*11, ODR09].

We used the LuxRender to render three scenes with different characteristics, shown in Figure 7. The scenes were rendered using the Metropolis sampler [**?**] and the bi-directional integrator [**?**]. Fixing the resolution at 1280×960, the rendering times for the *Bedroom*, *Bottles* and *Bamboo* scenes were 6, 48 and 12 hours, respectively (utilizing 16 physical cores on the CPU). Despite long rendering times, these images have moderate noise. Due to the high-frequency details in the *Bottles* and the *Bamboo* scenes, they are considered as the most challenging test scenarios. In particular, the *Bamboo* scene was rendered using a pinhole camera, leading to very sharp edges across the image plane. Since we wanted to compare our results to the ground truth, we performed a full rendering of each scene and tone-mapped the image. Afterwards, we randomly removed a percentage of pixels and constructed overlapping patches. Our method can be used on High Dynamic Range (HDR) results of the renderer directly, provided that the dictionary is also trained on HDR natural images. For all the results reported here, we computed PSNR on the reconstructed data in the 0-255 range, but before quantization.

For each color channel, we trained a 6-sparse ensemble of 32 dictionaries on 12×12 non-overlapping patches extracted from 86 natural images (i.e. $m_1 = 12$, $m_2 = 12$, $k = 32$, and $s_t = 6$). These images were collected from different databases and a subset is shown in Figure 6. We did not observe any changes in the results by using a different set of natural images. However, the number of training patches, n_t, affects the results. Since theoretical bounds for n_t are well-beyond what is required in practice [GE11], we heuristically require at least 10^5 training patches for all applications.

We compare our approach to the following methods: DLI, Steering Kernel Regression (SKR) [TFM07], K-SVD [AEB06, EA06] and Compressive Rendering (CR) [SD11]. The DLI method is implemented in MATLAB as the `griddata` function. For SKR, we used the reference implementation provided by the authors. This method is very sensitive to parameters and requires fine tuning for

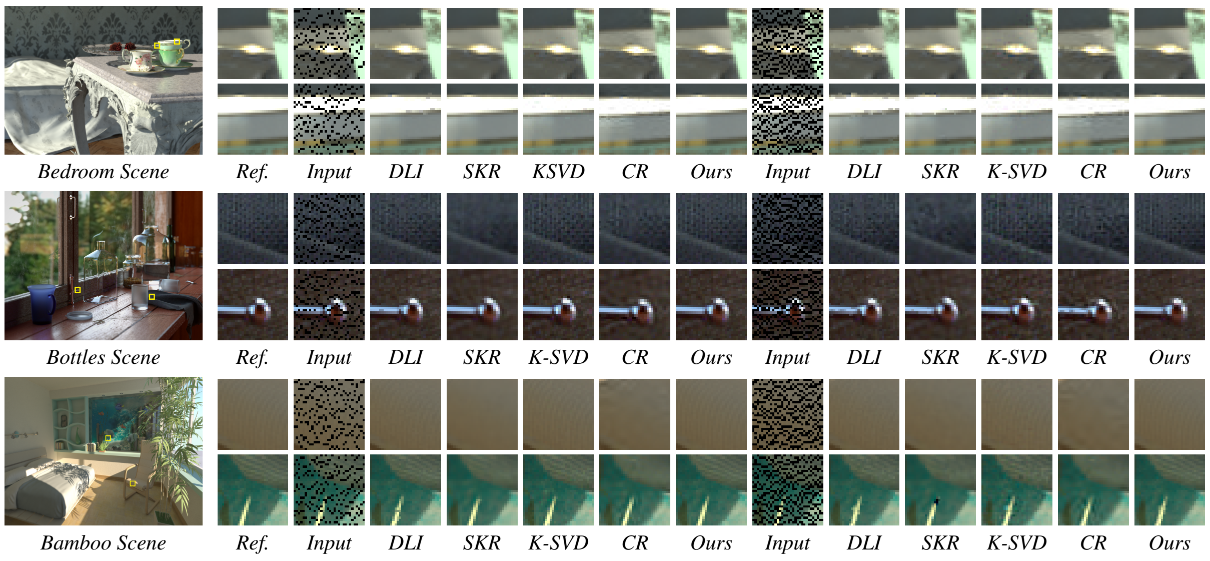


Figure 7: Image quality comparison for image plane reconstruction using DLI, SKR, CR, K-SVD and our method. For each scene, we extracted three insets. For all the methods, the insets represent reconstruction from 80% of pixels (left) and 60% of pixels (right).

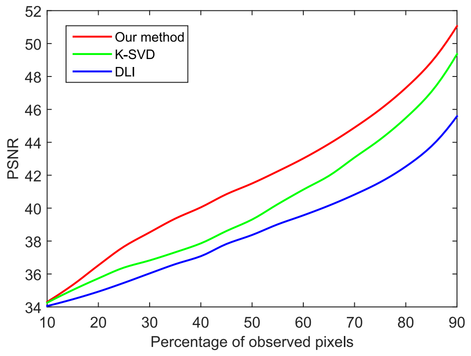


Figure 8: PSNR comparison for the *Bottles* scene between our method, K-SVD, and DLI for a large range of available pixels.

	Ratio	DLI	SKR	K-SVD	CR	Ours
Bedroom (PSNR)	80%	46.58	47.48	50.89	38.78	**52.16**
	70%	44.68	46.56	48.31	42.45	**49.41**
	60%	43.14	44.66	46.48	41.71	**47.15**
	20%	37.70	**38.73**	38.13	32.22	38.37
Bottles (PSNR)	80%	42.49	42.49	45.11	36.25	**46.88**
	70%	40.79	41.62	42.69	34.37	**44.41**
	60%	39.53	40.20	40.77	35.03	**42.46**
	20%	34.92	36.01	35.73	30.42	**36.03**
Bamboo (PSNR)	80%	38.12	40.56	41.56	34.57	**43.35**
	70%	36.60	39.12	39.17	33.21	**40.58**
	60%	35.45	36.76	37.01	32.42	**38.33**
	20%	30.60	**31.77**	31.43	26.41	30.81
Bedroom (Timing)	80%	17	432	1720	-	1566
	70%	16	424	1498	-	1425
	60%	15	394	1301	-	1308
	20%	15	290	962	-	955

Table 2: PSNR and timing results (in seconds) for image plane reconstruction.

each image and test scenario. To have a fair comparison, we optimized the parameters for SKR based on the *Bedroom* scene for 70% and 20% cases. We implemented the CR algorithm using the guidelines provided in [SD11]. Since our implementation was in MATLAB, we do report timing results. In addition, due to a high sensitivity to small changes in the parameters, we used the same strategy for parameter tuning as we used for SKR. For K-SVD, we used a patch size of 12×12 and trained a 6-sparse dictionary with 288 atoms, i.e. a two-times overcomplete dictionary. Although theoretically a more overcomplete dictionary leads to better results, we found that increasing the number of atoms does not have any effect other than increasing the computational time. The same parameters are used in [AEB06, MES08].

The image quality results are summarized in Figure 7, followed by PSNR results in Table 2. The timing results are also included in Table 2. We encourage the reader to compare full resolution images included in the supplementary materials. For the cases where our method is applied to image reconstruction with $60\% - 80\%$ of pixels rendered, we observe a large image quality improvement compared to previous work. Note that PSNR is a logarithmic function of the mean-square-error. For the 20% case, our method achieves competitive results compared to K-SVD and SKR. Looking at Figure 7, we see that SKR produces blurry results with washed out textures. The CR method generated the most blurred results, even compared to DLI. Reducing the size of blur kernel slightly sharpened the images but led to noise-like artifacts. Figure 8 plots PSNR values for the *Bottles* scene using $10\% - 90\%$ of pixels and compares our method,

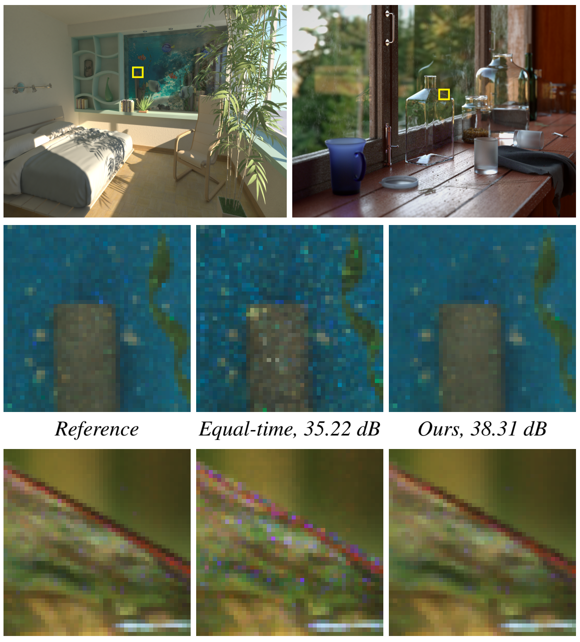


Figure 9: Comparison of our method using 60% of pixels to an equal-time rendering for the *Bamboo* scene (7 hours and 34 minutes) and the *Bottles* scene (29 hours and 10 minutes). PSNR is computed on full images.

DLI, and K-SVD. Our CS framework maintains a large improvement in PSNR over K-SVD and DLI when using 20% to 90% of pixels as input data.

The computational complexity of our algorithm is only dependent on the image resolution and the number of samples, m. Therefore, in Table 2 we report timing results only for the *Bedroom* scene. For the 70% case, the computation time for Algorithm 1 is 437 seconds, and 972 seconds for Smoothed-ℓ_0. Considering the *Bottles* scene for the 20%, 60%, 70% and 80% cases, the rendering time saved using our method is about 38, 18, 14, and 9 hours, respectively. An advantage of our method is that the reported speedup for 60%-80% cases is achieved without noticeable degradation of visual quality. In addition, our method can be used to generate preview-renderings using only a very small percentage of pixels, e.g. 20%. Moreover, since patches are reconstructed independently, the performance boost with multi-processor and multi-computer architectures grows linearly, as in conventional ray tracing.

We also compared our reconstruction results with an equal-time rendering, i.e. when the renderer generates an image in 60% of the time required to render the reference image. Figure 9 presents these results for the *Bamboo* and *Bottles* scenes. Apart from a large improvement in PSNR for our method, severe noise artifacts are evident in the equal time rendering.

Although we focused on photo-realistic rendering, our method is flexible enough to reconstruct non-photo-realistic images. Let us consider the worst-case scenario, as shown in Figure 10, where a completely synthetic image is subsampled. Reconstructing this image with the dictionary trained using natural images leads to artifacts around edges, see Figure 10c. A simple approach for solving this problem was presented in Section 3. We trained 8 dictionaries on variants of the image in Figure 10a. These new dictionaries were appended to the ensemble of 32 dictionaries we trained on natural images, leading to an ensemble with 40 dictionaries. Reconstruction result is shown in Figure 10d. Our extended dictionary ensemble has substantially reduced the artifacts around edges.

We also compared our results to state-of-the-art methods in image processing for interpolation of irregular pixel-samples. We use combined color patches of size 12×36 with different missing values for each color channel. This is the input data typically used in the image processing literature. The methods we compared to are DLI, Low Rank Tensor

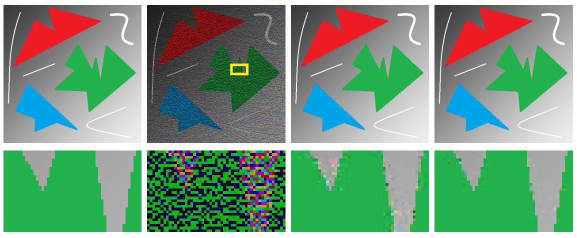


Figure 10: (a) A reference synthetic image, (b) Input image, (c) reconstruction using a dictionary ensemble trained on natural images (PSNR: 34.79dB), (d) Reconstruction using an expanded ensemble (PSNR: 36.42dB). We used 50% of pixels.

Image	Ratio	DLI	LRTC	NBDL	PLE	Ours
Castle	80%	32.64	38.22	41.51	48.26	**56.49**
	50%	28.69	31.01	36.45	38.34	**40.60**
	30%	26.86	27.90	32.02	33.01	**33.22**
	20%	25.95	26.21	29.12	30.07	**30.65**
Mushroom	80%	36.02	39.53	42.56	49.25	**54.83**
	50%	31.81	31.72	38.88	40.72	**41.69**
	30%	29.72	28.20	34.63	**35.36**	34.85
	20%	28.69	26.46	31.56	**32.06**	32.04
Train	80%	28.75	33.04	40.73	44.01	**52.94**
	50%	24.83	26.14	32.00	32.75	**35.12**
	30%	23.07	22.94	27.00	27.46	**27.56**
	20%	22.26	21.45	24.59	24.73	**25.14**
Horses	80%	33.92	38.76	41.97	48.83	**56.35**
	50%	29.89	31.23	37.27	38.52	**39.90**
	30%	28.09	28.02	31.54	**32.99**	32.75
	20%	27.30	26.48	28.81	30.26	**30.43**

Table 3: PSNR comparison for image reconstruction from irregular samples. The methods that we compare here are: DLI, LRTC, BP, PLE and our method

Completion (LRTC) [LMWY13], Nonparametric Bayesian Dictionary Learning (NBDL) [ZCP*12], and Piecewise Linear Estimator (PLE) [YSM12]. We used a standard dataset commonly used in the image processing community. For BP and PLE, the results reported in the corresponding papers are used. For LRTC, we used the reference implementation of the authors, and generated the results using the high accuracy variant of the LRTC algorithm, HiLRTC. For our method, we trained an 18-sparse ensemble of 32 dictionaries on the same training set we used before (Figure 6). Image quality results using PSNR are summarized in Table 3. Our method outperforms the state-of-the-art, PLE, by a very large margin for 50%-80% cases, and achieves competitive or better results for 20% and 30% cases.

8.2. Image Sequence Reconstruction

We applied our method for reconstructing a sequence of photo-realistically rendered images, i.e. a 3D data structure representing a video. In order to exploit the temporal coherency, we constructed patches from a small set of consecutive frames. As a result, we have $m_1 = s_1 c$ and $m_2 = s_2 * f$, where $s_1 \times s_2$ is the patch size in one image, $c = 3$ is the number of color channels, and $f = 3$ is the number of consecutive frames. One can set a larger value for f, which allows for taking less samples; at the same time, increasing f will increase the reconstruction time and will demand a larger training set. The comparison of our method with DLI, SKR and K-SVD is shown in Figure 11, Table 4 and the accompanying video. The image sequence consists of 150 frames and was rendered in 23 hours and 41 minutes. It is evident in Figure 11 that K-SVD produces severe noise-like artifacts (in the red part of the inset) and SKR generates blurred images. Our method is considerably faster and does not have these artifacts. Note that these results were obtained on a sequence without motion blur. Adding motion blur increases coherence between frames and, consequently, improves reconstruction result.

To obtain these results, we trained a 60-sparse ensemble of 32 dictionaries ($s_t = 60$ and $k = 32$) on an image sequence of the same scene but with a completely different camera path. The 60-sparse K-SVD dictionary was trained on the same training set and with 1152 atoms (two times overcomplete). The input to our method and K-SVD is the same, but DLI and SKR were applied on each image separately (it is not clear how to extend these methods to higher dimensions). The rendering and reconstruction time for our method using 70% of the pixels was 17 hours, leading to a 6 hour and 41 minute reduction in rendering time and without noticeable degradation in visual quality. The results reported here are for non-overlapping patches. One can create overlapping patches not only for each image in the sequence, but also in the temporal domain. For instance, when we have three consecutive frames, the overlapping temporal frames will be $[1,2,3]$, $[2,3,4]$, $[3,4,5]$, etc.

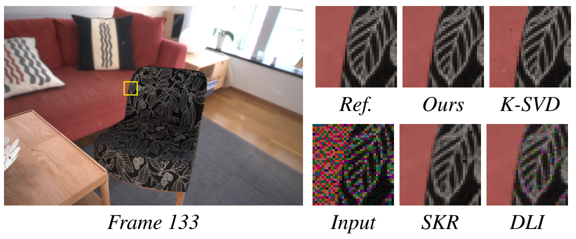


Figure 11: Reconstruction results for a single frame of an image sequence using 70% of pixels. Zoom in the digital version.

	Ratio	DLI	SKR	K-SVD	Our Method
PSNR	80%	38.67	40.21	49.45	**52.56**
	70%	36.89	37.54	46.51	**48.77**
	60%	35.60	34.27	43.99	**45.11**
Time	80%	44	2977	10856	3040
	70%	45	3025	7914	2406
	60%	48	3068	6164	1922

Table 4: PSNR and timing results (in seconds) for image sequence reconstruction.

8.3. Light-field Reconstruction

A light-field is a function defined as $f(u, v, \phi, \theta)$, where (u, v) and (ϕ, θ) define the spatial and angular domains, respectively. A discretization of this function leads to a 4D data structure. A common way to store the light-field of a scene is to capture photographs from different vantage points on a regular grid. Given such a data set, we define $m_1 = cs_1s_2$ and $m_2 = a_1a_2$, where $s_1 \times s_2$ is the patch size in one image, a_1 and a_2 represent grid dimensions (angular samples), and c is the number of color channels. This approach was used in a recent work by Marwah et al., Compressive Light-Field Photography (CLFP) [MWBR13], where authors reconstruct these patches given that the observed data only includes a single direction, i.e. one photograph. The goal is then to reconstruct remaining data in the angular domain. Such a limitation was imposed by the hardware design proposed in their paper. CLFP uses online dictionary learning and a compressed sensing framework similar to the one we used for K-SVD in Section 8.1.

The synthetic data-set used in [MWBR13] is a collection of 25 images of a rendered scene, hence $a_1 = a_2 = 5$. While CLFP uses a spatial patch size of 9×9, we divide the spatial domain into 5×5 patches ($s_1 = s_2 = 5$), leading to a light-field patch size of 75×25. We trained a 10-sparse ensemble of 32 dictionaries using the training set of [MWBR13], shown at the top of Figure 12. Results for CLFP are reported using the original implementation of the authors with the same parameters and the provided optimized dictionary. For both methods, we used binary sensing matrices as described before. Although CLFP uses Gaussian sensing matrices, we did not observe any change in PSNR or computation time using binary sensing matrices.

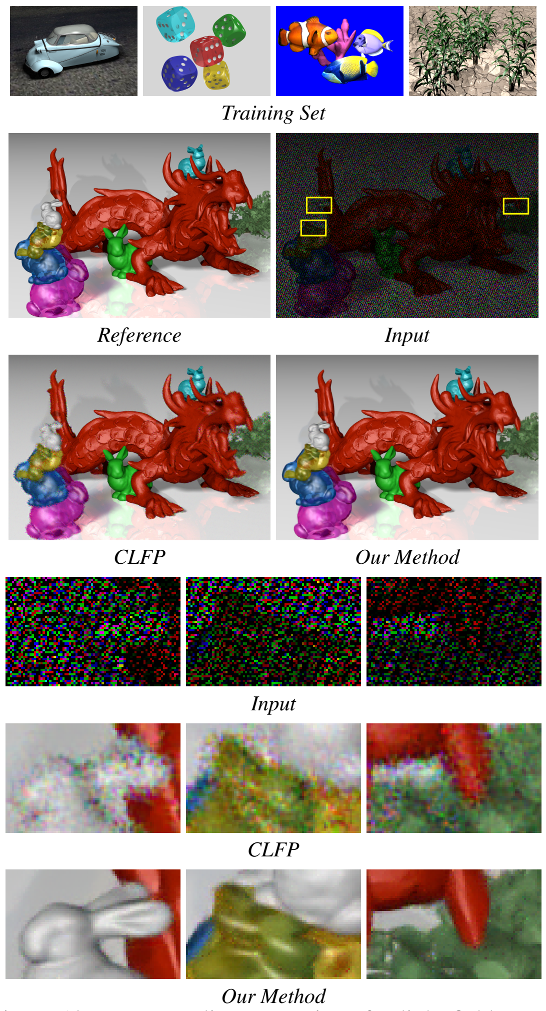


Figure 12: Image quality comparison for light-field reconstruction using overlapping patches.

Patch Config.	CLFP	Time	Our Method	Time
Non-overlapping	24.83	183	**31.17**	565
Overlapping	26.56	1907	**33.49**	2931

Table 5: PSNR and timing results for light-field reconstruction. Timing results are in seconds.

Table 5 reports the PSNR and timing results for our method and CLFP. Note that only 4% of the light-field data is available. The use of Smoothed-ℓ_0 solver already improves the results reported in [MWBR13] for non-overlapping patches by 1.83dB. We compared our results with this improved version of their method. For non-overlapping and overlapping patches, our method achieves 6.34dB and 6.93dB improvement over CLFP, respectively. Figure 12 compares a single image from the light-field reconstructed by both methods. In addition, the accompanying video contains an animated visualization of the reconstructed light-fields.

Comparing Tables 2 and 4 suggests that as the dimensionality of the signal grows, i.e. 2D to 3D, our method becomes considerably faster than K-SVD. However, looking at Table 5, we see that our method is slower for light-field data (4D) compared to the 2D and 3D cases. This is due to the fact that CLFP uses a K-SVD dictionary optimized for speed which is 1.2 times overcomplete (instead of 2) and is trained using coresets [MWBR13]. Regardless, since the parameters for CLFP are optimized and the results reported are the best achievable ones, the large improvement in image quality motivates the use of our method.

9. Discussion and Future Work

We presented an approach for reconstruction of visual data using a new compressed sensing framework based on an ensemble of dictionaries. For different applications, the dictionary ensemble is trained on a generic dataset, computed only once, and can be used in a wide variety of reconstruction scenarios. Based on the excellent performance of our method for reconstructing images, animations, and light-fields, we believe that it can also prove to be useful in other applications in computer graphics where compressed sensing has been successfully applied in the past, e.g. compressive light transport sensing [PML*09].

A limitation of our work for rendering is that if the missing pixels are computationally cheap to render, e.g. using simple scenes and coarse light transport models, our method might be slower than brute force rendering. However, since ℓ_1 reconstruction is the bottleneck, a GPU implementation of the Smoothed-ℓ_0 algorithm will greatly improve the computation time of our method. Other ℓ_1 solvers such as SPARSA [WNF09] have a GPU implementation but provide worse results in terms of reconstruction quality.

In this paper, we considered the pixels in the rendered image to be independent from each other. An interesting extension would be to instead consider sub-pixel samples. If the dictionary ensemble is trained using sub-pixel samples, it would be possible to reconstruct the final pixel values using standard smoothing filters taking into account effects such as anti-aliasing in the image plane. Extending this to capturing lens or temporal samples would enable also reconstruction of motion blur and depth-of-field effects. A benefit of our CS framework is that, for all these effects, the proposed approach can be utilized directly since only the input data for training and reconstruction is changed.

Acknowledgments

We would like to thank Vlad "SATtva" Miller and Matthew Harwood for providing us with the *Bottles* and *Bedroom* scenes, respectively. This project was funded by the Swedish Foundation for Strategic Research through grant IIS11- 0081, the Linnaeus research environment CADICS, and Linköping University Center for Industrial Information Technology.

References

[AEB06] Aharon M., Elad M., Bruckstein A.: K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing 54*, 11 (2006), 4311–4322. 3, 6, 7, 8

[BCW10] Baraniuk R., Cevher V., Wakin M.: Low-dimensional models for dimensionality reduction and signal recovery: A geometric perspective. *Proceedings of the IEEE 98*, 6 (2010), 959–971. 1, 6

[Can08] Candès E. J.: The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique 346*, 9–10 (2008), 589 – 592. 1

[CR07] Candès E., Romberg J.: Sparsity and incoherence in compressive sampling. *Inverse Problems 23*, 3 (2007), 969–985.

[CW08] Candès E., Wakin M.: An introduction to compressive sampling. *IEEE Signal Processing Magazine 25*, 2 (2008), 21–30.

[DB12] Duarte M., Baraniuk R.: Kronecker compressive sensing. *IEEE Transactions on Image Processing 21*, 2 (2012), 494–504. 4

[DCS09] Duarte-Carvajalino J., Sapiro G.: Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization. *IEEE Transactions on Image Processing 18*, 7 (2009), 1395–1408. 5

[Don06] Donoho D.: Compressed sensing. *Information Theory, IEEE Transactions on 52*, 4 (April 2006), 1289–1306. 1

[EA06] Elad M., Aharon M.: Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing 15*, 12 (2006), 3736–3745. 5, 7

[EHJT04] Efron B., Hastie T., Johnstone I., Tibshirani R.: Least angle regression. *Annals of Statistics 32*, 2 (2004), 407–499. 2

[EKB10] Eldar Y., Kuppinger P., Bolcskei H.: Block-sparse signals: Uncertainty relations and efficient recovery. *IEEE Trans. on Signal Processing 58*, 6 (2010), 3042–3054. 1, 6

[GE11] Gleichman S., Eldar Y.: Blind compressed sensing. *IEEE Transactions on Information Theory 57*, 10 (2011), 6958–6975. 7

[GRBR10] Gurumoorthy K. S., Rajwade A., Banerjee A., Rangarajan A.: A method for compact image representation using sparse matrix and tensor projections onto exemplar orthonormal bases. *IEEE Transactions on Image Processing 19*, 2 (2010), 322–334. 3

[LAC*11] Lehtinen J., Aila T., Chen J., Laine S., Durand F.: Temporal light field reconstruction for rendering distribution effects. *ACM Transactions on Graphics (Proc. of SIGGRAPH) 30*, 4 (2011), 55:1–55:12. 7

[LMWY13] Liu J., Musialski P., Wonka P., Ye J.: Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*, 1 (2013), 208–220. 10

[MBZJ09] Mohimani H., Babaie-Zadeh M., Jutten C.: A fast approach for overcomplete sparse decomposition based on smoothed ℓ_0 norm. *IEEE Transactions on Signal Processing 57*, 1 (2009), 289–301. 2, 7

[MES08] Mairal J., Elad M., Sapiro G.: Sparse representation for color image restoration. *IEEE Transactions on Image Processing 17*, 1 (2008), 53–69. 8

[MKU13] Miandji E., Kronander J., Unger J.: Learning based compression of surface light fields for real-time rendering of global illumination scenes. In *SIGGRAPH Asia Technical Briefs* (2013), pp. 24:1–24:4. 3

[MWBR13] Marwah K., Wetzstein G., Bando Y., Raskar R.: Compressive light field photography using overcomplete dictionaries and optimized projections. *ACM Transactions on Graphics (Proc. of SIGGRAPH) 32*, 4 (2013), 46:1–46:12. 1, 3, 7, 10, 11

[NV09] Needell D., Vershynin R.: Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Foundations of Computational Mathematics 9*, 3 (2009), 317–334.

[ODR09] Overbeck R. S., Donner C., Ramamoorthi R.: Adaptive wavelet rendering. *ACM Transactions on Graphics (Proc. of SIGGRAPH Asia) 28*, 5 (2009), 140:1–140:12. 7

[PML*09] Peers P., Mahajan D. K., Lamond B., Ghosh A., Matusik W., Ramamoorthi R., Debevec P.: Compressive light transport sensing. *ACM Transactions on Graphics 28*, 1 (2009), 3:1–3:18. 1, 11

[RS09] Rivenson Y., Stern A.: Compressed imaging with a separable sensing operator. *IEEE Signal Processing Letters 16*, 6 (2009), 449–452. 4

[RZE10] Rubinstein R., Zibulevsky M., Elad M.: Double sparsity: Learning sparse dictionaries for sparse signal approximation. *IEEE Transactions on Signal Processing 58*, 3 (2010), 1553–1564.

[SD10] Sen P., Darabi S.: Compressive estimation for signal integration in rendering. *Computer Graphics Forum (Proc. of EGSR) 29*, 4 (2010), 1355–1363. 1

[SD11] Sen P., Darabi S.: Compressive rendering: A rendering application of compressed sensing. *IEEE Transactions on Visualization and Computer Graphics 17*, 4 (2011), 487–499. 1, 7

[SRSE11] Sprechmann P., Ramirez I., Sapiro G., Eldar Y.: C-hilasso: A collaborative hierarchical sparse modeling framework. *IEEE Transactions on Signal Processing 59*, 9 (2011), 4183–4198. 6

[TFM07] Takeda H., Farsiu S., Milanfar P.: Kernel regression for image processing and reconstruction. *IEEE Transactions on Image Processing 16*, 2 (2007), 349–366. 7

[Tib94] Tibshirani R.: Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B 58* (1994), 267–288. 4

[WNF09] Wright S., Nowak R., Figueiredo M.: Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing 57*, 7 (2009), 2479–2493. 2, 11

[YSM12] Yu G., Sapiro G., Mallat S.: Solving inverse problems with piecewise linear estimators: From gaussian mixture models to structured sparsity. *IEEE Transactions on Image Processing 21*, 5 (2012), 2481–2499. 10

[ZCP*12] Zhou M., Chen H., Paisley J., Ren L., Li L., Xing Z., Dunson D., Sapiro G., Carin L.: Nonparametric bayesian dictionary learning for analysis of noisy and incomplete images. *IEEE Transactions on Image Processing 21*, 1 (2012), 130–144. 10

[ZMRE12] Zelnik-Manor L., Rosenblum K., Eldar Y.: Dictionary optimization for block-sparse representations. *IEEE Transactions on Signal Processing 60*, 5 (2012), 2386–2395. 1, 6