EFFICIENT ALGORITHMS FOR CONVOLUTIONAL INVERSE PROBLEMS IN MULTIDIMENSIONAL IMAGING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

DİDEM DOĞAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

FEBRUARY 2020

Approval of the thesis:

EFFICIENT ALGORITHMS FOR CONVOLUTIONAL INVERSE PROBLEMS IN MULTIDIMENSIONAL IMAGING

submitted by **DİDEM DOĞAN** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar Dean, Graduate School of **Natural and Applied Sciences**	____________
Prof. Dr. İlkay Ulusoy Head of Department, **Electrical and Electronics Engineering**	____________
Assist. Prof. Dr. Figen S. Öktem Supervisor, **Electrical and Electronics Engineering, METU**	____________

Examining Committee Members:

Prof. Dr. A. Aydın Alatan Electrical and Electronics Engineering, METU	____________
Assist. Prof. Dr. Figen S. Öktem Electrical and Electronics Engineering, METU	____________
Prof. Dr. Gözde Bozdağı Akar Electrical and Electronics Engineering, METU	____________
Prof. Dr. Orhan Arıkan Electrical and Electronics Engineering, Bilkent	____________
Assist. Prof. Dr. Elif Vural Electrical and Electronics Engineering, METU	____________

Date:

ABSTRACT

EFFICIENT ALGORITHMS FOR CONVOLUTIONAL INVERSE PROBLEMS IN MULTIDIMENSIONAL IMAGING

Doğan, Didem
M.S., Department of Electrical and Electronics Engineering
Supervisor: Assist. Prof. Dr. Figen S. Öktem

February 2020, 136 pages

Computational imaging is the process of indirectly forming images from measurements using image reconstruction algorithms that solve inverse problems. In many inverse problems in multidimensional imaging such as spectral and depth imaging, the measurements are in the form of superimposed convolutions related to the unknown image. In this thesis, we first provide a general formulation for these problems named as *convolutional inverse problems*, and then develop fast and efficient image reconstruction algorithms that exploit sparse models in analysis and synthesis forms. These priors involve sparsifying transforms or data-adaptive dictionaries that are patch-based and convolutional. The numerical performance of the developed algorithms is evaluated for a three-dimensional image reconstruction problem in spectral imaging. The results demonstrate the superiority of the convolutional dictionary prior over others. The developed algorithms are also extended to the compressive setting with compressed convolutional measurements.

Keywords: convolutional inverse problem, image reconstruction, sparsity, sparsifying transform, patch-based dictionary, convolutional dictionary, compressive sensing

ÖZ

ÇOK BOYUTLU GÖRÜNTÜLEMEDEKİ EVRİŞİMSEL TERS PROBLEMLER İÇİN ETKİN ALGORİTMALAR

Doğan, Didem
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Dr. Öğr. Üyesi. Figen S. Öktem

Şubat 2020 , 136 sayfa

Hesaplamalı görüntülemede, ters problem çözülerek dolaylı ölçümlerden görüntü geriçatım yoluyla elde edilir. Spektral ve derinlik görüntüleme gibi çok boyutlu görüntülemedeki birçok ters problemde, ölçümler bilinmeyen görüntünün evrişimlerinin üst üste binmesinden oluşur. Bu tezde, ilk olarak bu tür problemler *evrişimsel ters problemler* adı altında genel bir çatıda ele alınmakta, ve ardından analiz ve sentez formundaki seyreklik modelleri kullanılarak hızlı ve etkin görüntü geriçatım algoritmaları geliştirilmektedir. Bu modeller seyrekleştirici dönüşümler ve veriye uyarlanır yama tabanlı evrişimsel sözlükler içermektedir. Geliştirilen algoritmalar, sıkıştırılmış ölçümlerin olduğu durumlar için de kullanılabilmektedir. Algoritmaların sayısal başarımı spektral görüntülemede karşılaşılan üç boyutlu bir görüntü geriçatım problemi için analiz edilmektedir. Sonuçlar evrişimsel sözlük modelinin görüntü başarımı açısından diğer önsellerden üstün olduğunu göstermektedir.

Anahtar Kelimeler: evrişimsel ters problem, görüntü geriçatımı, seyreklik, seyrekleştirici dönüşüm, yama-tabanlı sözlük, evrişimsel sözlük, sıkıştırmalı algılama

To my family...

ACKNOWLEDGMENTS

First and foremost, I would like to express my deep gratitude and appreciation to my supervisor Assist. Prof. Dr. Figen S. Öktem for her guidance, patience, continuous support and encouragement that made this study possible. I learned a lot from her about research and writing. I thank her for teaching me looking from a scientific perspective and being patient about the results of the experiments.

I am grateful to Prof. Dr. Orhan Arıkan, Prof. Dr. Aydın Alatan, Asst. Prof. Dr. Elif Vural, and Prof. Dr. Gözde Bozdağı Akar for showing interest in my studies and accepting to review my thesis. I also would like to thank TUBITAK (The Scientific and Technological Council of Turkey) for their support within the scope of 2210/A scholarship. This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under grant 117E160 (3501 Research Program).

I would like to express my deepest gratitude for my family. I would like to thank my mother Zeynep and my father Ünal for their unconditional love, caring, patience and wisdom. My sisters Funda and Ayşe and my brother Alper always supported me in all my endeavors. Finally, I am thankful to my husband Hasan Can for his love, emotional support, and being my best friend.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
ADMM	Alternating Direction Method of Multipliers
CS	Compressive Sensing
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
EUV	Extreme Ultra-Violet
MAP	Maximum a Posteriori
N/A	Not Applicable
PSNR	Peak Signal-to-Noise Ratio
PSSI	Photon Sieve Spectral Imaging
SAM	Spectral Angular Mapper
SNR	Signal-to-Noise Ratio
SSIM	Structural Similarity Index Metric
TV	Total Variation

CHAPTER 1

INTRODUCTION

Multidimensional imaging, that is, capturing image data in more than two dimensions, has been a prominent field with ubiquitous applications in the physical and life sciences [1, 2]. The multidimensional image data, including the spatial, spectral, and temporal distributions of light (or an electromagnetic field), provide unprecedented information about the chemical, physical and biological properties of targeted scenes [3–6].

While the objective of conventional photography is to measure only the two-dimensional spatial distribution of light, the objective of multidimensional imaging is to form images of a radiating scene as a function of more than two variables. That is, the goal is to obtain a data cube of high dimensions, for example, in three spatial coordinates (x, y, z), wavelength (λ) and time (t). However, obtaining this high-dimensional image data with inherently two-dimensional detectors poses intrinsic limitations on the spatio-spectral-temporal extent of these techniques.

Conventional techniques circumvent this limitation by sequential scanning of a series of two-dimensional measurements to form the high-dimensional image data. For example, in spectral imaging, the three-dimensional data cube (x, y, λ) is typically obtained by either using a spectrometer with a long slit and scanning the scene spatially, or by using an imager with a series of spectral filters and scanning the scene spectrally. As a result, these scanning-based conventional methods generally suffer from low signal-to-noise ratio (SNR), high acquisition time, and temporal artifacts for dynamic scenes. Moreover, the attainable resolutions (such as temporal, spatial, and spectral) are inherently limited by the physical components involved.

To overcome these drawbacks, computational imaging approaches have been developed to pass on some of the burden to a reconstruction algorithm [6–14]. In these approaches, image data is reconstructed by combining information from multiplexed measurements with the additional prior (statistical or structural) knowledge about the unknown image.

In many image reconstruction problems in multidimensional imaging (such as spectral and depth imaging), the measurements are in the form of superimposed convolutions. That is, the relationship between the measured (sensor) data and the unknown image can often be adequately characterized by a single convolution operation or sum of multiple convolutions (hence, the image-formation model can be approximated by a linear and shift-invariant model). In this paper, we focus on the solution of these types of inverse problems, which are called here *convolutional inverse problems*. Examples of such inverse problems include the image deconvolution problem and its variants, and the three-dimensional image reconstruction problems encountered in various spectral and depth imaging modalities [6–14].

Due to the ill-posed nature of convolutional inverse problems, sparsity priors are frequently exploited in their solutions for effective regularization. Image reconstruction methods enforce the sparsity of the image via analysis or synthesis model. In the analysis approach, discrete derivative operators or sparsifying transforms can be utilized such as discrete cosine transform (DCT), wavelets, or their Kronecker-product forms [6, 13]. In the synthesis approach, fixed dictionaries are exploited, which are analytical or learned from a dataset [7]. More recently, however, learning sparse models adaptively from the data appears to be more effective. Online learning-based techniques have been viewed both in learning sparsifying transforms [15] and dictionaries [18]. Transform learning approaches are utilized in various works since online dictionary learning-based sparse reconstruction generally have higher computational complexity for large scale problems such as spectral imaging [12] and MR imaging [17]. Dictionary learning-based techniques have been exploited in solving various inverse problems in multi-dimensional imaging including MR imaging [19, 20], 3D ultrasound imaging [24], spectral unmixing [21], denoising and inpainting [22], and superresolution [23].

In transform or dictionary learning approaches, generally, image is partitioned into overlapping patches. This is because the representation of the overall image requires huge matrix-vector multiplications, which causes substantial computational cost. Instead of representing the entire image in the transform domain or in terms of dictionary elements, each patch is independently represented. Consequently, resulting sparse codes do not represent the entire image but a local patch. To mitigate the partitioning problem, convolutional sparse model, a variant of dictionary-based sparse representations, has been developed [28–30].

Convolutional sparse models provide better sparse representations than the traditional ones as they do not require to divide the image into overlapping patches and instead enforce a global sparse model for the entire image [30]. Several works have exploited convolutional two-dimensional sparse priors in a variety of inverse problems including super-resolution [32], image fusion [33,34], image enhancement [35], multichannel imaging [36,37], multimodal imaging [38], and tomographic reconstruction [39]. In the inverse problems with a two-dimensional image of interest, two-dimensional convolutional sparse priors are enforced [32–35]. Furthermore, two-dimensional priors are also utilized in the problems with a three-dimensional image of interest, but these works enforce the correlation of the data across third-dimension via additional priors [36–39]. On the other hand, three-dimensional convolutional sparse representations attract limited attention among the signal processing community [40,41].

In this thesis, we first introduce a unified framework for the solution of convolutional inverse problems by considering a general image-formation model. We then develop fast image reconstruction algorithms that can exploit sparse models in analysis or synthesis forms. In the analysis case, discrete derivative operators or sparsifying transforms can be utilized such as discrete cosine transform (DCT), wavelets, or their Kronecker-product forms [6, 13, 16]. In the synthesis case, convolutional or patch-based dictionaries can be utilized, which can also be adapted to correlations in different dimensions [7, 19–24, 36–41]. If the image of interest is correlated in all directions, then high-dimensional transforms or dictionaries can be used in these models. Based on the available prior knowledge about the image of interest, there may be correlations either all through the image data or only in certain dimensions. The inverse problem is formulated for both cases, and the resulting optimization problems are solved via the alternating direction method of multipliers (ADMM). The obtained reconstruction algorithms have efficient and closed-form update steps. To illustrate their performance, these algorithms are applied to three-dimensional reconstruction problems in computational spectral imaging, and their performance is numerically demonstrated for various cases with or without correlation along the third dimension.

In the former case, when there are correlations along the third dimension, there are some works in the literature that exploit multidimensional (3D) priors. Since the image data possesses correlation across all dimensions, at least three-dimensional priors should be exploited for the reconstructed data cube. For example, many works impose 3D transforms for sparsification in depth and compressive spectral imaging [6, 13]. Another work enforces sparsity with 3D patch-based dictionaries in medical imaging [20]. There are also works that utilize 3D convolutional dictionaries by considering temporal or spectral correlations [40, 41]. However, these works do not include a comprehensive analysis and comparison of different multidimensional priors for the imaging problems. In our work, we aim to fulfill this need for image reconstruction problems. Furthermore, our method is different from other conventional counterparts, which enforce three-dimensional correlation via additional priors. To exemplify, a work imposes two-dimensional convolutional dictionaries but enforces correlation of the spectrum via a cross-talk matrix of different channels [36]. Another one enforces the temporal smoothness of MR images with an additional TV regularizer [39].

In the latter case, when there are no correlations along the third dimension, there are some works in the literature that exploit 2D priors. Here, only spatial correlations are taken into account as spatial slices are generally correlated in imaging applications [11, 12]. Therefore, two-dimensional sparse priors are exploited for each spatial slice of the data. Similar to us, some works use patch-based transforms for the reconstruction of uncorrelated spectral scenes [11, 12]. Another one presents multiple methods for multi-channel images and ignores correlations along the third dimension [37].

In Chapter 2, we introduce the convolutional forward problem by considering various imaging problems. These include image deconvolution, multi-frame image deconvolution, and examples from spectral and depth imaging. The convolutional inverse problem is also formulated with various priors involving sparsifying transform, patch-based dictionary, convolutional dictionary, and convolutional dictionary with Tikhonov regularization.

In Chapter 3, we develop image reconstruction methods for the presented priors. In the first algorithm, fixed transforms are exploited for convolutional inverse problems as fixed-transforms provide fast-solutions. Second, we develop an image reconstruction algorithm exploiting patch-based sparse representations. An extension with online dictionary learning is also included to improve image reconstruction quality. Third, we develop an image reconstruction method with the convolutional dictionary. This algorithm also involves online dictionary learning. Lastly, we present an extension of the convolutional sparse prior with Tikhonov regularization. These methods are presented for three-dimensional image reconstruction problems for various cases with or without correlation along the third-dimension. Moreover, all methods are based on the alternating direction method of multipliers (ADMM). By solving several steps of the problem in the frequency domain, fast and high-quality reconstructions are enabled.

In Chapter 4, the numerical performance of the developed algorithms is evaluated for a computational spectral imaging problem. Firstly, a discrete spectrum case is considered. In this case, the reconstructed images are not correlated along the third dimension, and hence two-dimensional priors are exploited. Furthermore, we also present a comprehensive analysis of dictionaries with respect to various dictionary size and dictionary training set. We also compare the performance of different priors. Secondly, the numerical performance of the developed algorithms is evaluated for the continuous spectrum case. Here, the reconstructed images are correlated along the third dimension, and hence three-dimensional priors are enforced. We also optimize the dictionary size to attain the best performance among different dictionaries.

In Chapter 5, convolutional inverse problems with the compressive setting are introduced. Here, the entire three-dimensional data cube is reconstructed from highly compressed coded measurements. First, the compressive forward problem is presented for a general imaging problem. The inverse problem is formulated for this imaging modality using the analysis and synthesis priors as before. Then, the algorithms for the compressive image reconstruction problem is presented. Lastly, we numerically demonstrate the performance of these algorithms for different number of measurements and different designs of the imaging system.

In Chapter 6, we conclude the thesis and discuss future work.

CHAPTER 2

CONVOLUTIONAL INVERSE PROBLEMS

2.1 Introduction

In this chapter, we model convolutional inverse problems with a three-dimensional image of interest. Examples of these problems include image deconvolution, multi-frame image deconvolution, and three-dimensional image reconstruction problems encountered in various computational and depth imaging modalities [6–14]. These problems are presented from simplest to more complex. Then, we develop a general formulation of convolutional inverse problems and introduce three and two-dimensional sparse priors. We utilize both analysis and synthesis priors for the regularization, considering their merits and drawbacks. These priors involve a sparsifying transform, patch-based dictionary, and convolutional dictionary. We formulate the inverse problem with these priors for various scenarios, including online/offline learning. Furthermore, we also extended the convolutional dictionary with Tikhonov regularization.

2.2 Convolutional Forward Problem

In various imaging problems, the measurements can be modeled in the following general convolutional form:

$$y_k[n_1, n_2] = \sum_{s=1}^{S} x_s[n_1, n_2] * h_{k,s}[n_1, n_2] + w_k[n_1, n_2], \quad 1 \leq k \leq K \tag{2.1}$$

where n_1,n_2 = $-N/2, ..., N/2 - 1$. Here y_k's denote different measurements, x_s's represent different unknown images to be reconstructed (which denotes sth slice of the data cube across third dimension), $h_{k,s}$ denotes the blur acting on the sth image, x_s, for the kth measurement, y_k. This is a general form encountered in many different image reconstruction problems such as classical image deconvolution ($K = 1$, $S = 1$), multi-frame image deconvolution ($K > 1$, $S = 1$), and different three-dimensional image reconstruction problems in spectral [12] and depth imaging [6] ($K \geq 1$, $S > 1$). In spectral imaging, x_s corresponds to sth spectral band extracted from the spectral data cube, and hence each measurement represents superimposed convolutions of blurs with spectral bands. In depth imaging, each slice represents spatial image of sth depth from the data cube. Here measurements are the sum of convolutions of spatial images and blurs. Convolution can be represented as multiplication of circular convolution matrix and a vectorized image. With this, the following matrix-vector form is obtained from the above image-formation model as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{w} \tag{2.2}$$

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_{1,1} & \cdots & \mathbf{H}_{1,S} \\ \vdots & \ddots & \vdots \\ \mathbf{H}_{K,1} & \cdots & \mathbf{H}_{K,S} \end{pmatrix}, \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ : \\ \mathbf{y}_K \end{bmatrix}, \mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ : \\ \mathbf{x}_S \end{bmatrix} \text{ and } \mathbf{w} = \begin{bmatrix} \mathbf{w}_1 \\ : \\ \mathbf{w}_K \end{bmatrix}.$$

Here $\mathbf{y}_k \in \mathbb{R}^{N^2}$ is lexicographically ordered noisy kth measurement vector, $\mathbf{y}$ is the overall noisy measurement vector by vertically concatenating all the K measurements. Similarly, $\mathbf{x}_s \in \mathbb{R}^{N^2}$ denotes sth unknown image and the vector $\mathbf{x}$ is obtained by combining unknown images. Here, the matrix $\mathbf{H}_{k,s} \in \mathbb{R}^{N^2 \times N^2}$ is a circular convolution matrix which corresponds to the convolution operation with $h_{k,s}[n_1, n_2]$. $\mathbf{w}_k$ represents additive white Gaussian noise vector $\mathbf{w}_k \sim N(0, \sigma_k^2)$.

Image vector $\mathbf{x} \in \mathbb{R}^{N^2 S}$ will be considered for two cases. In the former case, there are correlations all through the image data. Here, $\mathbf{x}$ represents vectorized data cube $x[n_1, n_2, s]$. Hence three-dimensional analysis and synthesis priors are exploited for the image reconstruction. In the latter scenario, there are correlations only in certain dimensions for many imaging applications [11, 12, 14]. Therefore, $\mathbf{x}$ is behaved as a concatenation of separate spatial images $x_s[n_1, n_2]$ for $s = 1, 2, ..., S$, and hence two-dimensional analysis and synthesis priors are enforced.

2.3 Sample Imaging Problems

In this section, we present various problems including the form expressed in Eq. (2.1), from simplest to more complex.

2.3.1 Image Deconvolution (Deblurring)

This is a classical image deconvolution problem: A blurred image $\mathbf{Hx}$ is measured in the presence of an additive zero-mean white and homogenous Gaussian noise $\mathbf{w}$. The measured image $\mathbf{y}$ is denoted as

$$\mathbf{y} = \mathbf{Hx} + \mathbf{w}. \tag{2.3}$$

Note that, here, $\mathbf{H}$ matrix in the inverse problem corresponds to a single convolution matrix. Hence, it indeed represents a convolution of the blur function $h[n_1, n_2]$ with $x[n_1, n_2]$ for two-dimensional image data. This problem is viewed as the simplest version of convolutional inverse problems and can be extendible to higher dimensions.

2.3.2 Multi-frame image deconvolution

Here, we address the problem of restoring an image from its noisy convolutions with two or more blur functions.

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x} + \mathbf{w_k}. \tag{2.4}$$

It is a type of convolutional image reconstruction problems [10]. Each $\mathbf{H}_k\mathbf{x}$ multiplication in the inverse problem represent convolutions of the blur function $h_k[n_1, n_2]$ with $x[n_1, n_2]$. Note that this problem is also extendible to higher dimensions.

2.3.3 Computational Multi-Spectral Imaging

The problem formulation expressed in Eq. (2.1) is also encountered in various spectral image reconstruction problems. In this thesis, we focus on imaging with diffractive lenses, including "Photon Sieve Spectral Imaging" [11, 12].

2.3.3.1 Spectral Imaging with Diffractive Lenses

A sample convolutional inverse problem is spectral imaging with diffractive lenses, including photon sieve spectral imaging (PSSI). This technique uses an optical configuration that consists of a single diffractive imaging element, e.g., photon sieve [11, 12]. (Photon-sieve provides an alternative to lenses and mirrors and is obtained by replacing the open zones of Fresnel zone plates with circular holes.) The system structure is displayed in Fig. 2.1. We consider polychromatic radiation from the object, which is composed of S wavelength components, each with a different wavelength $\lambda_s(s = 1, 2, ..., S)$ and mutually incoherent from others. Since the focal length of the photon depends on the wavelength, each wavelength is focused at a different distance from the diffractive lens. Each measurement is taken at these distances, which are displayed in Fig. 2.1 as kth measurement plane. In each measurement plane, measurements consist of the superposition of blurred images. For example, a measurement is taken from a plane where one spectral component is in focus, the focused image of this spectral component overlaps with the defocused images of all other components. A total of K such measurements are obtained by using a moving detector. The image formation model that relates the intensities of individual spectral components to the measurements is the same with Eq. (2.1).

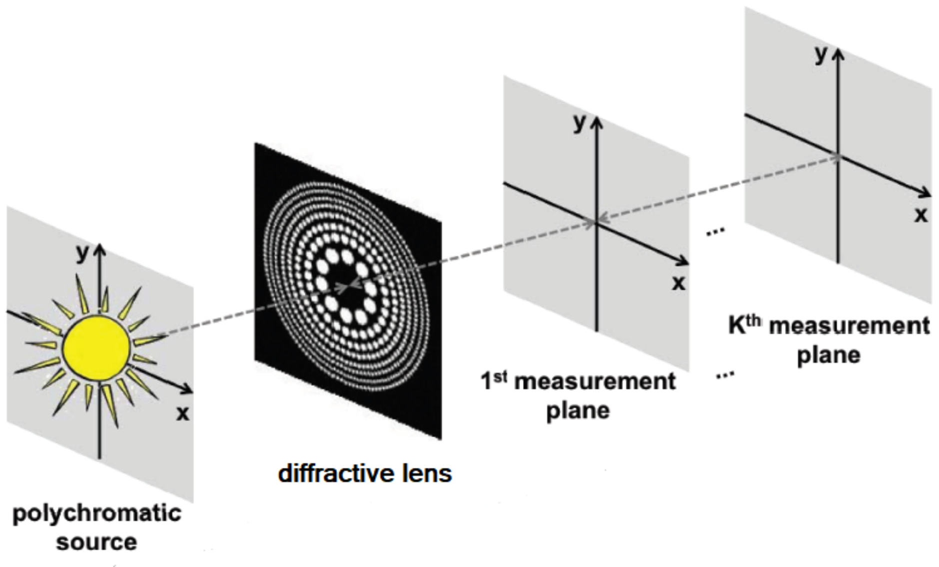


Figure 2.1: Diffractive Imaging System

2.3.4 Computational Depth Imaging

Convolutional inverse problems are also encountered in many computational depth imaging problems. For example, the DiffuserCam system is introduced in [6], which entails the sum of the convolution structure.

2.3.4.1 DiffuserCam

DiffuserCam is part of the class of the mask-based passive lensless imagers in which a phase or amplitude mask is placed a small distance in front of a sensor. The mask consists of a transparent phase object with smoothly varying thickness. The point spread functions, vary with the positions of the source, thereby encoding 3D information. Here, each 3D position in the volume generates a unique point spread function (PSF). By assuming that all points in the scene are incoherent with each other, the measurement can be modeled as convolutions of PSFs from different 3D positions as

$$y_k[n_1, n_2] = \mathbf{C} \sum_{s=1}^{S} x[-n_1, -n_2; s] * h_k[n_1, n_2; s] + w_k[n_1, n_2]. \tag{2.5}$$

where $h_k[n_1, n_2; s]$ is a blur function in each different depth. $x[n_1, n_2; s]$ represent the point at pixel (n_1, n_2) and depth s and $x[-n_1, -n_2; s]$ can be obtained by flipping the image slices. Here, C denotes cropping operation. If the crop operation is not considered, PSF wraps around the opposite side of the sensor, due to circular boundary conditions. This is explained in [6] in detail. C is a diagonal matrix and can be handled with variable splitting. Therefore, it does not disturb the "convolutional inverse problem" structure. Consequently, image reconstruction methods that are developed to solve convolutional inverse problems are applicable for this problem as well. The DiffuserCam system is displayed in Fig. 2.2. Each measurement y_k is taken at different planes and 3D reconstructions are obtained with computational image reconstruction.

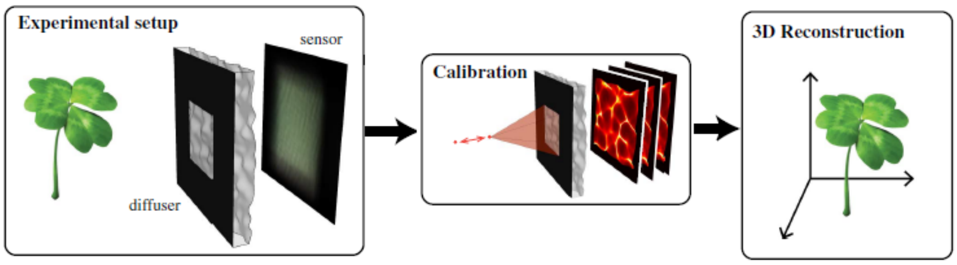


Figure 2.2: DiffuserCam Depth Imaging System

2.4 Convolutional Inverse Problem

In the inverse problem, the goal is to recover the unknown images, $\mathbf{x}$, from their noisy, superimposed, and blurred measurements, $\mathbf{y}$. This problem can be viewed as a type of multi-frame deconvolution problem involving multiple images. This deconvolution problem is inherently ill-posed, due to dependency between columns of $\mathbf{H}$. Therefore, it is necessary to replace the original ill-posed problem with another inverse problem providing better conditioning to the original problem.

Various approaches have been proposed to solve ill-posed linear inverse problems. A most common approach is regularization with the minimization of an appropriately formulated cost function. This approach derives from the use of prior knowledge concerning the unknown solution in the least-squares setting. The prior information can be incorporated in a deterministic way [43–45], or in a statistical setting [46], which is related to the Bayes paradigm of [47].

By incorporating the prior information available for the images, we formulate the problem as follows:

$$\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \mathcal{R}(\mathbf{x}) \tag{2.6}$$

This is a regularized least squares problem, which can also be related to maximum posterior estimation (MAP). Here the first term controls data fidelity, whereas the second term $\mathcal{R}(\mathbf{x})$ controls how well the reconstruction matches our prior knowledge of the solution, with the scalar parameter β trading off between these two terms.

Sparse models play an important role in computational imaging due to a significant degree of structure and redundancy present in the high-dimensional data [2]. The sparsity of natural multidimensional image data in another domain is enforced via $\mathcal{R}(\mathbf{x})$ to solve the inverse problem. Since natural multidimensional image data can be represented sparsely in some transform domain or in terms of dictionary elements, sparsity-based regularization is expressed in terms of ℓ_0-norm. Since this leads to an NP-hard problem, by convex relaxation of the ℓ_0 quasi-norm with the ℓ_1-norm, the problem can be converted to a convex optimization problem, which can be solved using standard convex optimization techniques [48, 54]. In the scope of the present work, both analysis and synthesis priors are exploited for $\mathcal{R}(\mathbf{x})$.

2.4.1 Analysis Prior

In this case, we use the fact that natural multidimensional image data is sparsifiable by analytical transforms such as wavelets, discrete cosine transform (DCT) and finite differences. In other words, the vectorized image is analyzed as $\mathbf{Tx}$ and analysis prior is represented as

$$\mathcal{R}(\mathbf{x}) = \mathbf{\Phi}(\mathbf{Tx}) \tag{2.7}$$

where $\mathbf{T}$ represents a fixed sparsifying transform matrix. There are popular and powerful choices of the regularizer $\mathbf{\Phi}(.)$. One popular choice is $\mathbf{\Phi}(\mathbf{Tx}) = ||\mathbf{Tx}||_1$, whose solution is obtained by nonlinear optimization techniques. Another powerful choice is isotropic TV, $\mathbf{\Phi}(\mathbf{Tx})$ = TV(x) with $\mathbf{T} = \mathbf{I}$ which requires iterative method for the solution [52]. Here, TV is defined as follows:

$$TV(\mathbf{x}) = \sum_{i,j} \nabla(\mathbf{x})[i,j]. \tag{2.8}$$

where

$$\nabla(\mathbf{x})[i,j] = \sqrt{(D_h(x))^2 + (D_v(x))^2} \tag{2.9}$$

and

$$D_h(x) = \mathbf{x}[i+1,j] - \mathbf{x}[i,j], \tag{2.10}$$

$$D_h(x) = \mathbf{x}[i,j+1] - \mathbf{x}[i,j], \tag{2.11}$$

Since analytical sparsifying transforms provide cost-efficient solutions, they are popular in signal and image reconstruction problems. Namely, instead of performing matrix-vector multiplications, analytical transforms such as discrete cosine transform (DCT), wavelets, or their Kronecker-product forms sparsify the images with fast transform operators in the implementation. Recently, learning sparse models adaptively from the data appeared to be more effective, especially the learning of sparsifying transforms [15, 17]. However, several works have shown that learning sparsifying transforms is a computationally-intensive process which does not bring significant performance improvement in image reconstruction problems [12]. Furthermore, it requires partitioning the image into overlapping patches, which results in that sparsifying transforms do not represent the entire image but a local patch [12]. Another drawback is that the learning of transforms eliminates fast implementation advantages of analytical transforms. Hence, small performance enhancement gained via transform-learning is dispelled by the excessive computational complexity. By considering the drawbacks of transform-learning, we prefer using approaches with fixed and analytical transforms for the solution of convolutional inverse problems.

2.4.2 Synthesis Prior with Patch-Based Dictionary

In image reconstruction problems, usage of synthesis prior is a popular approach [18–20]. Here, the unknown vector $\mathbf{x}$ can be sparsely represented as a linear combination of small number of columns from a synthesis dictionary as $\mathbf{Dz} = \mathbf{x}$, where $\mathbf{D}$ denotes the dictionary matrix and $\mathbf{z}$ stands for the sparse code. Note that $\mathbf{D}$ is not necessarily the inverse of $\mathbf{T}$, as it can be an overcomplete dictionary. Therefore, synthesis prior has the following generalized expression as

$$\mathcal{R}(\mathbf{x}) = \lambda||\mathbf{z}||_1 \text{ s.t. } \quad \mathbf{Dz} = \mathbf{x} \tag{2.12}$$

This representation requires inserting $\mathbf{Dz}$ into the data-fidelity term as $||\mathbf{y} - \mathbf{HDz}||_2^2$, which causes huge computational cost. In this work, however, we allow sparse approximation as $\mathbf{Dz} \approx \mathbf{x}$ and represent the synthesis prior as

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}} ||\mathbf{Dz} - \mathbf{x}||_2^2 + \lambda||\mathbf{z}||_1 \tag{2.13}$$

in [30, 55, 56]. In image representation problems, however, due to the increasing size of the image vectors, finding a representation for the entire image is computationally inefficient. Hence, image is partitioned into overlapping patches that are multiplication of common local dictionary and local sparse codes. Since images are of high-size, we use the patch-based version of the prior in Eq. (2.13). As mentioned before, we focus on three-dimensional data for various cases with or without correlation along the third-dimension. First, the synthesis prior with patch-based dictionary is expressed as

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_j} \frac{1}{2} \sum_{j=1}^{SN^2} ||\mathbf{D}_L \mathbf{z}_j - \mathbf{P}_j \mathbf{x}||_2^2 + \lambda \sum_{j=1}^{SN^2} ||\mathbf{z}_j||_1. \tag{2.14}$$

for three-dimensional data which is correlated along all dimensions. Here, $\mathbf{P}_j \in \mathbb{R}^{n^2 p \times N^2 S}$ is patch-extractor matrix, which extracts the patch vector of size $n^2 p$ from image vector $\mathbf{x} \in \mathbb{R}^{N^2 S}$. Here, $N^2 S$ patches are extracted. $\mathbf{D}_L \in \mathbb{C}^{n^2 p \times n^2 p}$ is a local dictionary which is used for all patches. $\mathbf{z}_j \in \mathbb{C}^{n^2 p}$, sparse representation vector which belongs to jth patch vector of the data cube. Similarly, the synthesis prior with patch-based dictionary is expressed as

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_{s,j}} \frac{1}{2} \sum_{j=1}^{N^2} \sum_{s=1}^{S} ||\mathbf{D}_L \mathbf{z}_{s,j} - \mathbf{P}_j \mathbf{x}_s||_2^2 + \lambda \sum_{j=1}^{N^2} \sum_{s=1}^{S} ||\mathbf{z}_{s,j}||_1 \tag{2.15}$$

for three-dimensional data which is correlated only in spatial dimensions. Here $\mathbf{P}_j \in \mathbb{R}^{n^2 \times N^2}$ is patch-extractor matrix, which extracts $n \times n$ sized patch from image vector $\mathbf{x}_s \in \mathbb{R}^{N^2}$. Here, SN^2 patches are extracted in total. $\mathbf{D}_L \in \mathbb{C}^{n \times n}$ is a local dictionary which is used for all patches. $\mathbf{z}_{s,j} \in \mathbb{C}^n$, sparse representation vector which belongs to jth patch vector of sth image.

In the scope of the present work, we also extend Eq. (2.14) and (2.15) for online dictionary learning. Here, the underlying dictionary is assumed unknown, unlike in Eq. (2.14). This allows the sparse model to be adapted to the specific objects being imaged. Our goal is simultaneously to enforce sparsity of reconstructed multidimensional images in an adaptive dictionary framework and obtain reconstructions that are consistent with the measurements. For this, Eq. (2.14) and (2.15) are modified including online dictionary learning and exploited in the place of regularization term $\mathcal{R}(\mathbf{x})$ as

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_j, \mathbf{D}_L} \frac{1}{2} \sum_{j=1}^{SN^2} ||\mathbf{D}_L \mathbf{z}_j - \mathbf{P}_j \mathbf{x}||_2^2 + \lambda \sum_{j=1}^{SN^2} ||\mathbf{z}_j||_1 \quad \text{s.t.} \quad ||\mathbf{D}_L||_2 = 1. \tag{2.16}$$

for three-dimensional data which is correlated along all dimensions and

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_{s,j},\mathbf{D}_L} \frac{1}{2}\sum_{j=1}^{N^2}\sum_{s=1}^{S} ||\mathbf{D}_L\mathbf{z}_{s,j} - \mathbf{P}_j\mathbf{x}_s||_2^2 + \lambda \sum_{j=1}^{N^2}\sum_{s=1}^{S} ||\mathbf{z}_{s,j}||_1 \quad \text{s.t.} \quad ||\mathbf{D}_L||_2 = 1 \tag{2.17}$$

for three-dimensional data which is correlated only in spatial dimensions. Here, the constraint on the norms of dictionary $\mathbf{D}_L$ is required to avoid the scaling ambiguity between dictionary and sparse codes.

The patch-based sparse model has some drawbacks. It is disadvantageous in terms of image reconstruction time, which complicates its usage in multidimensional image reconstruction problems. Another drawback of learning patch-based dictionaries is the requirement of partitioning the images into overlapping patches. Here, the consistency of pixels in the overlapped patches is ignored. Hence, the representation for each patch is obtained independently. As a result, learned features contain translated versions of each other, and latent interactions of the underlying signal is not grasped well [32]. Hence, resulting sparse codes do not represent the entire image but a local patch.

2.4.3 Synthesis Prior with Convolutional Dictionary

Convolutional models provide better sparse representations than the traditional ones as they do not require to divide the image into overlapping patches and instead enforce a global sparse model for the entire image [30]. The convolutional model intrinsically utilizes the consistency between pixels; therefore, it provides a global representation for the entire image without partitioning. Convolutional sparse representations model an image as a superimposed circular convolution of dictionary filters and sparse codes as

$$\sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m \approx \mathbf{x} \tag{2.18}$$

where $\mathbf{d}_m$ dictionary filter and $\mathbf{z}_m$ and sparse codes respectively for $m = 1, 2, ..., M$ [28, 30].

Convolutional sparse representations in Eq. (2.18) can be utilized as fixed dictionary-based regularizers in the place of $\mathcal{R}(\mathbf{x})$ as

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_m} \frac{1}{2}|| \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m - \mathbf{x}||_2^2 + \lambda \sum_{m=1}^{M} ||\mathbf{z}_m||_1 \tag{2.19}$$

for $\mathbf{x}$ represents vectorized three-dimensional image when the correlation of images across third-dimension plays a significant role. Here, $\mathbf{z}_m \in \mathbb{R}^{N^2 S}$ and $\mathbf{d}_m \in \mathbb{R}^{L^2 R}$. Note that N denotes image size in first and second dimension while S represents the image size in the third dimension. Similarly, L denotes dictionary size in first and second dimension while R represents the dictionary size in the third dimension. Here, the choice of the dictionary size is significantly smaller than the image size as $L << N$ and $R << S$. On the other hand, when spatial slices $\mathbf{x}_s$ are uncorrelated for $s = 1, 2, ..., S$, each $\mathbf{x}_s$ is behaved separately and sparse regularizer $\mathcal{R}(\mathbf{x})$ is expressed as

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_{s,m}} \frac{1}{2} \sum_{s=1}^{S} || \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_{s,m} - \mathbf{x}_s||_2^2 + \lambda \sum_{s=1}^{S} \sum_{m=1}^{M} ||\mathbf{z}_{s,m}||_1 \tag{2.20}$$

where $\mathbf{z}_{s,m} \in \mathbb{R}^{N^2}$ and $\mathbf{d}_m \in \mathbb{R}^{L^2}$. Note that a common dictionary set $\mathbf{d}_m$ is exploited for S images.

Note that the convolutional approach is a special case of the traditional model since the sum of convolutions is viewed as concatenated matrix-vector multiplication. The pursuit problem in Eq. (2.13) is viewed as a convolutional sparse representation problem when $\mathbf{D}$ has the structure of the concatenation of circular convolution matrices, and hence, is equivalent to Eq. (2.19) or (2.20) as proposed in [59]. Together, this structure enables efficient computation without dividing the image into overlapping patches.

In the scope of the present work, we also extend Eq. (2.19) for online dictionary learning. Here, the underlying dictionary is assumed unknown unlike in Eq. (2.19). For this, Eq. (2.19) is modified including online dictionary learning and exploited in the place of regularization term $\mathcal{R}(\mathbf{x})$ as

$$\begin{aligned} \mathcal{R}(\mathbf{x}) = &\min_{\mathbf{z}_m, \mathbf{d}_m} \frac{1}{2}|| \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m - \mathbf{x}||_2^2 + \lambda \sum_{m=1}^{M} ||\mathbf{z}_m||_1 \\ &\text{s.t.} \quad ||\mathbf{d}_m||_2 = 1 \quad m = 1, 2..M \end{aligned} \tag{2.21}$$

Here, the constraint on the norms of dictionary $\mathbf{d}_m$ is required to avoid the scaling ambiguity between dictionary and sparse codes. To efficiently solve the inverse problem in the frequency domain, an implicit zero-padding of $\mathbf{d}_m$'s to the size of the sparse codes $\mathbf{z}_m$'s is required. Explicit zero-padding operation is mathematically expressed as multiplication by $\mathbf{Q}$. By including zero-padding operation in the constraint set of the $\mathcal{R}(\mathbf{x})$ given in Eq. (2.21) is obtained as

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_m, \mathbf{d}_m} \frac{1}{2}||\sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m - \mathbf{x}||_2^2 + \lambda \sum_{m=1}^{M} ||\mathbf{z}_m||_1 \quad s.t. \quad \mathbf{d}_m \in S_d \quad \forall m \tag{2.22}$$

where

$$S_d = \{\mathbf{u} \in \mathbb{R}^{N^2} : (\mathbf{I} - \mathbf{Q}\mathbf{Q}^{\mathbf{T}})\mathbf{u} = 0, \quad ||\mathbf{u}||_2 = 1\}. \tag{2.23}$$

Same discussion is also valid for Eq. (2.20), but it is skipped for the sake of the brevity.

2.4.3.1 Convolutional Dictionary with Tikhonov Regularization

Several works have stated that convolutional dictionaries perform better for high-frequency images since it perfectly reconstructs the edges while creating some artifacts for smooth image components [30, 37, 65]. Since convolutional representations do not provide a good representation of the low-frequency components of an image, various approaches have been proposed to utilize convolutional dictionaries with high-pass filtered images. In many works, the inverse problem is reformulated by adding separate priors for the high-frequency and low-frequency components of the image [39, 41, 65]. For example, convolutional sparse prior is enforced for the high-frequency component, and a smoothing regularizer is exploited for the low-frequency component [41]. However, this approach is computationally intensive, as it requires further parameter optimizations. In some works, the problem formulation is inherently modified to enforce convolutional sparse representations for the high-frequency component without separating the image into high and low-frequency components [64].

Here, we enforce gradient minimization of sparse codes in this extended formulation. In order to eliminate artifacts from the images, smoothing the sparse codes $\{\mathbf{z}_m\}_{m=1}^{M}$ can be considered as an effective approach. Smoother sparse coefficients introduce fewer artifacts on the reconstructed images. The intuition arises from the assumption that high-frequency components $\mathbf{x}_{HPF}$ are better represented with convolutional sparse representations as

$$\mathbf{g} * \mathbf{x} = \mathbf{x}_{HPF} = \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m \tag{2.24}$$

where g represents a FIR invertible high-pass filter. By convolving right and left hand sides with g^{-1} we obtain original image $\mathbf{x}$ as follows:

$$\mathbf{x} = \mathbf{g}^{-1} * \mathbf{g} * \mathbf{x} = \mathbf{g}^{-1} * \mathbf{x}_{HPF} = \mathbf{g}^{-1} * (\sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m) = \sum_{m=1}^{M} \mathbf{d}_m * (\mathbf{g}^{-1} * \mathbf{z}_m). \tag{2.25}$$

Here, as both left and right sides are convolved by $\mathbf{g}^{-1}$, this operation corresponds to convolving $\{\mathbf{z}_m\}_{m=1}^{M}$ with a low-pass filter. To acquire low-pass filtered $\{\mathbf{z}_m\}_{m=1}^{M}$, smoothing with Tikhonov gradient regularization is considered as an effective approach. This method is proposed in [64] and expressed for three-dimensional dictionaries as

$$\begin{aligned} \mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_m} \frac{1}{2} || \sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m - \mathbf{x} ||_2^2 + \lambda \sum_{m=1}^{M} ||\mathbf{z}_m||_1 + \\ \frac{\mu}{2} \sum_{m=1}^{M} (||\mathbf{r}_1 * \mathbf{z}_m||_2^2 + ||\mathbf{r}_2 * \mathbf{z}_m||_2^2 + ||\mathbf{r}_3 * \mathbf{z}_m||_2^2) \end{aligned} \tag{2.26}$$

where $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ are filters that compute gradient in first, second and third dimensions. Similar representations are also derived for 2D convolutional priors. We can also obtain the formulations with dictionary learning by adding Gradient minimization terms to Eq. (2.22).

CHAPTER 3

IMAGE RECONSTRUCTION METHODS WITH DIFFERENT PRIORS

3.1 Introduction

In this chapter, we develop fast and efficient image reconstruction algorithms that exploit sparse models in analysis and synthesis forms. These priors involve sparsifying transforms or data-adaptive dictionaries that are patch-based and convolutional. These methods are presented for three-dimensional image reconstruction problems for two different cases with or without correlation along the third dimension. In the first case, three-dimensional transforms and dictionaries are exploited for the reconstructed data cube. In the latter case, however, each image is separately sparsified by two-dimensional transforms and dictionaries. These cases are called 3D and 2D cases in the rest of the thesis.

The presented methods are solved via the alternating direction method of multipliers (ADMM). ADMM is an optimization algorithm that is used in many signal and image reconstruction problems [32–37,40,41]. ADMM solves distributed unconstrained optimization problems by splitting them into sub-problems, which are solved alternatingly. By solving several steps of the problems in the frequency domain, fast and high-quality reconstructions are enabled for each algorithm.

Some parts of this chapter have been presented in [14] and accepted for publication [50] and in preparation for [51].

3.2 Image Reconstruction Method with Sparsifying Transforms

To develop the image reconstruction method, we first recall analysis prior in Eq. (2.7) as

$$\mathcal{R}(\mathbf{x}) = \mathbf{\Phi}(\mathbf{T}\mathbf{x}) \tag{3.1}$$

and insert this into the following equation

$$\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \mathcal{R}(\mathbf{x}). \tag{3.2}$$

Now, we obtain the objective function as follows

$$\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \lambda\Phi(\mathbf{T}\mathbf{x}). \tag{3.3}$$

The resulting optimization problem for analysis approach is solved for the unknown image $\mathbf{x}$ by using Alternating Direction Method of multipliers (ADMM) [60]. The objective function is formulated using augmented Lagrangian in ADMM framework as

$$\min_{\mathbf{x},\mathbf{t}} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \lambda\Phi(\mathbf{T}\mathbf{x}) \quad \text{s.t.} \quad \mathbf{T}\mathbf{x} = \mathbf{t}. \tag{3.4}$$

This problem is minimized with respect to variables $\mathbf{x}$ and $\mathbf{t}$. ADMM steps for this alternating minimization process have the following form:

$$\mathbf{x}^{l+1} = \arg\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \frac{\rho}{2}||\mathbf{T}\mathbf{x} - \mathbf{t}^l + \mathbf{u}^l||_2^2 \tag{3.5}$$

$$\mathbf{t}^{l+1} = \arg\min_{\mathbf{t}} \lambda\Phi(\mathbf{t}) + \frac{\beta}{2}||\mathbf{T}\mathbf{x}^{l+1} - \mathbf{t} + \mathbf{u}^l||_2^2. \tag{3.6}$$

$$\mathbf{u}^{l+1} = \mathbf{u}^l + \mathbf{T}\mathbf{x}^{l+1} - \mathbf{t}^{l+1}. \tag{3.7}$$

Here, $\mathbf{u}$ is defined as dual variable of $\mathbf{t}$. Eq. (3.7) represent the update of the dual variable. We now explain how to efficiently solve the problems in Eq. (3.5) and (3.6) which we refer as image update and auxiliary variable update.

3.2.1 Image Update

For image update, we need to solve the problem in Eq. (3.5). Since this problem is a least-squares problem, it has a closed-form solution as

$$\mathbf{x} = (\rho\mathbf{T}^H\mathbf{T} + \beta\mathbf{H}^H\mathbf{H})^{-1}(\beta\mathbf{H}^H\mathbf{y} + \rho\mathbf{T}^H(\mathbf{t} - \mathbf{u})). \tag{3.8}$$

where $\mathbf{T}^H\mathbf{T} = \mathbf{I}$ since $\mathbf{T}$ is assumed a unitary matrix. To reduce the computational complexity of obtaining the solution, resulting subproblem is solved in the frequency domain by exploiting the property that circulant convolution matrices are diagonalized by DFT matrix. Since $\mathbf{H}_{k,s}$ is a circular convolution matrix, it reduces to a diagonal block in the frequency domain. Here, $\mathbf{H}_{k,s} = \mathbf{F}_{2D}^H \mathbf{\Lambda}_{k,s} \mathbf{F}_{2D}$ where $\mathbf{F}_{2D} \in \mathbb{R}^{N^2 \times N^2}$ is a 2D DFT matrix. By inserting $\mathbf{H} = \tilde{\mathbf{F}}_{2D}^H \mathbf{\Lambda} \tilde{\mathbf{F}}_{2D}$, image update step can be expressed as

$$\mathbf{x} = \tilde{\mathbf{F}}_{2D}^H (\rho \mathbf{I} + \beta \mathbf{\Lambda}^H \mathbf{\Lambda})^{-1} (\beta \mathbf{\Lambda}^H \tilde{\mathbf{F}}_{2D} \mathbf{y} + \rho \tilde{\mathbf{F}}_{2D} \mathbf{T}^H (\mathbf{t} - \mathbf{u})). \tag{3.9}$$

For computation of Eq. (3.9), forming any of the matrices is not required since resulting matrices are block diagonal. Here we use the fact that the multiplication of a diagonal matrix and a vector corresponds to element-wise multiplication. Similarly, the multiplication of two diagonal matrices can be computed via element-wise multiplication. This provides huge savings for the memory as well as the computation time. We first compute $\mathbf{\Omega} = \Lambda^H \tilde{\mathbf{F}}_{2D} \mathbf{y}$ term. To obtain the diagonal of Λ, 2D Fourier transform of $h_{k,s}[n_1, n_2]$ is computed for $k = 1, 2, .., K$ and $s = 1, 2, .., S$. Next, $\tilde{\mathbf{F}}_{2D}\mathbf{y}$ also requires taking the FFT of $y_k[n_1, n_2]$ for $k = 1, ..., K$. Specifically, the operations required to form sth block of $\mathbf{\Omega}$, $\mathbf{\Omega_s} = \sum_{k=1}^{K} \mathbf{\Lambda}_{k,s}^H \mathbf{F}_{2D} \mathbf{y}_k$, corresponds to summation of element-wise multiplications. Since both Λ and $\mathbf{y}$ are not updated throughout iterations, $\mathbf{\Omega}$ is computed only one time. The second term, $\tilde{\mathbf{F}}_{2D} \mathbf{T}^H (\mathbf{t} - \mathbf{u})$, is updated in each ADMM iteration as follows: applying $\mathbf{T}^H$ corresponds to taking the inverse 3D transform of $(t[n_1, n_2, s] - u[n_1, n_2, s])$. Note that if 2D priors are used, $\mathbf{T}^H$ corresponds to taking the inverse 2D transforms of $(t_s[n_1, n_2] - u_s[n_1, n_2])$ for $s = 1, 2, ..., S$. Then, $\tilde{\mathbf{F}}_{2D}$ represents taking the 2D Fourier transform of the each spatial slice of the resulting 3D data-cube. Third, $\mathbf{\Psi} = \rho \mathbf{I} + \beta \mathbf{\Lambda}^H \mathbf{\Lambda}$, a matrix of $S \times S$ blocks, is computed. Here, each block is represented as

$$\mathbf{\Psi}_{i,j} = (\delta_{i,j} \mathbf{I} + \beta \sum_{k=1}^{K} \mathbf{\Lambda}_{i,k}^H \mathbf{\Lambda}_{k,j}) \tag{3.10}$$

where $\delta_{i,j}$ is Kronecker delta function $i, j = 1, 2, ..., S$. As $\mathbf{\Lambda}_{i,j}$'s are diagonal matrices, $\mathbf{\Psi}_{i,j}$ is also a diagonal matrix for $k = 1, 2, .., K$ and $s = 1, 2, .., S$. Therefore, each $\Psi_{i,j}$ is computed by the summation of element-wise multiplications in the frequency domain.

Efficient inversion of the $\mathbf{\Psi}$ follows from its block diagonal structure and is computed via recursive block matrix inversion approach [61]. For $S = 2$ case, this inverse can be computed as

$$\begin{bmatrix} \mathbf{\Psi}_{1,1} & \mathbf{\Psi}_{1,2} \\ \mathbf{\Psi}_{2,1} & \mathbf{\Psi}_{2,2} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{A} & \mathbf{\Psi^{-1}}_{1,1}\mathbf{\Psi}_{1,2}\mathbf{B} \\ \mathbf{B}\mathbf{\Psi}_{2,1}\mathbf{\Psi}_{1,1}^{-1} & -\mathbf{B} \end{bmatrix} \tag{3.11}$$

where $\mathbf{A} = \mathbf{\Psi}_{1,1}^{-1} - \mathbf{\Psi}_{1,1}^{-1}\mathbf{\Psi}_{1,2}\mathbf{B}\mathbf{\Psi}_{2,1}\mathbf{\Psi}_{1,1}^{-1}$ and $\mathbf{B} = -(\mathbf{\Psi}_{2,2} - \mathbf{\Psi}_{2,1}\mathbf{\Psi}_{1,1}^{-1}\mathbf{\Psi}_{1,2})^{-1}$. For $R > 2$, the overall matrix $\mathbf{\Psi}$ is partitioned into 2×2 blocks and each block is inverted recursively with Eq. (3.11). Since each $\mathbf{\Psi}_{i,j}$ is a diagonal matrix, the inversion is also element-wise. Remaining multiplications are also performed as element-wise using previous discussions. Although $\mathbf{\Psi}^{-1}$ is only computed once before the ADMM iterations and does not affect the computational complexity during the iterations, its pre-computation time is significantly reduced.

3.2.2 Auxiliary Variable Update

For auxiliary variable update we need to solve the problem in Eq. (3.6). By defining, $\mathbf{\Phi}(\mathbf{Tx}) = ||\mathbf{Tx}||_1$, Eq. (3.5) is solved via soft-thresholding as

$$\mathbf{t} = soft(\mathbf{Tx} + \mathbf{u}, \lambda/\rho) \tag{3.12}$$

Similar to previous discussion, a 3D transform of $x[n_1, n_2, s]$ is obtained which corresponds to $\mathbf{Tx}$. Note that if 2D priors are used, $\mathbf{T}$ corresponds to taking 2D transforms of $(x_s[n_1, n_2])$ for $s = 1, 2, ..., S$. Then, component-wise soft-thresholding is performed as

$$\mathbf{t} = sign(\mathbf{Tx} + \mathbf{u}) \odot max(0, |\mathbf{Tx} + \mathbf{u}| - \lambda/\rho). \tag{3.13}$$

As it is an element-wise opeation, it is performed without forming the vectors. If $\mathbf{\Phi}_{TV}(\mathbf{Tx})$ is chosen as an isotropic TV operator, $\mathbf{T} = \mathbf{I}$ is assigned, and

$$\mathbf{t} = \mathbf{\Psi}_{TV}(\mathbf{x} + \mathbf{u}) \tag{3.14}$$

is solved via Chambolle's algorithm for 3D case [52]. However, for 2D case, we need to separate the spatial slices into S components, namely $\mathbf{x}_s$ and $\mathbf{u}_s$ for $s = 1, 2, ..., S$ and update for each $\mathbf{t}_s$ as follows:

$$\mathbf{t}_s = \mathbf{\Phi}_{TV}(\mathbf{x}_s + \mathbf{u}_s) \quad for \quad s = 1, 2, .., S \tag{3.15}$$

Algorithm 1 Image Reconstruction Algorithm with Sparsifying Transform

```
Require: y: Measured image, H: Blur filter, T: Sparsifying Transform
Ensure: $x$: Reconstructed Image
1: Choose: $\lambda > 0$, $\rho > 0$, $\beta > 0$ , $\mathbf{t}^0$, $\mathbf{u}^0$
2: $(\rho\mathbf{T}^H\mathbf{T} + \beta\mathbf{H}^H\mathbf{H})^{-1}$ is computed.
3: $\beta\mathbf{H}^H\mathbf{y}$ is computed
4: repeat
5:     $\mathbf{x}^{l+1} = (\rho\mathbf{T}^H\mathbf{T} + \beta\mathbf{H}^H\mathbf{H})^{-1}(\beta\mathbf{H}^H\mathbf{y} + \rho\mathbf{T}^H(\mathbf{t} - \mathbf{u}))$
6:     $\mathbf{t}^{l+1} = \Psi_\Phi(\mathbf{T}\mathbf{x}^{l+1} - \mathbf{u}^l)$
7:     $\mathbf{u}^{l+1} = \mathbf{u}^l + \mathbf{T}\mathbf{x}^{l+1} - \mathbf{t}^{l+1}$
8: until Some convergence criterion is satisfied
```

3.2.3 ADMM Parameter Update

For the selection of suitable penalty parameter ρ, an adaptive strategy introduced in [60]

$$\rho_{l+1} = \begin{cases} \tau^{incr}\rho^l \text{ if } ||\mathbf{r}_l||_2 > \mu||\mathbf{s}_l||_2 \\ \rho^l/\tau^{decr} \text{ if } ||\mathbf{s}_l||_2 > \mu||\mathbf{r}_l||_2 \\ \rho^l \text{ otherwise} \end{cases} \tag{3.16}$$

is employed. Here, $\tau = 2$ and $\mu = 10$. $s_{l+1} = \rho(\mathbf{t}^l - \mathbf{t}^{l+1})$ and $\mathbf{r}_{l+1} = (\mathbf{T}\mathbf{x}^{l+1} - \mathbf{t}^{l+1})$ are primal and dual residuals respectively [60]. For stopping criteria, if $||\mathbf{x}^{l+1} - \mathbf{x}^l||_2/||\mathbf{x}^l||_2 < 10^{-4}$ the algorithm is stopped. An additional stopping criteria is derived from the following discussion:

$$\begin{aligned} \epsilon^{pri} &= \sqrt{n}\epsilon^{abs} + \epsilon^{rel}max(||\mathbf{T}\mathbf{x}^l||_2, ||\mathbf{t}^l||_2, 0), \\ \epsilon^{dual} &= \sqrt{n}\epsilon^{abs} + \epsilon^{rel}||\mathbf{u}^l||_2||_2 \end{aligned} \tag{3.17}$$

where size of $\mathbf{T}\mathbf{x}$ is represented via n. Here, $\epsilon^{abs} = 0$ and $\epsilon^{rel} = 10^{-3}$. By using the variables in Eq. (3.17), if $\epsilon^{pri} > r$ and $\epsilon^{dual} > s$ the algorithm is stopped.

3.3 Image Reconstruction Method with Patch-Based Dictionaries

In this section, we develop an image reconstruction method inserting patch-based 3D and 2D dictionaries with online learning [50]. By removing dictionary update steps, the algorithm without online dictionary learning can be easily obtained. We also emphasize the differences between the update steps with two and three-dimensional priors. Now we recall the regularizer with a patch-based dictionary in Eq. (2.16) given by

$$\min_{\mathbf{z}_j,\,\mathbf{D}_L} \frac{1}{2}\sum_{j=1}^{SN^2} ||\mathbf{D}_L\mathbf{z}_j - \mathbf{P}_j\mathbf{x}||_2^2 + \lambda\sum_{j=1}^{SN^2}||\mathbf{z}_j||_1 \quad \text{s.t.} \quad ||\mathbf{D}_L||_2 = 1. \tag{3.18}$$

and insert this into following equation

$$\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \mathcal{R}(\mathbf{x}). \tag{3.19}$$

Now, we obtain the objective function with patch-based dictionaries as

$$\min_{\mathbf{z}_j,\mathbf{x},\mathbf{D}_L} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \sum_{j=1}^{SN^2}(||\mathbf{D}_L\mathbf{z}_j - \mathbf{P}_j\mathbf{x}||_2^2 + ||\mathbf{z}_j||_1) \quad \text{s.t.} \quad \mathbf{D}_L \in S_D. \tag{3.20}$$

where

$$S_D = \{\mathbf{U} \in \mathbb{R}^{N^2} : ||\mathbf{U}||_2 = 1\}. \tag{3.21}$$

The resulting optimization problem is solved for the unknown images $\mathbf{x}$'s, the sparse coefficient vectors $\mathbf{z}_j$'s, and dictionary matrix $\mathbf{D}_L$'s by using Alternating Direction Method of multipliers (ADMM) [60]. To solve the resulting constrained optimization problem Eq. (3.20), it is converted into an unconstrained problem by adding the constraint to the objective function as an indicator function:

$$\min_{\mathbf{z}_j,\mathbf{x},\mathbf{D}_L} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \sum_{j=1}^{SN^2}(\frac{1}{2}||\mathbf{D}_L\mathbf{z}_j - \mathbf{P}_j\mathbf{x}||_2^2 + \lambda||\mathbf{z}_j||_1) + \iota_{\mathbf{S}D}(\mathbf{D}_L) \tag{3.22}$$

Here, auxiliary variable $\mathbf{t}_j$ for variable $\mathbf{z}_j$ and auxiliary variable $\mathbf{G}$ for variable $\mathbf{D}$ are inserted into Eq. (3.22) using variable-splitting:

$$\min_{\mathbf{z}_j,\mathbf{x},\mathbf{D}_L} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \sum_{j=1}^{SN^2}(\frac{1}{2}||\mathbf{D}_L\mathbf{z}_j - \mathbf{P}_j\mathbf{x}||_2^2 + \lambda||\mathbf{t}_j||_1) + \iota_{\mathbf{S}_\mathbf{D}}(\mathbf{G})$$
$$\text{s.t.} \quad \mathbf{D}_L = \mathbf{G} \quad \text{and} \quad \mathbf{z}_j = \mathbf{t}_j \quad \text{for} \quad j = 1, ..., SN^2 \tag{3.23}$$

This problem is solved using augmented Lagrangian and the alternating minimization approach in the ADMM framework. ADMM updates for the lth iteration are given by

$$\mathbf{x}^{l+1} = \arg\min_{x} \frac{\beta}{2}||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2 + \frac{1}{2}\sum_{j=1}^{N^2S} ||\mathbf{D}_L\mathbf{z}_j^l - \mathbf{P}_j\mathbf{x}||_2^2 \tag{3.24}$$

$$\mathbf{z}_j^{l+1} = \arg\min_{\mathbf{z}_j} \sum_{j=1}^{N^2S}(\frac{1}{2}||\mathbf{D}_L\mathbf{z}_j - \mathbf{P}_j\mathbf{x}||_2^2 + \frac{\rho}{2}||\mathbf{z}_j - \mathbf{t}_j^l + \mathbf{u}_j^l||_2^2) \tag{3.25}$$

$$\mathbf{t}_j^{l+1} = \arg\min_{\mathbf{t}_j} \sum_{j=1}^{N^2S}(\lambda||\mathbf{t}_j||_1 + \frac{\rho}{2}||\mathbf{z}_j^l - \mathbf{t}_j + \mathbf{u}_j^l||_2^2) \tag{3.26}$$

$$\mathbf{D}_L^{l+1} = \arg\min_{\mathbf{D}_L} \frac{1}{2}\sum_{j=1}^{N^2S} ||\mathbf{D}_L\mathbf{z}_j^l - \mathbf{P}_j\mathbf{x}^l||_2^2 + \frac{\sigma}{2}||\mathbf{D}_L - \mathbf{G}^l + \mathbf{E}^l||_2^2 \tag{3.27}$$

$$\mathbf{G}^{l+1} = \arg\min_{\mathbf{G}} \iota_{\mathbf{S_D}}(\mathbf{G}) + \frac{\sigma}{2}||\mathbf{D}_L - \mathbf{G}^l + \mathbf{E}^l||_2^2 \tag{3.28}$$

where $\mathbf{u}_j$ and $\mathbf{E}$ are ADMM dual variables and ρ and σ are parameters related to step size of the algorithm. We now explain these update steps separately which we refer as image update, sparse-code update, auxiliary variable $\mathbf{t}$-update, dictionary update, auxiliary variable g-update and dual variable update for both representation scenarios.

Note that since the two-dimensional patch-based dictionary is a special case of three-dimensional one and the derivations are very similar, the details of the modifications are omitted, and only small differences are emphasized for the sake of brevity.

3.3.1 Image Update

By expressing the convolution operations with convolution matrices, concatenating the resulting matrices and vectors and ignoring the iteration indices in Eq. (3.24), the x-update optimization problem can be rewritten in the following simplified form:

$$\mathbf{x} = \arg\min_{x} \frac{\beta}{2}||\mathbf{H}\mathbf{x} - \mathbf{y}||_2^2 + \frac{1}{2}||\mathbf{D}\mathbf{z} - \mathbf{P}\mathbf{x}||_2^2 \tag{3.29}$$

where $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_{N^2S} \end{bmatrix}$ for 3D case and $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_S \end{bmatrix}$, $\mathbf{z}_s = \begin{bmatrix} \mathbf{z}_{s,1} \\ \vdots \\ \mathbf{z}_{s,N^2} \end{bmatrix}$ for 2D case.

Furthermore, $\mathbf{P} = \begin{bmatrix} \mathbf{P}_1 \\ \vdots \\ \mathbf{P}_{N^2S} \end{bmatrix}$ for 3D case and $\mathbf{P} = \mathbf{I}_S \bigotimes \begin{bmatrix} \mathbf{P}_1 \\ \vdots \\ \mathbf{P}_{N^2} \end{bmatrix}$ for 2D case.

Also, $\mathbf{D} = \mathbf{I}_{SN^2} \bigotimes \mathbf{D}_L$ for both 3D and 2D cases. Here, $\mathbf{I}_{N^2S} \in \mathbb{R}^{N^2S}$ is an identity matrix and $\bigotimes$ Kronecker multiplication. Size of all the matrices and the vectors are shown in Table 3.1 and Table 3.2 for 3D and 2D cases respectively. This table also includes variables from the next sections.

Table 3.1: Size of the vectors and the matrices for 3D Case

Element	Size	Element	Size
$\mathbf{x}$	$\mathbb{R}^{N^2S}$	$\mathbf{D}_L$	$\mathbb{R}^{pn^2 \times pn^2}$
$\mathbf{z}_j$	$\mathbb{R}^{SN^2}$	$\mathbf{D}$	$\mathbb{R}^{N^2Spn^2 \times N^2Spn^2}$
$\mathbf{z}$	$\mathbb{R}^{N^4S^2}$	$\mathbf{H}_{i,j}$	$\mathbb{R}^{N^2 \times N^2}$
$\mathbf{P}_j$	$\mathbb{R}^{SN^2 \times SN^2}$	$\mathbf{H}$	$\mathbb{R}^{KN^2 \times SN^2}$
$\mathbf{P}$	$\mathbb{R}^{S^2N^4 \times SN^2}$	$\mathbf{Z}$	$\mathbb{R}^{n^2p \times SN^2}$
$\mathbf{X}$	$\mathbb{R}^{n^2p \times SN^2}$		

Table 3.2: Size of vectors and matrices for 2D Case

Element	Size	Element	Size
$\mathbf{x}_s$	$\mathbb{R}^{N^2}$	$\mathbf{P}$	$\mathbb{R}^{SN^4 \times SN^2}$
$\mathbf{x}$	$\mathbb{R}^{N^2S}$	$\mathbf{D}_L$	$\mathbb{R}^{n^2 \times n^2}$
$\mathbf{z}_{s,j}$	$\mathbb{R}^{N^2}$	$\mathbf{D}$	$\mathbb{R}^{N^2Sn^2 \times N^2Sn^2}$
$\mathbf{z}_s$	$\mathbb{R}^{N^4}$	$\mathbf{H}_{i,j}$	$\mathbb{R}^{N^2 \times N^2}$
$\mathbf{z}$	$\mathbb{R}^{N^4S}$	$\mathbf{H}$	$\mathbb{R}^{KN^2 \times SN^2}$
$\mathbf{P}_j$	$\mathbb{R}^{N^2 \times N^2}$	$\mathbf{Z}$	$\mathbb{R}^{n^2 \times SN^2}$
$\mathbf{X}$	$\mathbb{R}^{n^2 \times SN^2}$		

Since image update in Eq. (3.29) is a least-squares problem, it has a closed-form solution as

$$\mathbf{x} = (\mathbf{P}^H\mathbf{P} + \beta\mathbf{H}^H\mathbf{H})^{-1}(\beta\mathbf{H}^H\mathbf{y} + \mathbf{P}^H\mathbf{D}\mathbf{z}). \tag{3.30}$$

For 3D case, after mathematical operations, we obtain $\mathbf{P}^H\mathbf{P}$= $\sum_{j=1}^{N^2S}\mathbf{P}_j^T\mathbf{P}_j$ = $n^2p\mathbf{I}$ and $\mathbf{I}$ is $N^2S \times N^2S$ identity matrix. For 2D case, however, we obtain $\mathbf{P}^H\mathbf{P}$= $\mathbf{I}_S \bigotimes \sum_{j=1}^{N^2}\mathbf{P}_j^T\mathbf{P}_j = n^2\mathbf{I}$. Furthermore, $\mathbf{P}^H\mathbf{D}\mathbf{z}$ = $\mathbf{c}$ = $\sum_{j=1}^{N^2S}\mathbf{P}_j^T\mathbf{D}_L\mathbf{z}_j$ for 3D case and $\mathbf{P}^H\mathbf{D}\mathbf{z}$ = $\mathbf{c}$ = $[\mathbf{c}_1^T|...|\mathbf{c}_S^T]$ where $\mathbf{c}_s$ = $\sum_{j=1}^{N^2}\mathbf{P}_j^T\mathbf{D}_L\mathbf{z}_{s,j}$ for 2D case.

To reduce the computational complexity of obtaining the solution, resulting subproblem is solved in the frequency domain by exploiting the property that circulant convolution matrices are diagonalized by DFT matrix. Since $\mathbf{H}_{k,s}$ is a circular convolution matrix, it reduce to diagonal blocks in the frequency domain. This relationship can be expressed as $\mathbf{H}_{k,s} = \mathbf{F}_{2D}^H\mathbf{\Lambda}_{k,s}\mathbf{F}_{2D}$ where $\mathbf{F}_{2D} \in \mathbb{R}^{N^2\times N^2}$. By inserting $\mathbf{H} = \tilde{\mathbf{F}}_{2D}^H\mathbf{\Lambda}\tilde{\mathbf{F}}_{2D}$, image update step can be expressed as

$$\mathbf{x} = \tilde{\mathbf{F}}_{2D}^H(n^2p\mathbf{I} + \beta\mathbf{\Lambda}^H\mathbf{\Lambda})^{-1}(\beta\mathbf{\Lambda}^H\tilde{\mathbf{F}}_{2D}\mathbf{y} + \tilde{\mathbf{F}}_{2D}\mathbf{c}) \tag{3.31}$$

for 3D case. On the other hand, only difference is made by replacing the coefficient n^2p by n^2 for the 2D case.

For the computation of Eq. (3.31), forming any of the matrices is not required, since resulting matrices are block diagonal. Here, we use the fact that the multiplication of a diagonal matrix and a vector corresponds to an element-wise multiplication. Similarly, the multiplication of two diagonal matrices can be computed via element-wise multiplication. This step provides huge savings for the memory as well as the computation time. The computations of the inverse term $(I + \beta\mathbf{\Lambda^H\Lambda})^{-1}$ and $\beta\mathbf{\Lambda^H\tilde{F}_{2D}y}$ are same with image update of the method presented in Section 3.2. However, the term $\tilde{\mathbf{F}}_{2D}\mathbf{c}$, is updated in each ADMM iteration for 3D and 2D representations in a different way:

3D Case: First, local dictionary matrix $\mathbf{D}_L$ and each sparse vector $\mathbf{z}_j$ is multiplied. Next, to apply $\mathbf{P}_j^H$ operation, each $\mathbf{D}_L\mathbf{z}_j$ vector is reshaped as a 3D patch, inserted into their corresponding patch indices and summed for N^2S patches. Resulting $\mathbf{c} = \mathbf{P}^H\mathbf{D}\mathbf{z}$ is a 3D data cube having same size with the image $x[n_1, n_2, s]$. Then, $\tilde{\mathbf{F}}_{2D}\mathbf{P}^H\mathbf{D}\mathbf{z}$ is acquired by taking 2D Fourier transfom of each slice of $\mathbf{c}$.

2D Case: First, for the computation of $\mathbf{c}_s$, local dictionary matrix $\mathbf{D}_L$ and each sparse vector $\mathbf{z}_{s,j}$ is multiplied. Next, to apply $\mathbf{P}_j^H$ operation, each $\mathbf{D}_L\mathbf{z}_{s,j}$ vector is reshaped as a 2D patch, and inserted into their corresponding patch indices and summed for N^2 patches as $c_s = \sum_{j=1}^{N^2} \mathbf{P}_j^T \mathbf{D}_L \mathbf{z}_{s,j}$. Resulting $\mathbf{c}_s$ is a 2D data having same size with the image $x[n_1, n_2]$. After performing same operations for all $\mathbf{c}_s$, $\tilde{\mathbf{F}}_{2D}\mathbf{P}^H\mathbf{D}\mathbf{z}$ is acquired by taking 2D Fourier transfom of each $\mathbf{c}_s$.

3.3.2 Sparse Code Update

Similar to image update, resulting matrices and vectors are concatenated, and iteration indices are ignored in Eq. (3.25), which is a problem required to be solved in order to obtain sparse code $\mathbf{z}$ update.

3D Case: For 3D case, to solve the sparse code subproblem presented in Eq. (3.25), each $\mathbf{z}_j$ is separately obtained as:

$$\arg\min_{\mathbf{z}_j} \frac{1}{2}||\mathbf{D}_L\mathbf{z}_j - \mathbf{P}_j\mathbf{x}||_2^2 + \frac{\rho}{2}||\mathbf{z}_j - \mathbf{t}_j + \mathbf{u}_{s,j}||_2^2 \tag{3.32}$$

Since this formulation is a least-squares problem, it has a closed-form solution as

$$\mathbf{z}_j = (\rho I + \mathbf{D}_L^H\mathbf{D}_L)^{-1}(\mathbf{D}_L^H\mathbf{P}_j\mathbf{x} + \rho(\mathbf{t}_j - \mathbf{u}_j)) \tag{3.33}$$

Here, $\mathbf{P}_j\mathbf{x}$ correspond to extracted jth patch vector from the image $\mathbf{x}$. Unlike previous discussions, $\mathbf{D}$ is an actual matrix and the problem in Eq. (3.33) is solved by forming the matrices and vectors.

2D Case: For 2D case, to solve the sparse code subproblem presented in Eq. (3.25), each $\mathbf{z}_{s,j}$ is separately obtained as:

$$\arg\min_{\mathbf{z}_{s,j}} \frac{1}{2}||\mathbf{D}_L\mathbf{z}_{s,j} - \mathbf{P}_j\mathbf{x}_s||_2^2 + \frac{\rho}{2}||\mathbf{z}_{s,j} - \mathbf{t}_{s,j} + \mathbf{u}_{s,j}||_2^2 \tag{3.34}$$

Since this formulation is a least-squares problem, it has a closed-form solution as

$$\mathbf{z}_{s,j} = (\rho I + \mathbf{D}_L^H\mathbf{D}_L)^{-1}(\mathbf{D^H}\mathbf{P}_j\mathbf{x}_s + \rho(\mathbf{t}_{s,j} - \mathbf{u}_{s,j})) \tag{3.35}$$

3.3.3 Auxiliary Variable Update I

By concatenating $\mathbf{u}_{s,j}$ and $\mathbf{t}_{s,j}$ vectors lexicographically as $\mathbf{u}$ and $\mathbf{t}$, overall solution for Eq. (3.26) is obtained as

$$\arg\min_{\mathbf{t}} \lambda||\mathbf{t}||_1 + \frac{\rho}{2}||\mathbf{z} - \mathbf{t} + \mathbf{u}||_2^2. \tag{3.36}$$

This problem is solved via soft-thresholding as

$$\mathbf{t} = soft(\mathbf{z} + \mathbf{u}, \lambda/\rho) \tag{3.37}$$

Here, since all operations are element-wise, all are applied on the the images without forming matrices and vectors.

3.3.4 Dictionary Update

By expressing the convolution operations with convolution matrices, concatenating the resulting matrices and vectors and ignoring the iteration indices in Eq. (3.27), the dictionary update problem can be rewritten in the following simplified form:

$$\arg\min_{\mathbf{D}_L} \frac{1}{2}||\mathbf{D}_L\mathbf{Z} - \mathbf{X}||_2^2 + \frac{\sigma}{2}||\mathbf{D}_L - \mathbf{G} + \mathbf{E}||_2^2 \tag{3.38}$$

Here horizontal concatenations occurs as $\mathbf{X}$ = $[\mathbf{x}_1....\mathbf{x}_{N^2S}]$ and $\mathbf{Z}$ = $[\mathbf{z}_1....\mathbf{z}_{N^2S}]$ for 3D case and $\mathbf{X}$ = $[\mathbf{x}_{1,1}....\mathbf{x}_{S,N^2}]$ and $\mathbf{Z}$ = $[\mathbf{z}_{1,1}....\mathbf{z}_{S,N^2}]$ for 2D case. To obtain a typical least-squares form, hermitian operation is applied as

$$\arg\min_{\mathbf{D}_L} \frac{1}{2}||\mathbf{Z}^H\mathbf{D}_L^H - \mathbf{X}^{\mathbf{H}}||_2^2 + \frac{\sigma}{2}||\mathbf{D}_L^H - \mathbf{G}^H + \mathbf{E}^H||_2^2. \tag{3.39}$$

The least-squares solution is now obtained as

$$\mathbf{D}_L = (\mathbf{X}\mathbf{Z}^H + \sigma(\mathbf{G} - \mathbf{E}))(\mathbf{Z}\mathbf{Z}^H + \sigma\mathbf{I})^{-1} \tag{3.40}$$

Here, all matrices and vectors are formed unlike in previous discussions.

3.3.5 Auxiliary Variable Update II

Auxiliary variable g-update problem in Eq. (3.28) is represented as

$$\arg\min_{\mathbf{G}} \iota_{\mathbf{S_D}(\mathbf{G})} + \frac{\sigma}{2}||\mathbf{D}_L - \mathbf{G} + \mathbf{E}||_2^2. \tag{3.41}$$

The solution of the problem is obtained via geometry as

$$\mathbf{G} = \frac{\mathbf{D}_L + \mathbf{E}}{||\mathbf{D}_L + \mathbf{E}||_2}. \tag{3.42}$$

3.3.6 Dual Variable Update

ADMM dual variable updates are also expressed as

$$\mathbf{u}^{l+1} = \mathbf{u}^l + \mathbf{t}^{l+1} - \mathbf{z}^{l+1} \tag{3.43}$$

$$\mathbf{E}^{l+1} = \mathbf{E}^l + \mathbf{G}^{l+1} - \mathbf{D}_L^{l+1}. \tag{3.44}$$

as presented in [60].

3.3.7 ADMM Parameter Update

For the selection of suitable penalty parameter ρ and σ, an adaptive strategy introduced in [60]

$$\rho_{l+1} = \begin{cases} \tau^{incr}\rho^l \text{ if } ||\mathbf{r}_z^l||_2 > \mu||\mathbf{s}_z^l||_2 \\ \rho^k/\tau^{decr} \text{ if } ||\mathbf{r}_z^l||_2 > \mu||\mathbf{r}_z^l||_2 \\ \rho^k \text{ otherwise} \end{cases} \tag{3.45}$$

and

$$\sigma_{k+1} = \begin{cases} \tau^{incr}\sigma^l \text{ if } ||\mathbf{r}_D^l||_2 > \mu||\mathbf{s}_D^l||_2 \\ \sigma^l/\tau^{decr} \text{ if } ||\mathbf{s}_D^l||_2 > \mu||\mathbf{r}_D^l||_2 \\ \sigma^l \text{ otherwise} \end{cases} \tag{3.46}$$

are employed. Here, τ = 2 and μ = 10. $\mathbf{s}_z^{l+1} = \rho(\mathbf{t}^l - \mathbf{t}^{l+1})$ and $\mathbf{r}_z^{l+1} = \mathbf{z}^{l+1} - \mathbf{t}^{l+1}$ are primal and dual residuals of $\mathbf{z}$ respectively [60]. Similarly, $\mathbf{s}_D^{l+1} = \sigma(\mathbf{G}^l - \mathbf{G}^{l+1})$ and $\mathbf{r}_D^{l+1} = (\mathbf{D}^{l+1} - \mathbf{G}^{l+1})$ are primal and dual residuals of $\mathbf{G}$ In the place of stopping criteria, if $||\mathbf{x}^{l+1} - \mathbf{x}^l||_2/||\mathbf{x}^l||_2 < 10^{-4}$ the algorithm is stopped. An additional stopping criteria is derived from following discussion:

$$\begin{aligned} \epsilon_z^{pri} &= \sqrt{n_z}\epsilon^{abs} + \epsilon^{rel} max(||\mathbf{z}^l||_2, ||\mathbf{t}^l||_2, 0), \\ \epsilon_z^{dual} &= \sqrt{n_z}\epsilon^{abs} + \epsilon^{rel}||\boldsymbol{u}^l||_2||_2 \end{aligned} \tag{3.47}$$

Algorithm 2 Image Reconstruction Algorithm with Patch-Based Dictionary

```
Require: $\mathbf{y}$: Measured image, $\mathbf{H}$: Blur filter, $\mathbf{P}$: Patch-Extractor
Ensure: $x$: Reconstructed Image
 1: Choose: $\lambda > 0$, $\rho > 0$, $\beta = 0$, $\mathbf{t}^0$, $\mathbf{u}^0$
 2: $(\rho\mathbf{P}^H\mathbf{P} + \beta\mathbf{H}^H\mathbf{H})^{-1}$ is computed.
 3: $\beta\mathbf{H}^H\mathbf{y}$ is computed
 4: repeat
 5:     $\mathbf{x}^{l+1} = (\rho\mathbf{P}^H\mathbf{P} + \beta\mathbf{H}^H\mathbf{H})^{-1}(\beta\mathbf{H}^H\mathbf{y} + \rho\mathbf{P}^H\mathbf{D}^l\mathbf{z}^l)$
 6:     $\mathbf{z}_j^{l+1} = (\rho I + \mathbf{D}_L^{lH}\mathbf{D}_L^l)^{-1}(\mathbf{D}_L^{lH}\mathbf{P}\mathbf{x}^{l+1} + \rho(\mathbf{t}_j^l - \mathbf{u}_j^l))$
 7: for $j = 1, ..., SN^2$
 8:     $\mathbf{t}^{l+1} = soft(\mathbf{z}^{l+1} + \mathbf{u}^l, \lambda/\rho)$
 9:     $\mathbf{u}^{l+1} = \mathbf{u}^l + \mathbf{z}^{l+1} - \mathbf{t}^{l+1}$
10:     if $Update$ then
11:         $\mathbf{D}_L^{l+1} = (\mathbf{X}^{l+1}\mathbf{Z}^{(l+1)H} + \sigma(\mathbf{G}^l - \mathbf{E}^l))(\mathbf{Z}^{l+1}\mathbf{Z}^{(l+1)H} + \sigma\mathbf{I})^{-1}$
12:         $\mathbf{G}^{l+1} = \frac{(\mathbf{D}_L^l + \mathbf{E}^l)}{||\mathbf{D}_L^l + \mathbf{E}^l||_2}$.
13:         $\mathbf{E}^{l+1} = \mathbf{E}^l + \mathbf{G}^{l+1} - \mathbf{D}_L^{l+1}$.
14:     else
15:         $\mathbf{D}^{l+1} = \mathbf{D}^l$
16:     end if
17: until Some convergence criterion is satisfied
```

$$\begin{aligned} \epsilon_D^{pri} &= \sqrt{n_D}\epsilon^{abs} + \epsilon^{rel} max(||\mathbf{D}^l||_2, ||\mathbf{G}^l||_2, 0), \\ \epsilon_D^{dual} &= \sqrt{n_D}\epsilon^{abs} + \epsilon^{rel}||\mathbf{E}^l||_2||_2 \end{aligned} \tag{3.48}$$

where size of $\mathbf{z}$ is represented via n_z, and size of vectorized $\mathbf{D}$ is denoted by n_D. Here, $\epsilon^{abs} = 0$ and $\epsilon^{rel} = 10^{-3}$. Using Eq. (3.47) and (3.48), the stopping criteria is obtained. If any of $\epsilon_z^{pri} > r_z$, $\epsilon_z^{dual} > s_z$, $\epsilon_D^{pri} > r_D$ or $\epsilon_D^{dual} > s_D$ the algorithm is stopped.

3.4 Image Reconstruction Method With Convolutional Dictionaries

In this section, we develop an image reconstruction method inserting a convolutional dictionary with online learning, which is expressed in Eq. (2.21) [14,50]. By removing dictionary update steps, the algorithm without online dictionary learning can be easily obtained. Note that we emphasize the differences between the update steps of the algorithms with two and three-dimensional models. Now we recall the regularizer with the patch-based dictionary in Eq. (2.21) given by

$$\begin{aligned} \mathcal{R}(\mathbf{x}) = \min_{\mathbf{z}_m, \mathbf{d}_m} \frac{1}{2}\|\sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m - \mathbf{x}\|_2^2 + \lambda \sum_{m=1}^{M} \|\mathbf{z}_m\|_1 \\ \text{s.t.} \quad \|\mathbf{d}_m\|_2 = 1 \quad m = 1, 2..M \end{aligned} \tag{3.49}$$

Now, we insert this prior with convolutional dictionary into the following formulation

$$\min_{\mathbf{x}} \frac{\beta}{2}\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \mathcal{R}(\mathbf{x}) \tag{3.50}$$

Finally, the objective function is obtained as

$$\begin{aligned} \min_{\mathbf{z}_m, \mathbf{x}, \mathbf{d}_m} \frac{\beta}{2}\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \frac{1}{2}\|\sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m - \mathbf{x}\|_2^2 \\ + \lambda \sum_{m=1}^{M} \|\mathbf{z}_m\|_1 \quad s.t. \quad \mathbf{d}_m \in S_d \quad \forall m \end{aligned} \tag{3.51}$$

where

$$S_d = \{\mathbf{u} \in \mathbb{R}^{N^2} : (\mathbf{I} - \mathbf{Q}\mathbf{Q}^{\mathbf{T}})\mathbf{u} = 0, \quad \|\mathbf{u}\|_2 = 1\}. \tag{3.52}$$

The resulting optimization problem is solved for the unknown images $\mathbf{x}$'s, the sparse coefficient vectors $\mathbf{z}_m$'s and dictionary filters $\mathbf{d}_m$'s by using Alternating Direction Method of multipliers (ADMM) [60]. To solve the resulting constrained optimization problem Eq. (3.51), it is converted into an unconstrained problem by adding the constraint to the objective as an indicator function as

$$\begin{aligned} \min_{\mathbf{z}_m, \mathbf{x}, \mathbf{d}_m} \frac{\beta}{2}\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \frac{1}{2}\|\sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m - \mathbf{x}\|_2^2 \\ + \lambda \sum_{m=1}^{M} \|\mathbf{z}_m\|_1 + \sum_{m=1}^{M} \iota_{\mathbf{S_d}}(\mathbf{d_m}). \end{aligned} \tag{3.53}$$

Here, auxiliary variable $\mathbf{t}_m$ for variable $\mathbf{z}_m$ and auxiliary variable $\mathbf{g}_m$ for variable $\mathbf{d}_m$

are inserted into Eq. (3.53) using variable-splitting as

$$\begin{aligned}\min_{\mathbf{z}_m, x, \mathbf{d}_m} \; & \frac{\beta}{2}||\mathbf{y}-\mathbf{H}\mathbf{x}||_2^2 + \frac{1}{2}||\sum_{m=1}^{M}\mathbf{d}_m * \mathbf{z}_m - \mathbf{x}||_2^2 \\ & + \lambda\sum_{m=1}^{M}||\mathbf{t}_m||_1 + \sum_{m=1}^{M}\iota_{S_d}(\mathbf{g}_m) \\ & \text{s.t.} \quad \mathbf{z}_m = \mathbf{t}_m, \quad \mathbf{d}_m = \mathbf{g}_m\end{aligned} \tag{3.54}$$

This problem is solved using augmented Lagrangian and the alternating minimization approach in the ADMM framework. ADMM updates for the lth iteration are given by

$$\mathbf{x}^{l+1} = \arg\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y}-\mathbf{H}\mathbf{x}||_2^2 + \frac{1}{2}||\sum_{m=1}^{M}\mathbf{d}_m^l * \mathbf{z}_m^l - \mathbf{x}||_2^2 \tag{3.55}$$

$$\mathbf{z}_m^{l+1} = \arg\min_{\mathbf{z}_m} \frac{1}{2}||\sum_{m=1}^{M}\mathbf{d}_m^l * \mathbf{z}_m - \mathbf{x}^l||_2^2 + \frac{\rho}{2}\sum_{m=1}^{M}||\mathbf{z}_m - \mathbf{t}_m^l + \mathbf{u}_m^l||_2^2 \tag{3.56}$$

$$\mathbf{t}_m^{l+1} = \arg\min_{\mathbf{t}_m} \lambda\sum_{m=1}^{M}||\mathbf{t}_m||_1 + \frac{\rho}{2}\sum_{m=1}^{M}||\mathbf{z}_m^{l+1} - \mathbf{t}_m^l + \mathbf{u}_m^l||_2^2 \tag{3.57}$$

$$\mathbf{d}_m^{l+1} = \arg\min_{\mathbf{d_m}} \frac{1}{2}||\sum_{m=1}^{M}\mathbf{d}_m * \mathbf{z}_m^l - \mathbf{x}_s^l||_2^2 + \frac{\sigma}{2}\sum_{m=1}^{M}||\mathbf{d}_m - \mathbf{g}_m^l + \mathbf{e}_m^l||_2^2 \tag{3.58}$$

$$\mathbf{g}_m^{l+1} = \arg\min_{\mathbf{g}_m} \sum_{m=1}^{M}\iota_{S_d}(\mathbf{g}_m) + \frac{\sigma}{2}\sum_{m=1}^{M}||\mathbf{d}_m^l - \mathbf{g}_m + \mathbf{e}_m^l||_2^2 \tag{3.59}$$

where $\mathbf{u}_m$ and $\mathbf{e}_m$ are ADMM dual variables and ρ and σ are parameters related to step size of the algorithm. Note that since two-dimensional convolutional representation is a special case of three-dimensional dimensional one, its update steps are obtained very similarly for Eq. (3.55), (3.56), (3.57), (3.58) and (3.59). We now explain these update steps separately which we refer as image update, sparse-code update, auxiliary variable $\mathbf{t}$-update, dictionary update, auxiliary variable $\mathbf{g}$-update and dual variable update in detail for both 3D and 2D cases.

3.4.1 Image Update

We will present the steps of image reconstruction for three-dimensional and two-dimensional dictionaries separately. Section names are called as 3D case and 2D

Table 3.3: Size of vectors and matrices for 3D Case

Element	Size
$\mathbf{x}$	$\mathbb{R}^{N^2S}$
$\mathbf{z}_m, \mathbf{t}_m, \mathbf{u}_m$	$\mathbb{R}^{SN^2}$
$\mathbf{z}, \mathbf{t}, \mathbf{u}$	$\mathbb{R}^{MSN^2}$
$\mathbf{D}_m$	$\mathbb{R}^{SN^2 \times SN^2}$
$\mathbf{D}$	$\mathbb{R}^{SN^2 \times MSN^2}$
$\mathbf{d}_m, \mathbf{g}_m, \mathbf{e}_m$	$\mathbb{R}^{SN^2}$
$\mathbf{d}, \mathbf{g}, \mathbf{e}$	$\mathbb{R}^{MSN^2}$
$\mathbf{H}_{k,s}$	$\mathbb{R}^{N^2 \times N^2}$
$\mathbf{H}$	$\mathbb{R}^{KN^2 \times SN^2}$

Table 3.4: Size of vectors and matrices for 2D Case

Element	Size
$\mathbf{x_s}$	$\mathbb{R}^{N^2}$
$\mathbf{x}$	$\mathbb{R}^{N^2S}$
$\mathbf{z}_{s,m}$	$\mathbb{R}^{N^2}$
$\mathbf{z}_s, \mathbf{t}_s, \mathbf{u}_s$	$\mathbb{R}^{MN^2}$
$\mathbf{z}, \mathbf{t}, \mathbf{u}$	$\mathbb{R}^{MSN^2}$
$\mathbf{D}_m$	$\mathbb{R}^{N^2 \times N^2}$
$\mathbf{D}$	$\mathbb{R}^{N^2 \times MN^2}$
$\tilde{\mathbf{D}}$	$\mathbb{R}^{SN^2 \times MSN^2}$
$\mathbf{d}_m, \mathbf{g}_m, \mathbf{e}_m$	$\mathbb{R}^{N^2}$
$\mathbf{d}, \mathbf{g}, \mathbf{e}$	$\mathbb{R}^{MN^2}$
$\mathbf{H}_{k,s}$	$\mathbb{R}^{N^2 \times N^2}$
$\mathbf{H}$	$\mathbb{R}^{KN^2 \times SN^2}$

case, respectively.

3.4.1.1 3D Case

By expressing the convolution operations with convolution matrices, concatenating the resulting matrices and vectors and ignoring the iteration indices in Eq. (3.55), the image update problem can be rewritten in the following simplified form:

$$\mathbf{x} = \arg\min_{\mathbf{x}} \frac{1}{2}||\mathbf{D}\mathbf{z} - \mathbf{x}||_2^2 + \frac{\beta}{2}||\mathbf{H}\mathbf{x} - \mathbf{y}||_2^2. \tag{3.60}$$

Here, $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_M \end{bmatrix}$ and $\mathbf{D}$ = [$\mathbf{D}_1$ ... $\mathbf{D}_M$] where $\mathbf{D}_m$ is circular convolution matrix of dictionary filter $\mathbf{d}_m$.

Since image update in Eq. (3.60) is a least-squares problem, it has a closed-form solution:

$$\mathbf{x} = (I + \beta\mathbf{H}^{\mathbf{H}}\mathbf{H})^{-1}(\beta\mathbf{H}^{\mathbf{H}}\mathbf{y} + \mathbf{Dz}) \tag{3.61}$$

To reduce the computational complexity of obtaining the solution, resulting subproblem is solved in the DFT domain by exploiting the property that circulant convolution matrices are diagonalized by DFT matrix. Since $\mathbf{H}_{k,s}$ and $\mathbf{D}_m$ are circular convolution matrices, these matrices reduce to diagonal blocks in the frequency domain. This relationship can be expressed as $\mathbf{H}_{k,s} = \mathbf{F}_{2D}^H \mathbf{\Lambda}_{k,s}\mathbf{F}_{2D}$ and $\mathbf{D_m} = \mathbf{F}_{3D}^H\mathbf{\Theta}_m\mathbf{F}_{3D}$ where $\mathbf{F}_{2D} \in \mathbb{R}^{N^2\times N^2}$ and $\mathbf{F}_{3D} \in \mathbb{R}^{N^2S\times N^2S}$ DFT matrices. By inserting $\mathbf{H} = \tilde{\mathbf{F}}_{2D}^H\mathbf{\Lambda}\tilde{\mathbf{F}}_{2D}$ and $\mathbf{D} = \mathbf{F}_{3D}^H\mathbf{\Theta}\tilde{\mathbf{F}}_{3D}$ image update step can be expressed as

$$\mathbf{x} = \tilde{\mathbf{F}}_{2D}^H(I + \beta\mathbf{\Lambda}^H\mathbf{\Lambda})^{-1}(\beta\mathbf{\Lambda}^H\tilde{\mathbf{F}}_{2D}\mathbf{y} + \tilde{\mathbf{F}}_{2D}\mathbf{F}_{3D}^H\mathbf{\Theta}\tilde{\mathbf{F}}_{3D}\mathbf{z}). \tag{3.62}$$

where $\tilde{\mathbf{F}}_{2D}$ and $\tilde{\mathbf{F}}_{3D}$'s are the block 2D and 3D DFT matrices respectively. Further simplification of $\tilde{\mathbf{F}}_{2D}\mathbf{F}_{3D}^H$ is performed via expression of $\mathbf{F}_{3D} = \mathbf{F}_D \bigotimes \mathbf{F}_{2D}$ where $\mathbf{F}_D$ stands for one-dimensional DFT matrix. By using the properties of Kronecker product we obtain

$$\tilde{\mathbf{F}}_S\mathbf{W}^H = (\mathbf{I}_S \otimes \mathbf{F})(\mathbf{G}^H \otimes \mathbf{F}^H) = \mathbf{G}^H \otimes \mathbf{I}_{N^2}. \tag{3.63}$$

which is equivalent to one-dimensional inverse DFT across third-dimension. Here subscripts denote the size of $\mathbf{I}$.

Here, the diagonal of the diagonal matrix $\mathbf{F}_{3D}\mathbf{\Theta}_m$ can be computed by taking the 3D Fourier Transform of $d_m[n_1, n_2, s]$. Similarly, $\mathbf{F}_{3D}\mathbf{z}$ requires taking the 3D Fourier Transform of $z_m[n_1, n_2, s]$. Next, $\mathbf{\Theta}\tilde{\mathbf{F}}_{3D}\mathbf{z}$ = $\sum_{m=1}^{M}\mathbf{\Theta}_m\tilde{\mathbf{F}}_{3D}\mathbf{z}_m$ is performed as summation of element-wise multiplications. Final form of $\tilde{\mathbf{F}}_{2D}\mathbf{F}_{3D}^H\mathbf{\Theta}\tilde{\mathbf{F}}_{3D}\mathbf{z}$ is acquired by taking 1D inverse Fourier transform of $\mathbf{\Theta}\tilde{\mathbf{F}}_{3D}\mathbf{z}$ along the third-dimension. Computations of the inverse term $(I + \beta\mathbf{\Lambda}^{\mathbf{H}}\mathbf{\Lambda})^{-1}$ and $\beta\mathbf{\Lambda}^{\mathbf{H}}\tilde{\mathbf{F}}_{2D}\mathbf{y}$ are same with image update of patch-based dictionary in Chapter 2. Note that, here instead of obtaining $\mathbf{x}$ in time domain, we leave this in the 2D frequency domain as $\tilde{\mathbf{F}}_{2D}\mathbf{x}$ to use in the sparse code $\mathbf{z}$ update.

3.4.1.2 2D Case

For image update with 2D dictionary, similar expressions with (3.61) are obtained by replacing $\mathbf{D}$ with $\tilde{\mathbf{D}}$ and with a small difference in concatenation of $\mathbf{z}$. Here, Eq. (3.61) is modified for 2D case as

$$\mathbf{x} = (I + \beta \mathbf{H}^H \mathbf{H})^{-1}(\beta \mathbf{H}^H \mathbf{y} + \tilde{\mathbf{D}}\mathbf{z}) \tag{3.64}$$

where $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_S \end{bmatrix}$ and $\mathbf{z}_s = \begin{bmatrix} \mathbf{z}_{1,1} \\ \vdots \\ \mathbf{z}_{1,M} \end{bmatrix}$. Note that, $\mathbf{D}$ = [$\mathbf{D}_1$ ... $\mathbf{D}_M$] and $\tilde{\mathbf{D}}$ = $\mathbf{I}_S \bigotimes \mathbf{D}$. As Here, $\mathbf{D}_m = \mathbf{F}_{2D}^H \mathbf{\Theta}_m \mathbf{F}_{2D}$ we obtain the following diagonalization as $\tilde{\mathbf{D}} = \tilde{\mathbf{F}}_{2D}^H \tilde{\mathbf{\Theta}} \tilde{\mathbf{F}}_{2D}$. Hence, two-dimensional image update is expressed as

$$\mathbf{x} = \tilde{\mathbf{F}}_{2D}^H (I + \beta \mathbf{\Lambda}^H \mathbf{\Lambda})^{-1}(\beta \mathbf{\Lambda}^H \tilde{\mathbf{F}}_{2D}\mathbf{y} + \tilde{\mathbf{\Theta}} \tilde{\mathbf{F}}_{2D}\mathbf{z}). \tag{3.65}$$

Here, the diagonal of the diagonal matrix $\mathbf{F}_{2D}\mathbf{\Theta}_m$ can be computed by taking the 2D Fourier Transform of $d_m[n_1, n_2]$. Similarly, $\mathbf{F}_{2D}\mathbf{z}$ requires taking the 2D Fourier Transform of $z_{s,m}[n_1, n_2]$. Next, $\mathbf{\Theta}\tilde{\mathbf{F}}_{2D}\mathbf{z}$ = $\sum_{s=1}^{S} \sum_{m=1}^{M} \mathbf{\Theta}_m \tilde{\mathbf{F}}_{2D}\mathbf{z}_{s,m}$ is performed as summation of element-wise multiplications. Computations of the inverse term $(I + \beta \mathbf{\Lambda}^H \mathbf{\Lambda})^{-1}$ and $\beta \mathbf{\Lambda}^H \tilde{\mathbf{F}}_{2D}\mathbf{y}$ are same with image update of patch-based dictionary in Chapter 2. Note that, here instead of obtaining $\mathbf{x}$ in time domain, we leave this in the 2D frequency domain as $\tilde{\mathbf{F}}_{2D}\mathbf{x}$ to use in the sparse code $\mathbf{z}$ update.

3.4.2 Sparse Code Update

3.4.2.1 3D Case

Similar to image update, resulting matrices and vectors are concatenated and iteration indices are ignored in Eq. (3.56), sparse code update is represented in following form:

$$\mathbf{z} = \arg\min_{\mathbf{z}} \frac{1}{2} ||\mathbf{D}\mathbf{z} - \mathbf{x}||_2^2 + \frac{\rho}{2} ||\mathbf{z} - \mathbf{t} + \mathbf{u}||_2^2 \tag{3.66}$$

where $\mathbf{u}_m$ and $\mathbf{t}_m$ vectors are vertically concatenated in lexigocraphic order as $\mathbf{u}$ and $\mathbf{t}$.

For sparse code $\mathbf{z}$ update, we need to solve the problem in Eq. (3.66). To solve this problem, we need to solve the following linear system

$$(\rho I + \mathbf{D}^H\mathbf{D})\mathbf{z} = (\mathbf{D}^H\mathbf{x} + \rho(\mathbf{t} - \mathbf{u})). \tag{3.67}$$

Considering similar steps with image update, sparse code update is also efficiently computed in the frequency domain:

$$(\rho I + \mathbf{\Theta}^H\mathbf{\Theta})\tilde{\mathbf{F}}_{3D}\mathbf{z} = (\mathbf{\Theta}^H\mathbf{F}_{3D}\mathbf{x} + \rho\tilde{\mathbf{F}}_{3D}(\mathbf{t} - \mathbf{u})) \tag{3.68}$$

First, $\tilde{\mathbf{F}}_{3D}(\mathbf{t}-\mathbf{u})$ and $\mathbf{\Theta}$ is computed by taking 3D FFT of $(t_m[n_1, n_2, s]-u_m[n_1, n_2, s])$ and $\mathbf{D}_m$ for $m = 1, ..., M$. By taking the 1D FFT of $\tilde{\mathbf{F}}_{2D}\mathbf{x}$ along the third dimension, we acquire $\mathbf{F}_{3D}\mathbf{x}$. Then, $\mathbf{\Theta}^H\mathbf{F}_{3D}\mathbf{x}$ is computed via sum of element-wise multiplications.

$(\rho I + \mathbf{\Theta}^H\mathbf{\Theta})$ consists of sum of a block diagonal rank-one matrix and a diagonal matrix. Instead of solving sparse code update problem in Eq. (3.68) via recursive block-matrix inversion, rank-one property makes available a cost efficient solution. We exploit a fast Sherman-Morrison method by rearranging the terms of the linear system in Eq. (3.68), as proposed in [30]. Assume that index $[n_1, n_2, s]$ is denoted by n for simplicity. After rearranging the terms of $\mathbf{\Theta}_m[n]$'s, we obtain the vectors $\bar{\mathbf{\Theta}}_n[m]$ of length M. Then, we denote the right-hand side of Eq. (3.68) by $\mathbf{c}$ and its rearranged version as $\bar{\mathbf{c}}_n$. By using Sherman-Morrison formula, we obtain the following solution as

$$\bar{\mathbf{z}}_n = \rho^{-1}(\bar{\mathbf{c}}_n - \frac{\bar{\mathbf{\Theta}}_n^H\bar{\mathbf{c}}_n}{\rho + \bar{\mathbf{\Theta}}_n^H\bar{\mathbf{\Theta}}_n}\bar{\mathbf{\Theta}}_n) \tag{3.69}$$

where $\bar{\mathbf{z}}_n$ corresponds to the rearranged version of $\tilde{\mathbf{F}}_{3D}\mathbf{z}$. Using this, we form the sparse code $\mathbf{z}$.

3.4.2.2 2D Case

Here the sparse code $\mathbf{z}$-update problem in Eq. (3.67) is modified by replacing $\mathbf{D}$ by $\tilde{\mathbf{D}}$ as follows:

$$(\rho I + \tilde{\mathbf{D}}^H\tilde{\mathbf{D}})\mathbf{z} = (\tilde{\mathbf{D}}^H\mathbf{x} + \rho(\mathbf{t} - \mathbf{u})). \tag{3.70}$$

In this case, by inserting $\mathbf{D}_m = \mathbf{F}_{2D}^H \boldsymbol{\Theta}_m \mathbf{F}_{2D}$ and $\tilde{\mathbf{D}} = \tilde{\mathbf{F}}_{2D}^H \tilde{\boldsymbol{\Theta}} \tilde{\mathbf{F}}_{2D}$, the solution is expressed as

$$(\rho I + \tilde{\boldsymbol{\Theta}}^H \tilde{\boldsymbol{\Theta}}) \tilde{\mathbf{F}}_{2D} \mathbf{z} = (\tilde{\boldsymbol{\Theta}}^H \tilde{\mathbf{F}}_{2D} \mathbf{x} + \rho \tilde{\mathbf{F}}_{2D} (\mathbf{t} - \mathbf{u})) \tag{3.71}$$

The linear system is separable to S linear systems for S spatial images as

$$(\rho I + \boldsymbol{\Theta}^H \boldsymbol{\Theta}) \tilde{\mathbf{F}}_{2D} \mathbf{z}_s = (\boldsymbol{\Theta}^H \mathbf{F}_{2D} \mathbf{x}_s + \rho \tilde{\mathbf{F}}_{2D} (\mathbf{t}_s - \mathbf{u}_s)) \tag{3.72}$$

and each one is efficiently solved via fast Sherman-Morrison approach. By 2D fourier transforms instead of 3D transforms same steps are applied. Note that s now represent number of separate images instead of the third dimension index. Hence, steps are changed accordingly. For example, $\tilde{\mathbf{F}}_{2D}(\mathbf{t} - \mathbf{u})$ and $\boldsymbol{\Theta}$ is computed by taking 2D Fourier transform of $(t_{s,m}[n_1, n_2] - u_{s,m}[n_1, n_2])$ and $\mathbf{D}_m$ for $m = 1, ..., M$. Remaining changes are applied similarly.

3.4.3 Auxiliary Variable Update I

By concatenating $\mathbf{u}_s$ and $\mathbf{t}_s$ vectors lexicographically as $\mathbf{u}$ and $\mathbf{t}$, overall solution for the problem in Eq. (3.57) is obtained as

$$\mathbf{t} = \arg\min_{\mathbf{t}} \lambda ||\mathbf{t}||_1 + \frac{\rho}{2} ||\mathbf{z} - \mathbf{t} + \mathbf{u}||_2^2. \tag{3.73}$$

for both 2D representations and 3D representations. This problem is solved via soft-thresholding as

$$\mathbf{t} = soft(\mathbf{z} + \mathbf{u}, \lambda/\rho) \tag{3.74}$$

Then, component-wise soft-thresholding is performed as

$$\mathbf{t} = sign(\mathbf{x} + \mathbf{u}) \odot max(0, |\mathbf{x} + \mathbf{u}| - \lambda/\rho). \tag{3.75}$$

As it is an element-wise opeation, it is completed without forming the vectors.

3.4.4 Dictionary Update

3.4.4.1 3D Case

For the solution of the problem in Eq. (3.58) which is required to obtain dictionary update, we follow a similar path to the sparse code update by modifying of Eq. (3.58) as

$$\mathbf{d} = \arg\min_{\mathbf{d}} \frac{1}{2}||\mathbf{Zd} - \mathbf{x}||_2^2 + \frac{\sigma}{2}||\mathbf{d} - \mathbf{g} + \mathbf{e}||_2^2 \tag{3.76}$$

where $\mathbf{Z}$ = $[\mathbf{Z}_1 \; ... \; \mathbf{Z}_M]$ and $\mathbf{d}_m$ vectors are vertically concatenated as $\mathbf{d}$. The linear system providing the solution of least-squares problem is

$$(\sigma\mathbf{I} + \mathbf{Z}^H\mathbf{Z})\mathbf{d} = \mathbf{Z}^H\mathbf{x} + \rho(\mathbf{g} - \mathbf{e}). \tag{3.77}$$

By inserting $\mathbf{Z}$ = $\mathbf{F}_{3D}^H\mathbf{\Gamma}\tilde{\mathbf{F}}_{3D}$ to (3.77), the solution in DFT domain can be represented as

$$(\sigma\mathbf{I} + \mathbf{\Gamma}^H\mathbf{\Gamma})\tilde{\mathbf{F}}_{3D}\mathbf{d} = \mathbf{\Gamma}^H\mathbf{F}_{3D}\mathbf{x} + \rho\tilde{\mathbf{F}}_{3D}(\mathbf{g} - \mathbf{e}).. \tag{3.78}$$

The left-hand side consists of the sum of a block diagonal and rank-1 matrix and a diagonal matrix and solved by the fast Sherman-Morrison approach that we used for the solution of sparse code update in Eq. (3.67).

3.4.4.2 2D Case

When 2D dictionaries are exploited, concatenation of vectors slightly differs than 3D case in Eq. (3.76). By considering the 2D convolutional dictionary, the dictionary update is expressed as

$$\mathbf{d} = \operatorname*{argmin}_{\mathbf{d}} \frac{1}{2}\sum_{s=1}^{S}||\mathbf{Z}_s\mathbf{d} - \mathbf{x}_s||_2^2 + \frac{\sigma}{2}||\mathbf{d} - \mathbf{g} + \mathbf{e}||_2^2 \tag{3.79}$$

where $\mathbf{Z}_s$ = $[\mathbf{Z}_{s,1} \; ... \; \mathbf{Z}_{s,M}]$ and $\mathbf{d}_m$ vectors are vertically concatenated as $\mathbf{d}$. Here, least-squares solution is

$$\mathbf{d} = (\sigma\mathbf{I} + \sum_{s=1}^{S}\mathbf{Z}_s^H\mathbf{Z}_s)^{-1}(\sum_{s=1}^{S}\mathbf{Z}_s^H\mathbf{x}_s + \rho(\mathbf{g} - \mathbf{e})). \tag{3.80}$$

Here, block matrix inversion used in $\mathbf{x}$-update can also be exploited for Eq. (3.80). However, since sparse code $\mathbf{z}$ term change in every iteration, inverse term should be computed as well. Computing this inverse in each iteration is computationally costly. Instead, we use the linear system providing the least-squares solution is expressed as

$$(\sigma\mathbf{I} + \sum_{s=1}^{S}\mathbf{Z}_s^H\mathbf{Z}_s)\mathbf{d} = \sum_{s=1}^{S}\mathbf{Z}_s^H\mathbf{x}_s + \rho(\mathbf{g} - \mathbf{e}). \tag{3.81}$$

By inserting $\mathbf{Z}_s = \mathbf{F}_{2D}^H \mathbf{\Gamma}_S \tilde{\mathbf{F}}_{2D}$ to (3.81), the solution in the frequency domain can be represented as

$$(\sigma \mathbf{I} + \sum_{s=1}^{S} \mathbf{\Gamma}_s^H \mathbf{\Gamma}_s) \tilde{\mathbf{F}}_{2D} \mathbf{d} = \sum_{s=1}^{S} \mathbf{\Gamma}_s^H \mathbf{F}_{2D} \mathbf{x}_s + \rho \tilde{\mathbf{F}}_{2D} (\mathbf{g} - \mathbf{e}). \tag{3.82}$$

The term in the left hand-side of Eq. (3.82) is rank-S term, and hence this linear system is not solvable via efficient Sherman-Morrison formula. Hence, several other methods for the solution are presented such as Iterated Sherman-Morrison Approach, Conjugate Gradient method, spatial tiling and consensus framework [31]. Iterated Sherman-Morrison approch has been utilized for the solution of this term, this is iterative version of Sherman-Morrison approach and presented in [30]. Iterated Sherman-Morrison is also presented in kkkıAppendix A.

3.4.5 Auxiliary Variable Update II

For both 2D and 3D cases, the problem in Eq. (3.59) for the solution of auxiliary variable g-update is separable to M subproblems. Each subproblem can be represented as

$$\arg\min_{\mathbf{g}_m} \iota_{S_d}(\mathbf{g}_m) + \frac{\sigma}{2} ||\mathbf{d}_m - \mathbf{g}_m + \mathbf{e}_m||_2^2. \tag{3.83}$$

The solution of the problem is obtained via geometry as

$$\mathbf{g_m} = \frac{\mathbf{Q}\mathbf{Q}^T(\mathbf{d}_m + \mathbf{e}_m)}{||\mathbf{Q}\mathbf{Q}^T(\mathbf{d}_m + \mathbf{e}_m)||_2}. \tag{3.84}$$

Here, summation and division operations are element-wise. Note that in the implementation $\mathbf{Q}^T$ operation crops resulting $\mathbf{d}_m + \mathbf{e}_m$ as $L \times L \times R$ which is target dictionary size for 3D case. Next, $\mathbf{Q}$ operation zero-pads this crop to the size of $N \times N \times S$ which is the original image size. Similarly, $\mathbf{Q}^T$ operation crops resulting $\mathbf{d}_m + \mathbf{e}_m$ as $L \times L$ for 2D case. Next, $\mathbf{Q}$ operation zero-pads this crop to the size of $N \times N$e.

3.4.6 Dual Variable Update

ADMM dual variable updates are expressed as

$$\mathbf{u}^{l+1} = \mathbf{u}^l + \mathbf{t}^{l+1} - \mathbf{z}^{l+1} \tag{3.85}$$

Algorithm 3 Image Reconstruction Algorithm with Convolutional 3D Dictionary

```
Require: $\mathbf{y}$: Measured image, $\mathbf{H}$: Blur filter,
Ensure: $x$: Reconstructed Image
1:  Choose: $\lambda > 0$, $\rho > 0$, $\beta = 0$, $\mathbf{t}^0$, $\mathbf{u}^0$
2:  $(\rho\mathbf{I} + \beta\mathbf{H}^H\mathbf{H})^{-1}$ is computed.
3:  $\beta\mathbf{H}^H\mathbf{y}$ is computed
4:  repeat
5:      $\mathbf{x}^{l+1} = (\rho\mathbf{I} + \beta\mathbf{H}^H\mathbf{H})^{-1}(\beta\mathbf{H}^H\mathbf{y} + \rho\mathbf{D}\mathbf{z}^l)$
6:      $\mathbf{z}^{l+1} = (\rho I + \mathbf{D}^{lH}\mathbf{D}^l)^{-1}(\mathbf{D}^{lH}\mathbf{x}^{l+1} + \rho(\mathbf{t}^l - \mathbf{u}^l))$
7:      $\mathbf{t}^{l+1} = soft(\mathbf{z}^{l+1} + \mathbf{u}^l, \lambda/\rho)$
8:      $\mathbf{u}^{l+1} = \mathbf{u}^l + \mathbf{z}^{l+1} - \mathbf{t}^{l+1}$
9:      if $Update$ then
10:         $\mathbf{D}^{l+1} = (\sigma\mathbf{I} + \mathbf{Z}^{(l+1)H}\mathbf{Z}^{l+1})^{-1}(\mathbf{Z}^{(l+1)H}\mathbf{x} + \rho(\mathbf{g}^l - \mathbf{e}^l))$.
11:         $\mathbf{g}_m^{l+1} = \frac{\mathbf{Q}\mathbf{Q}^T(\mathbf{d}_m^{l+1} + \mathbf{e}_m^l)}{||\mathbf{Q}\mathbf{Q}^T(\mathbf{d}_m^{l+1} + \mathbf{e}_m^l)||_2}$ for $m = 1, ..., M$
12:         $\mathbf{e}^{l+1} = \mathbf{e}^l + \mathbf{g}^{l+1} - \mathbf{d}^{l+1}$.
13:     else
14:         $\mathbf{D}^{l+1} = \mathbf{D}^l$
15:     end if
16: until Some convergence criterion is satisfied
```

$$\mathbf{e}^{l+1} = \mathbf{e}^l + \mathbf{g}^{l+1} - \mathbf{d}^{l+1}. \tag{3.86}$$

3.4.7 ADMM Parameter Update

Same update scheme used in patch-based scenario is applied for the algorithm with convolutional prior. However, some modifications are performed as $\mathbf{s}_d^{l+1} = \sigma(\mathbf{g}^l - \mathbf{g}^{l+1})$ and $\mathbf{r}_d^{l+1} = (\mathbf{d}^{l+1} - \mathbf{g}^{l+1})$ and

$$\begin{aligned} \epsilon_d^{pri} &= \sqrt{n_d}\epsilon^{abs} + \epsilon^{rel} max(||\mathbf{d}^l||_2, ||\mathbf{g}^l||_2, 0), \\ \epsilon_d^{dual} &= \sqrt{n_d}\epsilon^{abs} + \epsilon^{rel}||\mathbf{e}^l||_2||_2 \end{aligned} \tag{3.87}$$

where the size of vectorized $\mathbf{d}$ is denoted by n_d. Here, $\epsilon^{abs} = 0$ and $\epsilon^{rel} = 10^{-3}$. Variables introduced in Eq. (3.47) and (3.87) determine the stopping criteria, if any of $\epsilon_z^{pri} > r_z$, $\epsilon_z^{dual} > s_z$, $\epsilon_d^{pri} > r_d$ or $\epsilon_d^{dual} > s_d$ the algorithm is stopped.

3.5 Image Reconstruction with Convolutional Dictionaries and Tikhonov Regularization

In this section, we insert convolutional prior with Tikhonov regularization in Eq. (2.26) into $\mathcal{R}(\mathbf{x})$ and the following formulation of the problem is obtained as

$$\begin{aligned} \min_{\mathbf{z}_m,\mathbf{x},\mathbf{d}_m} \; & \frac{\beta}{2}||\mathbf{H}\mathbf{x}-\mathbf{y}||_2^2 + \frac{1}{2}||\sum_{m=1}^{M}\mathbf{d}_m * \mathbf{z}_m - \mathbf{x}||_2^2 + \sum_{m=1}^{M}\iota_{S_d}(\mathbf{g}_m) + \\ & \sum_{m=1}^{M}||\mathbf{t}_m||_1 + \frac{\mu}{2}\sum_{m=1}^{M}(||\mathbf{r}_1 * \mathbf{z}_m||_2^2 + ||\mathbf{r}_2 * \mathbf{z}_m||_2^2 + ||\mathbf{r}_3 * \mathbf{z}_m||_2^2) \\ & \text{s.t.} \quad \mathbf{z}_m = \mathbf{t}_m, \quad \mathbf{d}_m = \mathbf{g}_m \end{aligned} \tag{3.88}$$

where $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ are filters that compute the gradient in the first, the second and the third dimensions. This modification only change sparse code update stage as follows:

$$\mathbf{z} = \arg\min_{\mathbf{z}} \frac{1}{2}||\mathbf{D}\mathbf{z}-\mathbf{x}||_2^2 + \frac{\rho}{2}||\mathbf{z}-\mathbf{t}+\mathbf{u}||_2^2 + \frac{\mu}{2}(||\mathbf{R}_1\mathbf{z}||_2^2 + ||\mathbf{R}_2\mathbf{z}||_2^2 + ||\mathbf{R}_3\mathbf{z}||_2^2) \tag{3.89}$$

where $\mathbf{R}_1$, $\mathbf{R}_2$ and $\mathbf{R}_3$ are circular convolution matrices of $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ respectively. The modified linear system which gives the solution is expressed as:

$$(\rho\mathbf{I} + \mu\mathbf{R}_1^H\mathbf{R}_1 + \mu\mathbf{R}_2^H\mathbf{R}_2 + \mu\mathbf{R}_3^H\mathbf{R}_3 + \mathbf{D}^H\mathbf{D})\mathbf{z} = (\mathbf{D}^H\mathbf{x} + \rho(\mathbf{t}-\mathbf{u})) \tag{3.90}$$

This step is solved in the frequency-domain using Sherman-Morrison formula. Since $\mathbf{R}_1$, $\mathbf{R}_2$ and $\mathbf{R}_3$'s are circular convolution matrices, they correspond to the diagonal matrices in the frequency domain. Detailed solution is not presented as we already a very similar solution in Eq. (3.67). [64] claims that this version of the algorithm provides higher image reconstruction performance. Also note that new formulation in 3.88 is expressed for 2D case as follows:

$$\begin{aligned} \min_{\mathbf{z}_{s,m},\mathbf{x}_s,\mathbf{d}_m} \; & \frac{\beta}{2}||\mathbf{H}\mathbf{x}-\mathbf{y}||_2^2 + \frac{1}{2}\sum_{s=1}^{S}||\sum_{m=1}^{M}\mathbf{d}_m * \mathbf{z}_{s,m} - \mathbf{x}_s||_2^2 + \sum_{m=1}^{M}\iota_{S_d}(\mathbf{g}_m) + \\ & \lambda\sum_{s=1}^{S}\sum_{m=1}^{M}||\mathbf{t}_{s,m}||_1 + \frac{\mu}{2}\sum_{s=1}^{S}\sum_{m=1}^{M}(||\mathbf{r}_1 * \mathbf{z}_{s,m}||_2^2 + ||\mathbf{r}_2 * \mathbf{z}_{s,m}||_2^2) \\ & \text{s.t.} \quad \mathbf{z}_{s,m} = \mathbf{t}_{s,m}, \quad \mathbf{d}_m = \mathbf{g}_m \end{aligned} \tag{3.91}$$

This modification only change sparse code update stage as

$$\mathbf{z} = \arg\min_{\mathbf{z}} \frac{1}{2}||\tilde{\mathbf{D}}\mathbf{z} - \mathbf{x}||_2^2 + \frac{\rho}{2}||\mathbf{z} - \mathbf{t} + \mathbf{u}||_2^2 + \frac{\mu}{2}(||\tilde{\mathbf{R}}_1\mathbf{z}||_2^2 + ||\tilde{\mathbf{R}}_2\mathbf{z}||_2^2) \tag{3.92}$$

where $\tilde{\mathbf{R}}_1$ and $\tilde{\mathbf{R}}_2$ are circular convolution matrices of $\tilde{\mathbf{r}}_1$ and $\tilde{\mathbf{r}}_2$ respectively. The solution is modified as follows,

$$\mathbf{z} = (\rho I + \mu\tilde{\mathbf{R}}_1^H\tilde{\mathbf{R}}_1 + \mu\tilde{\mathbf{R}}_2^H\tilde{\mathbf{R}}_2 + \tilde{\mathbf{D}}^H\tilde{\mathbf{D}})^{-1}(\tilde{\mathbf{D}}^H\mathbf{x} + \rho(\mathbf{t} - \mathbf{u})) \tag{3.93}$$

Detailed solutions of these updates are not presented since we already discussed very similar solution in Eq. (3.70).

3.6 Computational Complexity

3.6.1 3D Case

In Table 3.5, the computational cost of the algorithms for each variable update is presented in detail. For the sparsifying transform algorithm, computational complexity is dominated by the image update with the computational cost of $\mathcal{O}(S^2N^2 + N^2Slog(N^2S))$. However, other updates of the problem are computationally cheaper. Here, $N \times N \times S$ is the size of the data cube. Since online learning is not included for sparsifying transform, it has the lowest computational cost among three algorithms. In the patch-based dictionary algorithm, the steps which cause the highest costs are image update and sparse code update with the complexity of $\mathcal{O}(N^4S^2)$ and $\mathcal{O}(N^2Sn^6q^3)$. $n \times n \times q$ is the size of the 3D patch extracted from the data-cube. Lastly, for the convolutional dictionary, the steps which result in the highest computational complexity are $\mathbf{z}$, $\mathbf{t}$ and $\mathbf{d}$ updates. Similarly, FFT computations of these updates dominate the computational cost, which can be concluded as $\mathcal{O}(SMN^2log(N^2S))$. Note that M corresponds to the number of dictionary filters. The computational cost of both patch-based and convolutional dictionary-based algorithms with/without online dictionary learning can be inferred from this table.

Table 3.5: Computational cost of algorithms for 3D case

Updates	Sparsifying Transform	Patch-Based Dictionary	Convolutional Dictionary
Image Update	$\mathcal{O}(S^2N^2 + N^2Slog(N^2S))$	$\mathcal{O}(N^4S^2)$	$\mathcal{O}(S^2N^2)$
Sparse Code Update	-	$\mathcal{O}(N^2Sn^6q^3)$	$\mathcal{O}(SMN^2log(N^2S))$
Auxiliary $\mathbf{t}$ Update	$\mathcal{O}(N^2Slog(N^2S))$	$\mathcal{O}(N^2Sn^2q)$	$\mathcal{O}(SMN^2log(N^2S))$
Dual $\mathbf{u}$ Update	$\mathcal{O}(SN^2)$	$\mathcal{O}(N^2Sn^2q)$	$\mathcal{O}(SMN^2)$
Dictionary Update	-	$\mathcal{O}(n^6q^3 + N^2Sn^2q)$	$\mathcal{O}(SMN^2log(N^2S))$
Auxiliary $\mathbf{g}/\mathbf{G}$ Update	-	$\mathcal{O}(n^2q)$	$\mathcal{O}(MN^2Slog(N^2S))$
Dual $\mathbf{e}/\mathbf{E}$ Update	-	$\mathcal{O}(n^2q)$	$\mathcal{O}(SMN^2)$

Table 3.6: Computational cost of algorithms for 2D case

Updates	Sparsifying Transform	Patch-Based Dictionary	Convolutional Dictionary
Image Update	$\mathcal{O}(N^2Slog(N^2))$	$\mathcal{O}(N^4S)$	$\mathcal{O}(S^2N^2)$
Sparse Code Update	-	$\mathcal{O}(N^2Sn^6)$	$\mathcal{O}(SMN^2log(N^2))$
Auxiliary $\mathbf{t}$ Update	$\mathcal{O}(N^2Slog(N^2))$	$\mathcal{O}(N^2Sn^2)$	$\mathcal{O}(SMN^2log(N^2))$
Dual $\mathbf{u}$ Update	$\mathcal{O}(SN^2)$	$\mathcal{O}(N^2Sn^2)$	$\mathcal{O}(SMN^2)$
Dictionary Update	-	$\mathcal{O}(n^6)$	$\mathcal{O}(SMN^2log(N^2))+$ $\mathcal{O}(S^2MN^2)$
Auxiliary $\mathbf{g}/\mathbf{G}$ Update	-	$\mathcal{O}(n^2)$	$\mathcal{O}(MN^2Slog(N^2))$
Dual $\mathbf{e}/\mathbf{E}$	-	$\mathcal{O}(n^2)$	$\mathcal{O}(SMN^2)$

3.6.2 2D Case

In Table 3.6, computational cost of the algorithms for each variable update is presented in detail. For sparsifying transform algorithm, computational complexity is dominated by image update with computational cost of $\mathcal{O}(N^2Slog(N^2))$. Here, $N \times N \times S$ is the size of the data cube. Since online learning is not included for sparsifying transform, it has the lowest computational cost among three algorithms. In the patch-based dictionary algorithm, the steps which causes the highest costs are image update and sparse code update with the complexity of $\mathcal{O}(N^4S)$ and $\mathcal{O}(N^2Sn^6)$. $n \times n$ is the size of the 2D patch extracted from each slices of the data-cube. Lastly, for convolutional dictionary, the step which results in the highest computational complexity is the $\mathbf{d}$ update. Similarly, FFT computations of this update dominates the computational cost which can be concluded as $\mathcal{O}(SMN^2log(N^2))$. Note that M corresponds to number of dictionary filters. The computational cost of both patch-based and convolutional dictionary based-algoritms with/without online dictionary learning can be inferred from this table.

CHAPTER 4

PERFORMANCE COMPARISON

4.1 Introduction

Here the numerical performance of the developed algorithms is evaluated for a computational spectral imaging problem [11, 12]. In the analysis case, discrete derivative operators or sparsifying transforms are utilized such as discrete cosine transform (DCT), wavelets, or their Kronecker-product forms [6, 13, 16]. In the synthesis case, convolutional or patch-based dictionaries are utilized, which can also be adapted to correlations in different dimensions [7, 19–24, 30, 32–34, 40, 41]. Online dictionary learning is also performed to improve image reconstruction quality.

Before the numerical results, we present reconstruction performance metrics that we used to evaluate the image reconstruction qualities of the algorithms. Then, a discrete spectrum photon sieve spectral imaging (PSSI) problem is solved. For this problem, the reconstructed images are not correlated along the third dimension, and hence two-dimensional priors are exploited. Furthermore, we also present a comprehensive analysis of dictionaries with respect to changing dictionary size and dictionary training set. Secondly, the numerical performance of the developed algorithms is evaluated for continuous spectrum problems. In this case, we first evaluate the performance of the algorithms for a denoising problem to measure the image representation quality of the priors. Then, we investigate the numerical performance of the algorithms for the PSSI problem. Here, the reconstructed images are correlated along the third dimension, and hence three-dimensional priors are enforced. We also optimize the dictionary size to attain the best performance among different dictionaries. Note that, since extracting three-dimensional patches and performing dictionary learning on these dictionaries is computationally intensive, we do not evaluate the performance of the patch-based algorithm for the continuous spectrum problem.

4.1.1 Reconstruction Performance Metrics

The image reconstruction fidelity is measured by numerically comparing the reconstructed images with the true intensity images. For the comparison, three quality metrics are used: Peak signal-to-noise ratio (PSNR), structural similarity index (SSIM), and spectral angular mapper (SAM). Note that the results that will be presented here are the average success of various reconstructed images.

Peak Signal-To-Noise Ratio: PSNR is computed by dividing the maximum possible intensity value for the reference image to the mean squared error (MSE) between reference and reconstructed images. We define two different PSNRs for the 2D case and 3D cases. First, for the 2D case, we reconstruct S images that are uncorrelated in third-dimension. PSNR of each sth spatial image is determined as follows:

$$PSNR(\hat{x}_s[n_1,n_2],x_s[n_1,n_2]) = 20\log_{10}\left(\frac{max(x_s[n_1,n_2])}{\sqrt{\frac{\sum_{n_1,n_2}(x_s[n_1,n_2]-\hat{x}_s[n_1,n_2])^2}{N^2}}}\right) \tag{4.1}$$

For overall PSNR we take the average of individual PSNRs.

$$Av.PSNR = \frac{PSNR(x_s[n_1,n_2],\hat{x}_s[n_1,n_2])}{S} \tag{4.2}$$

Next, for 3D case, PSNR is obtained directly PSNR of each sth spatial image is determined as follows:

$$PSNR(\hat{x}[n_1,n_2,s],x[n_1,n_2,s]) = 20\log_{10}\left(\frac{max(x[n_1,n_2,s])}{\sqrt{\frac{\sum_{n_1,n_2,s}(x[n_1,n_2,s]-\hat{x}[n_1,n_2,s])^2}{N^2S}}}\right) \tag{4.3}$$

Structural Similarity Index: SSIM is an image quality metric that compares a reconstructed image to a reference image in terms of the structural similarity of luminance, contrast, and structure [66]. SSIM is defined as

$$SSIM(\hat{x},x) = [l(\hat{x},x)]^{\alpha}[c(\hat{x},x)]^{\beta}[s(\hat{x},x)]^{\gamma} \tag{4.4}$$

where

$$l(\hat{x},x) = \frac{2\mu_{\hat{x}}\mu_x + c_1}{\mu_{\hat{x}}^2+\mu_x^2+c_1} \tag{4.5}$$

$$c(\hat{x}, x) = \frac{2\sigma_{\hat{x}}\sigma_x + c_2}{\sigma_{\hat{x}}^2 + \sigma_x^2 + c_2} \tag{4.6}$$

$$s(\hat{x}, x) = \frac{2\sigma_{\hat{x}x} + c_3}{\sigma_{\hat{x}}\sigma_x + c_3} \tag{4.7}$$

Here, parameters $\mu_{\hat{x}}$, μ_x, $\sigma_{\hat{x}}$, σ_x and $\sigma_{\hat{x}x}$ are local sample means, standard deviations and correlation coefficient of $\hat{x}$ and x. Furthermore, α, β and γ are the weight parameters which determine the relative importance of the three components, and c_1, c_2 and c_3 are small constants used to stabilize these expressions when the denominator is small.

Spectral Angular Mapper: SAM resolves spectral similarity between the reconstructed image and the reference image by taking the average of the calculation of a spectral angle between reconstructed and reference spectral vectors which have a common origin. After computing an angle for each spectral vector, we take the average of these angles for all pixels. SAM is applied for the data cubes that have a correlation across the third dimension. Therefore, we utilized this metric only for the 3D case.

$$\theta = \frac{1}{N^2} \sum_{n_1, n_2} \cos^{-1}\left(\frac{\sum_{s=1}^{S} \hat{x}[n_1, n_2, s] x[n_1, n_2, s]}{\sqrt{\sum_{s=1}^{S} \hat{x}^2[n_1, n_2, s]}\sqrt{\sum_{s=1}^{S} x^2[n_1, n_2, s]}}\right) \tag{4.8}$$

Note that decreasing SAM value indicates increasing image reconstruction quality for the data-cube.

4.1.2 Numerical results for the 2D Case

In this section, we will numerically demonstrate the performance of the presented algorithms for the photon spectral imaging problem (PSSI) in the discrete spectrum. Therefore, 2D transforms and dictionaries are exploited as the scenes are uncorrelated along the spectral dimension.

4.1.2.1 Simulation Setting

The number of measurements and the number of unknown images are chosen as $K = S = 3$. In the photon sieve design, the outer diameter of the photon sieve is chosen as 25 mm and the diameter of the smallest hole as 5 μm. For the spectral imaging system, wavelengths emitted from a polychromatic source is taken as $\lambda_1 = 33.3$ nm, $\lambda_2 = 33.4$ nm, and $\lambda_3 = 33.5$ nm. The photon sieve system takes measurements at the focal planes of each of these wavelengths.

Test images are taken from solar images of AIA 335 telescope of NASA [68]. A sample set of 128×128 solar EUV images shown in Fig. 4.4a, 4.4e and 4.4i are used as the true spectral images at these wavelengths, and the measurements are generated using the forward model in Eq. (2.2) with signal-to-noise ratios (SNRs) of 15, 20, 25, 30, 35 and 40 dB. Each measurement is the superposition of differently blurred spectral images. Here Fig. 4.2a, 4.2e and 4.2i display measured intensities with 20 dB SNR. The contributions from each spectral band to these measurements are shown in Fig. 4.2b, 4.2c, 4.2d, 4.2f, 4.2g, 4.2h, 4.2j, 4.2k and 4.2l. These contributions are acquired by convolving point spread functions in Fig. 4.3a,4.3b,4.3c,4.3d,4.3e,4.3f,4.3g,4.3h and 4.3i with original images in Fig. 4.4a, 4.4e and 4.4i.

We use a sparsifying transform, patch-based dictionary, and convolutional dictionary for regularization. Each requires different parameter selections.

Parameter selection with Sparsifying Transform: The parameters of the algorithm are adjusted, as shown in Table 4.1. Here $\beta = 1$ and does not require a parameter optimization. Moreover, $\rho = 10\lambda$ is adjusted. Note that since β is set as constant, λ decreases with decreasing noise level (increasing SNR). Namely, decreasing noise requires less dependence on the prior term and more dependence on the data fidelity term.

Table 4.1: Parameter Selection with Sparsifying Transform for various SNRs

Updates	15 dB	20 dB	25 dB	30 dB	35 dB	40 dB
λ	10^{-2}	10^{-2}	3×10^{-3}	10^{-3}	3×10^{-4}	10^{-4}

Parameter selection with Patch-Based Dictionary: Here, the constant weight 1 is used for dictionary representation term $||\mathbf{Dz}-\mathbf{x}||_2^2$. Hence, all other paramters require optimization. The parameters are adjusted as (λ,β) = $(0.05, 50)$ for 15 dB, where β increases with increasing SNR (decreasing noise). Decreasing the noise requires more dependence on data fidelity term and less dependence on the noise term. Also other parameters set as ρ = 1 and $\sigma = 1$. The parameter adjustments of (λ,β) are displayed in Table 4.2

Table 4.2: Parameter Selection with Patch-Based Dictionary for various SNRs

Updates	15 dB	20 dB	25 dB	30 dB	35 dB	40 dB
λ	0.05	0.05	0.05	0.05	0.05	0.05
β	50	100	250	500	2000	3000

Parameter selection with Convolutional Dictionary: Here, the parameters are adjusted as (λ,β) = $(0.2, 1)$ for 15 dB, where β is increases with increasing SNR. Also other parameters set as ρ = $50\lambda + 0.5$ and $\sigma = 1$. The parameter adjustments of (λ,β) are displayed in Table 4.3

Table 4.3: Parameter Selection with Convolutional Dictionary for various SNRs

Updates	15 dB	20 dB	25 dB	30 dB	35 dB	40 dB
λ	0.2	0.2	0.15	0.15	0.15	0.15
β	1	2	4	8	12	50

Image reconstruction performances concerning different sizes for randomly initialized dictionaries have been compared to determine the optimal size of the dictionaries. The best dictionary size is determined as 36×36 for patch-based dictionaries where patch size is 6×6. Image reconstruction with $K = P = 3$ takes 230 seconds on a computer with 8 GB of RAM and i7 7500U 2.70 GHz CPU. A convolutional dictionary requires determining the number of dictionary filters. The best convolutional dictionary size is chosen as 12×12 and the number of dictionary filters is set to $M = 4$. A single reconstruction takes approximately 20 seconds even with online dictionary learning, which is substantially lower than the patch-based alternative. Fixed sparsifying transforms do not require size optimization, and reconstruction time is around 20 seconds.

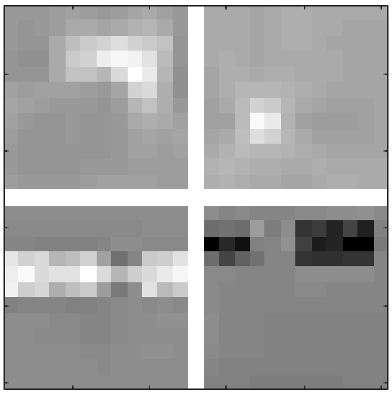

(a)

Figure 4.1: (a) 4 Dictionary filters trained with solar images from Telescope AIA335

To reconstruct images, fixed sparsifying transform, patch-based dictionary, and convolutional dictionaries with and without Tikhonov regularization are exploited.

- Dictionaries are randomly initialized when we perform online learning for both patch-based and convolutional dictionaries.
- When online learning is not performed, an inverse DCT has been exploited for the patch-based dictionary.
- We use a patch-based dictionary trained with 262144 patches extracted from 16 images shown in Fig. B.3a by using K-SVD algorithm. (K-SVD is a state-of-the-art dictionary learning algorithm for patch-based dictionaries [18]).
- When online learning is not performed, the convolutional dictionary is trained with the same solar images in Fig. B.3a and the resulting convolutional dictionary is displayed in Fig. 4.1a. To train the convolutional dictionary, we use the algorithm presented in Section 3.4 by removing the image update steps.

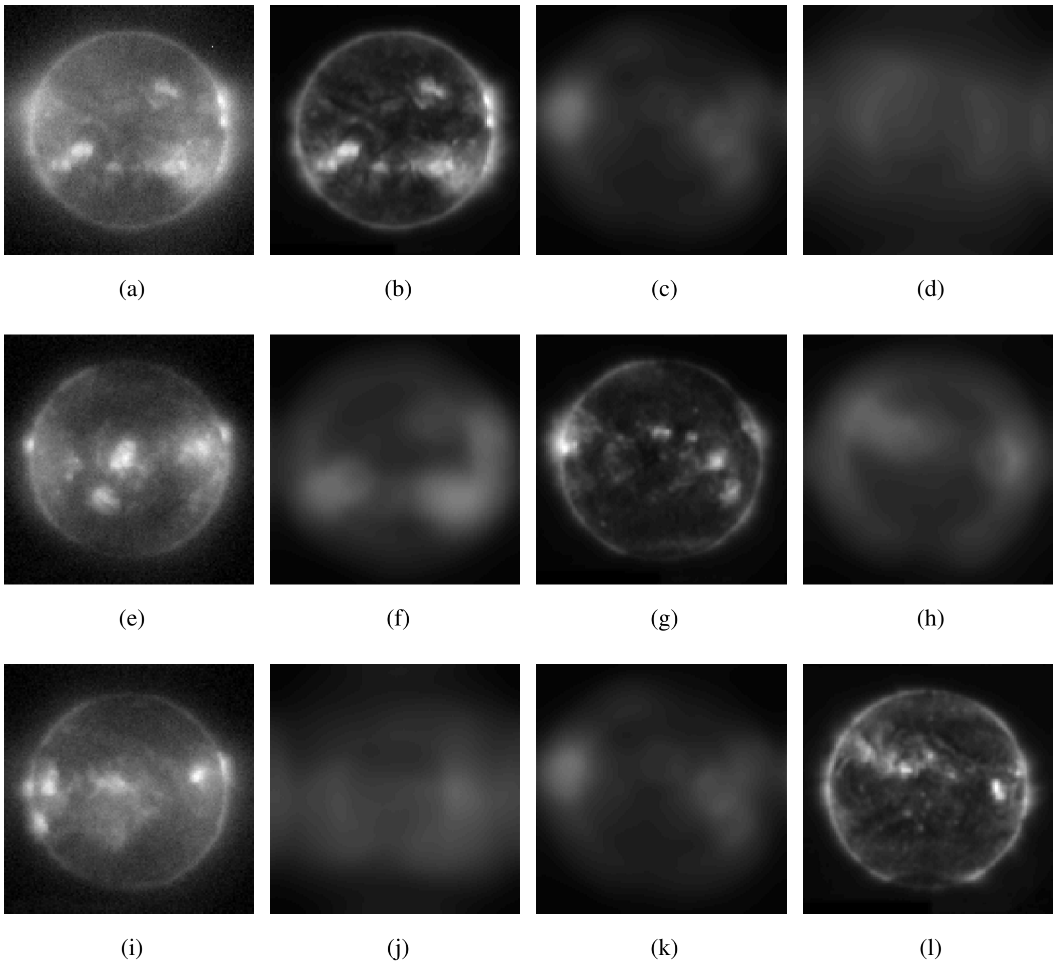


Figure 4.2: (a) Measured intensities for 20 dB SNR at the first focal plane (a), at the second focal plane (e), and at the third focal plane (i), the underlying images of the first source at the first, second, and third focal planes (b)-(d), the underlying images of the second source at the first, second, and third focal planes (f)-(h), and the underlying images of the third source at the first, second, and third focal planes (j)-(l).

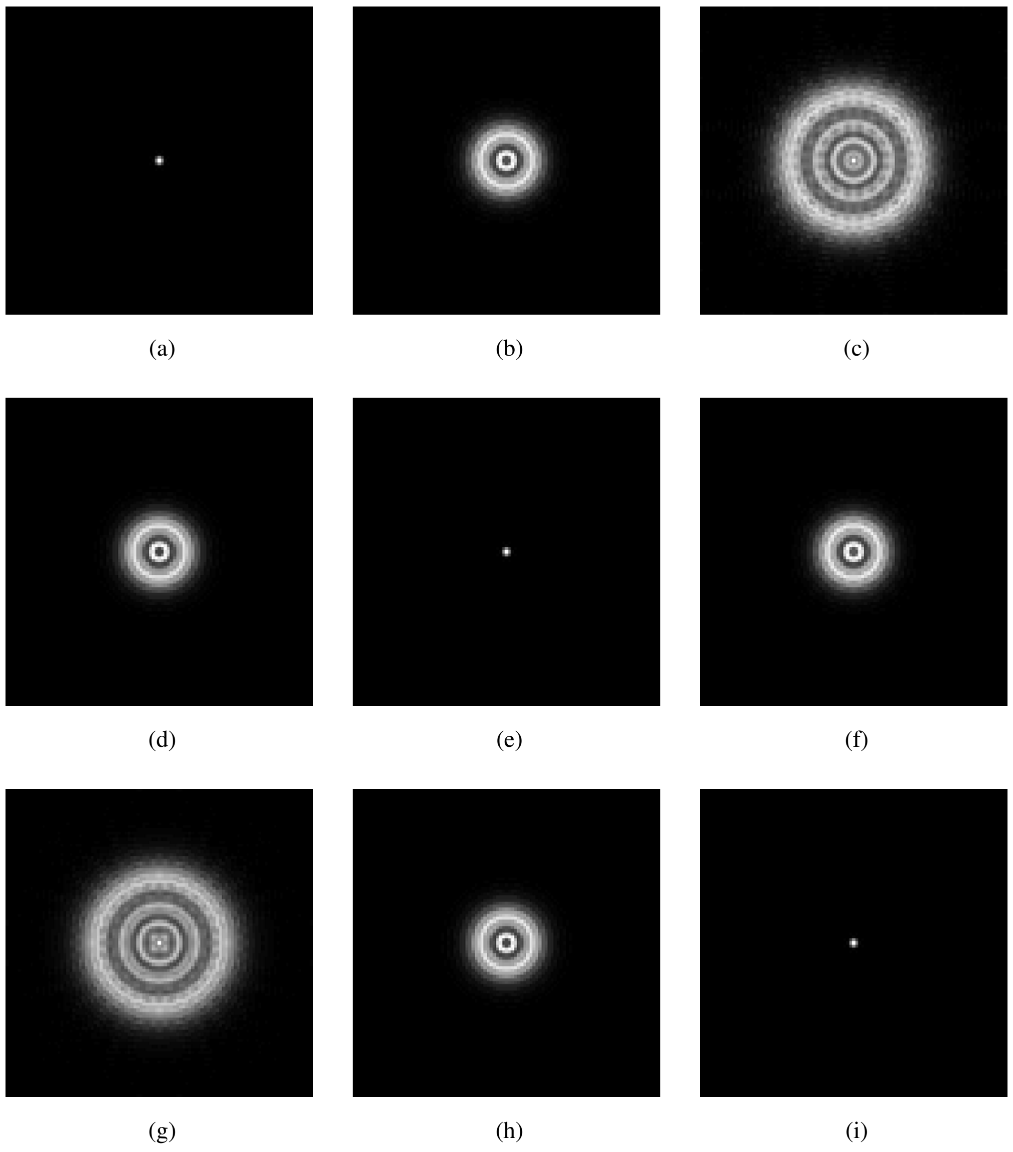


Figure 4.3: Sampled and zoomed point-spread functions of the system for the focused and defocused cases (a)-(i), respectively.

Table 4.4: Reconstruction PSNR and SSIM at different measurement SNRs for Three Algorithms

SNR (dB)	TV Transform	Wavelet Transform	PatchDic Inverse DCT	PatchDic KSVD	PatchDic Online Learning	ConvDic without Learning	ConvDic Online Learning	ConvDic Tikhonov No Learning	ConvDic Tikhonov Learning
15	30.57/0.80	28.22/0.68	30.21/0.81	30.76/0.81	31.33/0.80	30.33/0.80	30.68/0.82	30.55/0.81	30.95/0.83
20	32.62/0.88	31.34/0.85	31.80/0.85	32.45/0.86	32.97/0.86	32.17/0.85	32.57/0.87	32.18/0.86	32.67/0.87
25	34.22/0.91	34.47/0.86	33.61/0.88	33.81/0.88	34.56/0.89	34.13/0.89	34.50/0.91	34.36/0.90	34.51/0.91
30	35.74/0.94	33.69/0.88	35.12/0.91	35.42/0.91	36.23/0.93	36.00/0.93	36.42/0.94	36.43/0.93	36.52/0.94
35	37.58/0.96	34.95/0.91	36.38/0.93	36.68/0.94	36.67/0.94	37.57/0.95	37.88/0.96	37.80/0.95	38.01/0.96
40	39.57/0.97	36.37/0.93	38.09/0.95	38.42/0.95	39.21/0.96	39.21/0.96	39.80/0.97	39.63/0.97	39.83/0.97

4.1.2.2 Simulation Results

In Table 4.4, we present the image reconstruction PSNR and SSIM values for different SNRs, i.e. 15 dB to 40 dB with 5 dB steps for comparison for photon sieve spectral imaging problem. These values are obtained as taking the average of 10 Monte-Carlo results for 5 different solar image sets. As each set consists of three images, we also take the average of the results for three reconstructed images. We compare the results of isotropic TV and Wavelet transform for sparsifying transforms. Furthermore, we compare the performance of the inverse DCT, and a dictionary learned via the K-SVD algorithm for the patch-based dictionary. We refer to the algorithm with patch-based dictionaries as PatchDic. We also include a randomly initialized dictionary which is adaptively learned from the data. For a regular convolutional dictionary (ConvDic), we evaluate the performance of the dictionaries with/without online learning. We repeated the same experiments by adding Tikhonov regularization.

- Reconstruction PSNR is above 30 dB even when the input SNR is 20 dB and increases significantly with increasing SNR value.
- Table 4.4 suggests that online-dictionary learning increases the image reconstruction quality for both patch-based and convolutional dictionaries.
- Tikhonov regularization increases the performance of the convolutional dictionary for both cases.
- The highest performance is attained using patch-based and convolutional dictionaries with online learning; however, the image reconstruction time of patch-based dictionary is around 10 times higher than the sparsifying transform and the convolutional dictionary.
- When online dictionary learning is available, the convolutional dictionary with Tikhonov regularization also has comparable performance with the patch-based one.
- Although sparsifying transforms and convolutional dictionaries have similar reconstruction times, the convolutional dictionary has slightly better performance.

The reconstructed images using isotropic TV reqularization are shown in Fig. 4.4b,4.4f and 4.4j with sparsifying transform, 4.4c,4.4g and 4.4k with patch-based dictionary (online learning) and 4.4d,4.4h and 4.4l with convolutional dictionary (online learning), for the three sources, together with the original scenes in Fig. 4.4a,4.4e and 4.4i for comparison, when the input SNR is 20 dB. Visual inspection shows that the characteristic features of each spectral image are recovered successfully from the noisy photon sieve measurements. However, the reconstruction with convolutional dictionary is visually more successful than other counterparts.

4.1.2.3 Dictionary Size Selection of Synthesis Priors

In this section, we present the dictionary size selection of synthesis priors. Overall results are previously presented, but we did not detail how we selected dictionary sizes. Since solar images from AIA 335 [68] has very similar characteristics, we select three images in Fig. 4.4b,4.4f and 4.4j as a sample test set. Using these images, we perform our experiments with dictionaries having different sizes.

Patch-Based Dictionary In order to determine the optimal size of the dictionaries, image reconstruction performances concerning different sizes of dictionaries have been compared. Table. 4.5 and 4.6 illustrate PSNR and SSIM values for different sizes of randomly initialized dictionaries for 20 and 30 dB respectively. We also show reconstruction time in the same table. Each result is an average of 10 Monte Carlo realizations for three reconstructed images. When the dictionary size is changed, the highest performance attained at a dictionary sizes of 36×36, 49×49 and 64×64. Since the lowest reconstruction time is obtained with a dictionary of size 36×36, the dictionary filter size is determined as 36×36 for all experiments. A single reconstruction takes approximately 230 seconds on a computer with 8 GB of RAM and i7 7500U 2.70 GHz CPU.

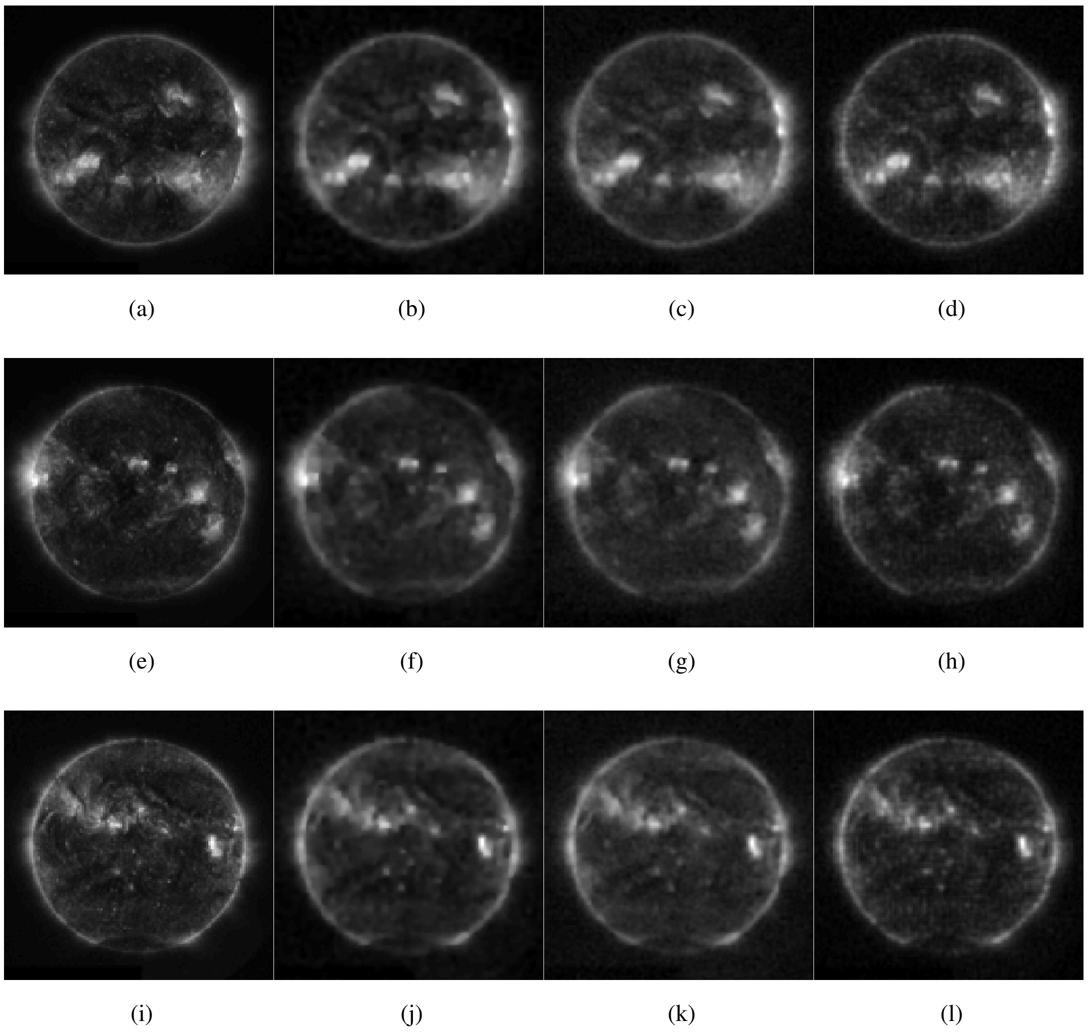


Figure 4.4: (a),(e),(i) Original images (b),(f),(j) Reconstructed images under with sparsifying transform (c),(g),(k) Reconstructed images via patch-based dictionary with online dictionary learning (d), (h), (l) Reconstructed images with convolutional dictionary with online dictionary learning under 20 dB SNR

Table 4.5: Reconstruction PSNR, SSIM and time of patch-based dictionary with changing the size of dictionary for 20 dB SNR

Size of $\mathbf{D}$	PSNR/SSIM	Reconstruction Time
(4×4)	$26.74/0.48$	90 seconds
(9×9)	$30.87/0.68$	125 seconds
(16×16)	$31.82/0.78$	148 seconds
(25×25)	$32.57/0.83$	174 seconds
(36×36)	$32.76/0.86$	195 seconds
(49×49)	$32.74/0.87$	275 seconds
(64×64)	$32.78/0.88$	340 seconds
(81×81)	$32.29/0.88$	395 seconds
(100×100)	$32.66/0.88$	525 seconds

Table 4.6: Reconstruction PSNR, SSIM and time of patch-based dictionary with changing the size of dictionary for 30 dB SNR

Size of $\mathbf{D}$	PSNR/SSIM	Reconstruction Time
(4×4)	$31.71/0.74$	90 seconds
(9×9)	$34.98/0.84$	125 seconds
(16×16)	$36.05/0.89$	148 seconds
(25×25)	$36.13/0.92$	174 seconds
(36×36)	$36.27/0.93$	195 seconds
(49×49)	$36.18/0.93$	275 seconds
(64×64)	$36.11/0.94$	340 seconds
(81×81)	$35.92/0.94$	395 seconds
(100×100)	$36.12/0.94$	525 seconds

Convolutional Dictionary In order to determine the optimal size and number of dictionary filters, image reconstruction performances concerning different sizes and the number of filters for randomly initialized dictionaries have been compared. Table 4.7 and 4.8 illustrate PSNR and SSIM values for different number of filters and dictionary sizes for 20 and 30 dB SNRs. The increasing number of filters causes a proportional increment in image reconstruction time but does not cause a substantial change in image reconstruction quality for simulations conducted under 20 and 30 dB SNRs. With changing dictionary size, the highest performance attained at a dictionary size of 12×12; however, reconstruction time does not vary significantly. For this, the dictionary filter size is chosen as 12×12, and the number of dictionaries is set to $M = 4$. A single reconstruction takes approximately 20 seconds on a computer with 8 GB of RAM and i7 7500U 2.70 GHz CPU.

Table 4.7: Reconstruction PSNR and image reconstruction time of convolutional dictionary with changing size of dictionary for 20 dB SNR

NOF	8×8	12×12	16×16	20×20	Rec. Time
2	32.45	32.97	32.63	31.43	15 seconds
4	32.27	33.13	32.88	32.84	21 seconds
8	32.08	33.04	32.78	32.52	41 seconds
16	31.91	33.17	32.77	32.43	105 seconds
32	31.56	33.19	32.59	32.51	197 seconds
64	32.37	32.88	32.10	32.32	380 seconds

Table 4.8: Reconstruction PSNR and image reconstruction time of the convolutional dictionary with changing size of dictionary for 30 dB SNR

NOF	8×8	12×12	16×16	20×20	Rec. Time
2	36.23	36.83	36.68	36.11	15 seconds
4	36.18	36.97	36.92	36.38	21 seconds
8	36.21	36.72	36.67	36.28	41 seconds
16	35.39	36.86	36.81	36.42	105 seconds
32	35.67	36.94	36.64	36.37	197 seconds
64	36.06	36.36	36.52	36.19	380seconds

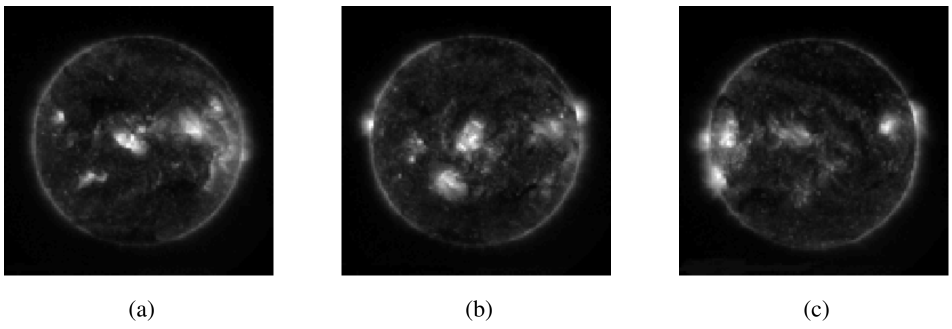


Figure 4.5: (a),(b),(c) Original Images which are used in simulations for various dictionaries and settings

4.1.2.4 Analysis of Various Dictionaries and Settings

The simulations presented in this section are conducted to determine the optimal dictionary training dataset. Since solar images from AIA335 telescope [68] have similar characteristics, we performed these experiments for a sample image set in Fig. 4.5a, 4.5b and 4.5c. These test images are taken from AIA 335 telescope of NASA.

Patch-Based Dictionary In this section, image reconstruction and image representation qualities with different dictionaries are evaluated for images in Fig. 4.5a, 4.5b and 4.5c. The image reconstruction quality is measured via peak-signal-to-noise ratio (PSNR) value between original and reconstructed images. On the other hand, image representation quality measures the PSNR between represented image $\mathbf{Dz}$ and reconstructed image $\mathbf{x}$. The image "representation" quality is computed to observe if there is a direct relation between image reconstruction quality and image representation quality.

For dictionary training, the algorithm presented in Chapter 3 for patch-based image reconstruction has been used. Here, we eliminate the data fidelity term and image update step from the algorithm to modify the algorithm for dictionary learning. Additionally, we also use the K-SVD algorithm for dictionary training for one setting.

Six different dictionaries have been utilized in this experiment set. The first one is a random dictionary. The second, the third and the fourth ones are dictionary trained with MATLAB standard images in Fig. B.1a, dictionary trained with various solar images as shown in Fig. B.2a, dictionary trained with AIA 335 telescope images as displayed in Fig. B.3a respectively. These dictionaries are learned via the algorithm presented in Chapter 3. The fifth dictionary is trained with AIA 335 telescope images by using the K-SVD algorithm. Lastly, an inverse DCT transform, which has the same size as other dictionaries have been exploited.

Size of each dictionary is determined as 36×36 and each patch is of size 6×6. Here, four settings are evaluated based on image reconstruction performance. The first setting excludes dictionary learning updates from the algorithm while the second setting includes online dictionary learning. In the third setting, dictionaries are updated once in five iterations. In the last setting, we do not solve the PSSI problem, but we separately measure the image representation quality of three images and take the average. Hence, this setting evaluates only the image representation performance of different dictionaries.

- Without online dictionary learning, the inverse DCT dictionary results in the highest PSNR. Since DCT is a popular and successful transformation for imaging problems, this result is expected. Only the dictionary trained via the K-SVD algorithm has a similar performance with the inverse DCT. Other dictionaries, which are trained with the algorithm in Chapter 3, cannot compete with the performance of the inverse DCT as the dictionaries might not be trained properly.

- Online dictionary learning increased the reconstruction performance for all dictionaries. Image reconstruction and representation PSNRs are similar for each dictionary. It shows that dictionary initialization is not important when dictionaries are updated online.

- Partially updating the dictionary does not have a significant impact on the reconstruction and representation of PSNR values.

- All dictionaries except inverse DCT and the one trained with K-SVD are deficient in terms of "representation" performance. It demonstrates that other dictionaries are not properly trained.

Table 4.9: Reconstruction PSNR of patch-based dictionary with different dictionaries for 30 dB SNR

Patch-Based (30 dB)	Reconstruction without Online Learning	Reconstruction with Online Learning	Reconstruction with Partial Dictionary Learning	Representation
Random - Rec	N/A	37.29	37.35	N/A
Random - Rep	N/A	34.13	34.37	N/A
Standard - Rec	34.51	37.47	37.43	N/A
Standard - Rep	23.87	34.67	35.12	26.12
Sun - Rec	34.63	37.49	37.47	N/A
Sun - Rep	24.01	34.92	35.18	26.73
AIA335 - Rec	35.60	37.43	37.51	N/A
AIA335 - Rep	25.72	35.02	35.39	27.47
AIA335(KSVD) - Rec	37.12	37.48	37.46	N/A
AIA335(KSVD) - Rep	34.72	34.98	35.31	38.47
DCT - Rec	37.28	37.31	37.37	N/A
DCT - Rep	34.86	34.93	35.42	38.19

Convolutional Dictionary In this section, image reconstruction and image representation qualities with different dictionaries is evaluated for images in Fig. 4.5a, 4.5b and 4.5c. Four different dictionaries has been utilized in this experiment set; random dictionary set, dictionary filter set in Fig. B.1b trained with MATLAB standard images in Fig. B.1a , dictionary filter set in Fig. B.2b trained with solar images from various wavelengths as shown in Fig. B.2a and dictionary filter set in Fig. B.3b trained with AIA 335 telescope images as displayed in Fig. B.3a.

Here, the dictionary size is chosen 12×12, and dictionary filter number is set as 36. (We can use different number of dictionary filters, it effects the implementation time but does not have significant impact on the reconstruction quality as shown in Table 4.7.) For dictionary training, the algorithm presented in Chapter 3 for convolutional image reconstruction has been used. Here, we eliminate the data fidelity term and image update steps from the algorithm to modify the algorithm for the dictionary learning. Here, four settings are evaluated based on image reconstruction performance. In the first setting, we exclude the dictionary learning steps from the algorithm while in the second setting, we adaptively update the dictionary in the image reconstruction algorithm. In the third setting, dictionaries are updated once in five iterations. The last setting includes only image representation performance of different dictionaries.

- Without online dictionary learning, the highest image reconstruction PSNR is obtained for the dictionary trained with the images of the AIA335 telescope. It shows that the selection of the dictionary is significant if the dictionary is not updated online. The image reconstruction quality increases with the similarity between the test and the training images.
- When the dictionary is updated online, all dictionaries (trained with standard images, different solar images, AIA335) have similar performances. Consequently, dictionary initialization is not critical with online dictionary learning.
- Online dictionary learning increased the performance of image reconstruction, as data adaptivity increases reconstruction performance.
- Partially updating the dictionary decreases both reconstruction and representation PSNRs. Therefore, this algorithm highly depends on the online update of the dictionary.
- In the last case, all dictionaries have similar representation performances. It shows that all dictionaries are learned properly with a proposed dictionary learning algorithm.

Table 4.10: Reconstruction PSNR of convolutional dictionary with different dictionaries for 30 dB SNR

Convolutional (30dB)	Reconstruction without Online Learning	Reconstruction with Online Learning	Reconstruction with Partial Dictionary Learning	Representation
Random - Rec	N/A	37.32	36.46	N/A
Random - Rep	N/A	33.69	31.60	N/A
Standard - Rec	36.28	37.28	36.97	N/A
Standard - Rep	35.50	33.84	32.81	38.88
Sun - Rec	36.41	37.24	37.24	N/A
Sun - Rep	35.78	34.26	33.12	40.63
AIA335 - Rec	37.12	37.36	37.25	N/A
AIA335 - Rep	36.18	34.27	33.52	41.00

4.1.3 Numerical Results for the 3D Case

In this section, we will numerically demonstrate the performance of the presented algorithms for denoising and photon sieve spectral imaging problems (PSSI) in a continuous spectrum. Therefore, three-dimensional transforms and dictionaries are exploited as the scenes are correlated along the spectral dimension.

4.1.3.1 Image Denoising

Image denoising can be considered as a very simple convolutional inverse problem when the convolution filter $h[n_1, n_2, s] = \delta[n_1, n_2, s]$. Before solving the more complex inverse problems, we evaluate the image denoising performance of the presented methods with analysis and synthesis priors. Image denoising gives an intuition about the representation quality of the priors. We consider a dataset of size $256 \times 256 \times 16$ (16 wavelengths from $510 - 660$ nm with 10 nm spacing) in the visible band that was obtained from an online hyperspectral image database referred as Objects [69], Beads [70], Flowers [70], Pompoms [70] and Threads [70]. The measurements are generated by adding Gaussian noise with signal-to-noise ratios (SNRs) of 10, 20, and 30 dBs.

We first exploit a three-dimensional fixed transform : Kronecker basis $\mathbf{T} = \mathbf{T_1} \bigotimes \mathbf{T_2}$ where $\mathbf{T_1}$ is the Kronecker product of 2D Symmlet-8 basis and $\mathbf{T_2}$ is the 1D cosine basis. This is widely used in three-dimensional imaging as it performs better than other transforms such as 3D TV, 3D DCT [13]. Second, we perform the algorithm with online convolutional dictionary learning. Here, we use a convolutional dictionary, which is initialized with pre-trained dictionaries. For pre-training, the dictionary filters are randomly initialized. The dictionaries are coarsely trained with 25 spectral data cubes of size $256 \times 256 \times 16$, which are shown in Fig. B.5. The dictionary learning algorithm introduced in [31] is exploited for pre-training. Note that, by removing data-fidelity term and image update steps from the image reconstruction algorithm with convolutional prior in Chapter 3.4, we can obtain this convolutional dictionary learning algorithm.

Parameter selection with 2D Symmlet $\bigotimes$ 1D DCT transform is shown in Table 4.11

where $\beta = 1$ and with convolutional dictionary is shown in Table 4.12. Image reconstruction time is around 200 seconds for sparsifying transform and 300 seconds for the convolutional dictionary. The dictionary size is $32 \times 32 \times 5$ of 6 filters.

Table 4.11: Parameter Selection for various SNRs with Sparsifying Transform

Updates	10 dB	20 dB	30 dB
λ	0.1	0.1	0.01
ρ	10	50	25

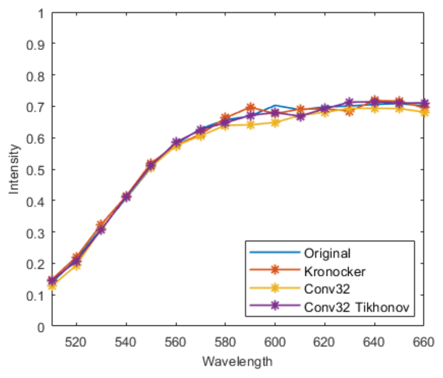
Table 4.12: Parameter Selection for various SNRs with Convolutional Dictionary

Updates	10 dB	20 dB	30 dB
λ	0.001	0.001	0.001
β	0.04	0.2	0.5
ρ	10	10	10
σ	1000	1000	1000

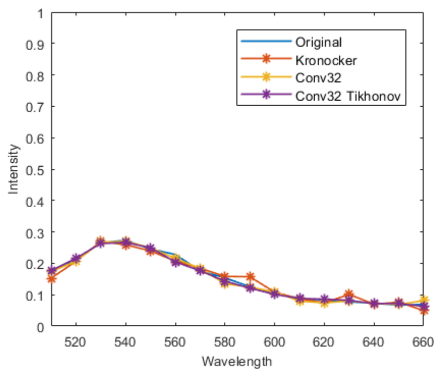
Table 4.13 shows the average reconstruction PSNR, SAM and SSIM values under 10 dB, 20 dB and 30 dB SNRs. These values are obtained as taking an average of 10 Monte-Carlo results for Objects, Beads, Flowers, Pompoms, and Threads. First, image reconstruction quality with the convolutional dictionary is mostly greater than the algorithm with a fixed transform in terms of PSNR and SAM. Furthermore, SSIM levels are close to each other for 2D Symmlet $\bigotimes$ 1D DCT transform and convolutional dictionary. When SNR decreases (for noisy cases), we observe a considerably better image reconstruction performance for the convolutional dictionary. Second, a convolutional dictionary with Tikhonov regularization has similar performance with the regular convolutional dictionary in terms of PSNR, SAM, and SSIM values.

The reconstructed images using 2D Symmlet $\bigotimes$ 1D DCT transform, convolutional dictionary, and convolutional dictionary with Tikhonov regularization are shown in Fig. 4.7a and 4.7b for the sixteen bands, together with the original scenes for comparison, when the input SNR is 10 dB. (Since denoising is a very simple inverse problem, high noise levels are selected to show the reconstruction quality.) We also display the difference between the original image and reconstructed images in the same order. Even though image reconstruction quality is very high for all priors, the convolutional dictionary provides better image reconstruction quality than the sparsifying transform. The visual inspection also suggests that the Tikhonov regularization slightly increases the performance of the convolutional dictionary.

Furthermore, to demonstrate successful recovery along the spectral dimension, we also display the spectrum of the point at $\mathbf{P1}$ and $\mathbf{P2}$ in Figure 4.6a and 4.6b for Objects Data respectively. The reconstructed spectra at these points are plotted with the spectra of the original image. It can be seen that the spectrum is reconstructed better with the convolutional dictionary than the 2D Symmlet $\bigotimes$ 1D DCT transform. We also observe a slight increase in the performance of the convolutional dictionary with Tikhonov regularization.



(a)

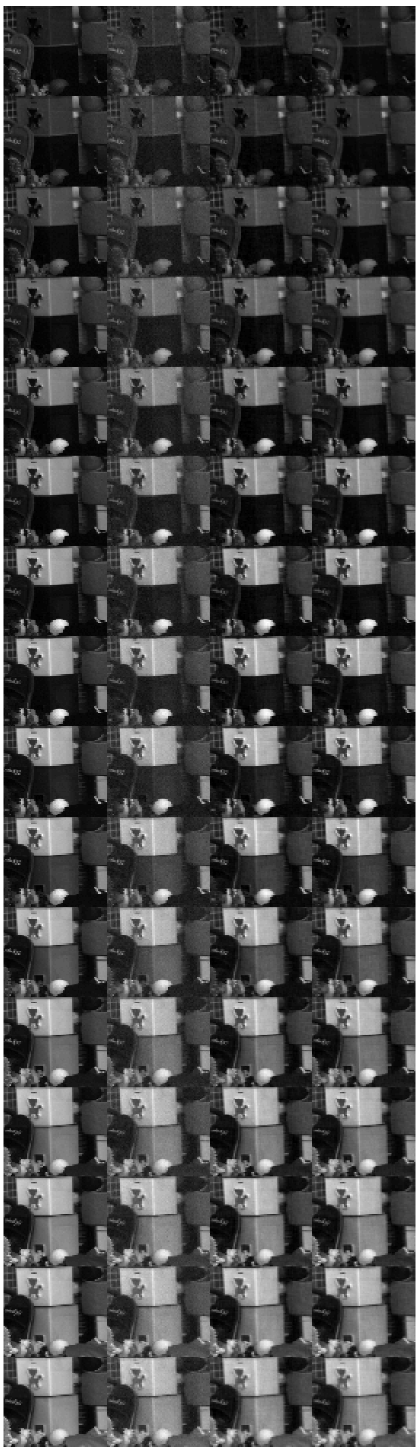


(b)

Figure 4.6: (a) Spectrum for Point 1, (b) Spectrum for Point 2 with 10 dB SNR for Objects

Table 4.13: Denoising performance comparison in terms of PSNR, SAM and SSIM for 2D Symmlet $\bigotimes$ 1D DCT transform and convolutional dictionaries for Objects, Beads, Flowers, Pompoms, and Threads Data

Dataset	SNR (dB)	2D Wavelet + 1D DCT	Convolutional Dictionary	Convolutional Dictionary with Tikhonov
Objects	10	$33.04/5.60°/0.82$	$34.12/3.87°/0.92$	$35.28/3.58°/0.92$
	20	$39.91/2.60°/0.95$	$40.80/2.35°/0.95$	$40.62/2.48°/0.95$
	30	$45.51/1.50°/0.98$	$45.44/1.53°/0.98$	$44.35/1.75°/0.98$
Beads	10	$29.91/10.71°/0.82$	$29.94/8.57°/0.88$	$30.57/7.10°/0.90$
	20	$37.45/4.92°/0.96$	$37.47/4.89°/0.96$	$38.17/4.80°/0.96$
	30	$45.15/2.55°/0.99$	$44.03/3.14°/0.98$	$44.24/2.99°/0.99$
Flowers	10	$34.46/13.03°/0.86$	$35.53/12.23°/0.89$	$35.71/12.01°/0.89$
	20	$41.38/8.72°/0.97$	$41.40/8.74°/0.97$	$41.44/8.66°/0.97$
	30	$48.56/4.65°/0.99$	$48.52/4.52°/0.99$	$48.61/4.49°/0.99$
Pompoms	10	$34.37/6.55°/0.85$	$35.25/6.06°/0.89$	$35.17/5.91°/0.90$
	20	$41.03/2.97°/0.96$	$41.18/3.10°/0.96$	$41.21/2.85°/0.96$
	30	$47.84/1.82°/0.99$	$47.96/1.78°/0.99$	$48.01/1.65°/0.99$
Threads	10	$36.47/6.20°/0.92$	$37.32/6.24°/0.93$	$37.41/6.15°/0.93$
	20	$42.91/2.79°/0.97$	$43.44/2.75°/0.97$	$44.01/2.68°/0.97$
	30	$50.23/1.82°/0.99$	$50.12/1.92°/0.99$	$50.41/1.77°/0.99$


(a)

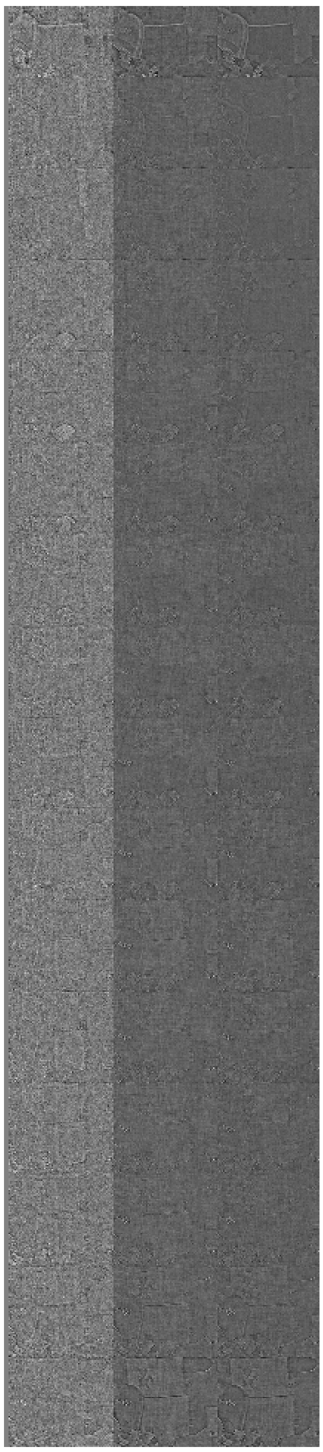

(b)

Figure 4.7: (a) Left-to-right: Original images, Reconstructed spectral images with 2D Symmlet $\bigotimes$ 1D DCT (Kronecker) transform, convolutional dictionary, convolutional dictionary with Tikhonov regularization under 10 dB SNR (b) The difference between original image and reconstructed images

4.1.3.2 Photon Sieve Spectral Imaging Problem with Continuous Spectrum

Here, reconstruction results are presented for a realistic scenario in visible bands. We consider a dataset of size $256 \times 256 \times 16$ (16 wavelengths from $510 - 660$ nm with 10 nm spacing) in the visible band that was obtained from an online hyperspectral image database referred as Objects [69], Beads [70], Flowers [70], Pompoms [70] and Threads [70]. For the photon sieve, a sample design used with the outer diameter of the photon sieve as 3.61 mm and the diameter of the smallest hole as 15 μm, resulting in a focal length of 9 cm at 585 nm. We take measurements at the focal planes corresponding to these wavelengths. The number of measurements $K = 16$ and different SNR levels are considered. The measurements are generated using the forward model in Eq. (2.1) with signal-to-noise ratios (SNRs) of 20, 30 and 40 dBs.

We first exploit a three-dimensional fixed transform for the image reconstruction algorithm: Kronecker basis $\mathbf{T} = \mathbf{T_1} \bigotimes \mathbf{T_2}$ where $\mathbf{T_1}$ is the Kronecker product of 2D Symmlet-8 basis and $\mathbf{T_2}$ is the 1D cosine basis. Second, we use the image reconstruction algorithm with online dictionary learning. This convolutional dictionary is initialized with pre-trained dictionaries. For pre-training, the dictionary filters are randomly initialized, as shown in 4.8a. The dictionaries are coarsely trained with 25 spectral data cubes of size $256 \times 256 \times 16$, which are shown in Fig. B.5. The dictionary learning algorithm introduced in [31] is exploited for pre-training. Note that by removing data-fidelity term and image update steps from the image reconstruction algorithm in Chapter 3.4, we can obtain the convolutional dictionary learning algorithm. The resulting pre-trained dictionary filter set is shown in 4.8b.

Parameter selection with sparsifying transform is shown in Table 4.14 where $\beta = 1$. Here, ρ and σ are iteratively updated. Parameter selection for convolutional dictionary is also presented in Table 4.15.It should be noted that the adjustment of five parameters is a very time-consuming process, as each requires testing on separate intervals. Furthermore, since each iteration takes a long time (at least 2000 seconds), parameter optimization is cumbersome for the algorithm with the convolutional dictionary. Hence, a parameter optimization algorithm can be exploited instead of an exhaustive search. Furthermore, we need to determine the optimal dictionary sizes. The best reconstruction quality is attained via the dictionary of size $32 \times 32 \times 5$ and a number of filters of 6, respectively. Note that the patch-based dictionaries are not exploited in 3D form due to their substantial computational complexity.

Table 4.14: Parameter Selection for various SNRs with Sparsifying Transform

Updates	20 dB	30 dB	40 dB
λ	0.5	0.1	0.01
ρ	250	50	25

Table 4.15: Parameter Selection for various SNRs with Convolutional Dictionary

Updates	20 dB	30 dB	40 dB
λ	0.001	0.001	0.001
β	0.01	0.1	0.2
ρ	10	10	10
σ	1000	1000	1000

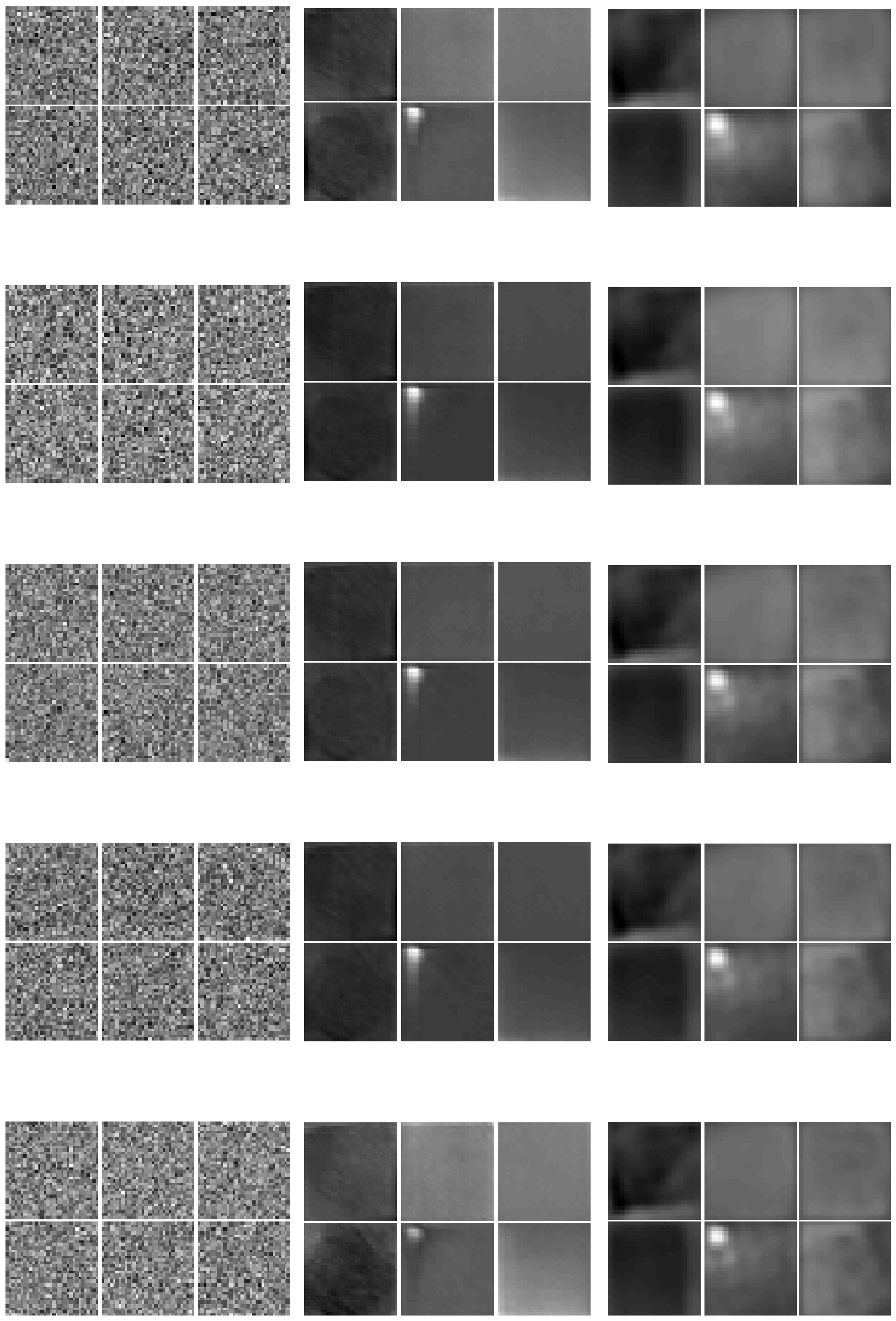
Table 4.16: Parameter Selection for various SNRs with Convolutional Dictionary (Tikhonov Regularization)

Updates	20 dB	30 dB	40 dB
λ	0.001	0.001	0.0005
β	0.01	0.2	0.5
ρ	10	10	10
σ	1000	1000	1000
μ	0.1	0.1	0.1

Here, image reconstruction with the sparsifying transform is faster than the convolutional dictionary. The algorithm with the sparsifying transform takes around 980 seconds, and the algorithm with a convolutional dictionary takes around 2300 seconds on the average for various SNR levels. As we discussed in Chapter 3, the algorithm with the convolutional dictionary has higher computational complexity than the sparsifying transform. On the other hand, the convolutional dictionary with Tikhonov regularization has a very similar image reconstruction time with the regular convolutional dictionary.

Table 4.17: Image reconstruction performance comparison in terms of PSNR, SAM and SSIM for 2D Symmlet $\bigotimes$ 1D DCT transform and convolutional dictionaries for Objects, Beads, Flowers, Pompoms, and Threads Data

Dataset	SNR (dB)	2D Wavelet + 1D DCT	Convolutional Dictionary	Convolutional Dictionary with Tikhonov
Objects	20	$24.82/11.78°/0.82$	$26.71/10.16°/0.77$	$26.82/9.64°/0.84$
	30	$26.18/10.57°/0.88$	$28.42/8.64°/0.85$	$28.54/8.49°/0.89$
	40	$27.76/9.30°/0.89$	$29.63/7.71°/0.89$	$30.15/7.44°/0.91$
Beads	20	$23.16/13.81°/0.79$	$24.21/13.86°/0.82$	$24.94/11.96°/0.84$
	30	$25.66/12.04°/0.88$	$27.23/11.74°/0.90$	$28.10/9.96°/0.90$
	40	$28.11/11.08°/0.93$	$29.83/10.45°/0.93$	$29.46/9.03°/0.94$
Flowers	20	$27.94/20.06°/0.77$	$28.19/21.20°/0.78$	$28.31/19.76°/0.79$
	30	$31.09/16.05°/0.87$	$30.47/18.64°/0.85$	$31.19/16.86°/0.88$
	40	$33.46/15.04°/0.92$	$33.12/14.94°/0.92$	$34.09/13.83°/0.93$
Pompoms	20	$28.02/9.40°/0.83$	$28.54/11.50°/0.81$	$28.48/10.43°/0.83$
	30	$29.29/9.01°/0.87$	$29.57/10.80°/0.86$	$30.20/9.42°/0.88$
	40	$30.62/8.85°/0.91$	$30.90/9.83°/0.89$	$31.41/8.71°/0.92$
Threads	20	$28.83/11.78°/0.84$	$29.51/12.46°/0.84$	$29.39/11.82°/0.85$
	30	$31.14/10.79°/0.90$	$31.82/11.79°/0.87$	$32.13/10.58°/0.90$
	40	$34.04/9.32°/0.94$	$34.12/9.72°/0.93$	$34.29/9.17°/0.95$



(a) (b) (c)

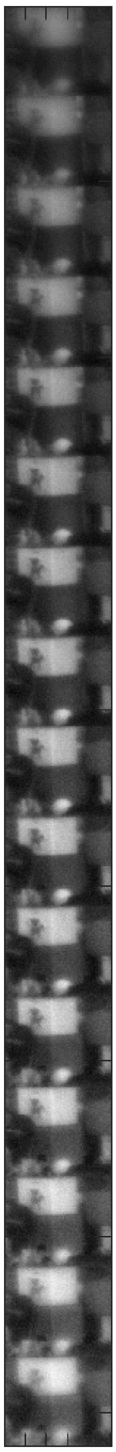
Figure 4.8: (a) Randomly Initialized 6 dictionary filters of size $32 \times 32 \times 5$ (Top-bottom shows bands), (b) Pre-trained dictionary filters, (c) Final dictionary filters for image reconstruction with Objects data in PSSI problem

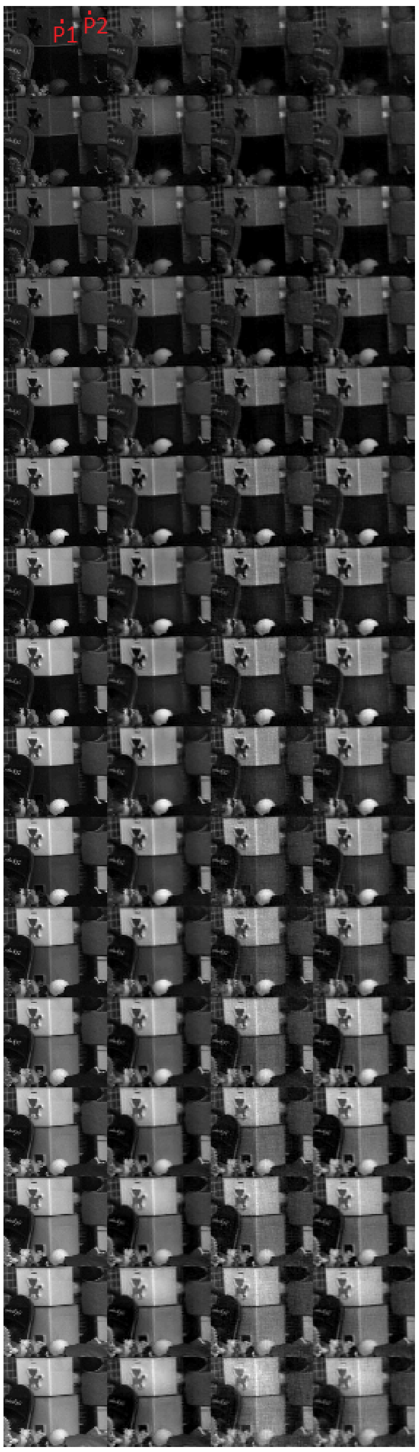
Table 4.17 shows the average reconstruction PSNR, SAM and SSIM values under 20 dB, 30 dB and 40 dB SNRs. These values are obtained as taking an average of 10 Monte-Carlo results for Objects, Beads, Flowers, Pompoms and Threads. First, image reconstruction quality with the convolutional dictionary is greater than the algorithm with 2D Symmlet $\bigotimes$ 1D DCT transform in terms of PSNR for all SNR levels. However, SAM and SSIM levels are worse for convolutional dictionary. Here, SAM is a critical performance metric to evaluate image reconstruction quality in the spectral dimension. Therefore, the convolutional dictionary falls behind the 2D Symmlet $\bigotimes$ 1D DCT transform in spectral image reconstruction. Furthermore, SSIM mostly degrades with the artifacts observed on the images, and hence, it can be inferred that regular convolutional dictionary creates some artifacts on the images. Second, convolutional dictionary with Tikhonov regularization has similar performance with the regular convolutional dictionary in terms of PSNR value. Here, SAM and SSIM values are significantly enhanced via Tikhonov regularization. Therefore, usage of Tikhonov regularization is very advantegous for spectral image reconstruction. It also enhances the image reconstruction quality in spatial dimension as it removes the artifacts from the images.

The reconstructed images using 2D Symmlet $\bigotimes$ 1D DCT transform, convolutional dictionary, and convolutional dictionary with Tikhonov regularization are shown in Fig. 4.9b for the sixteen sources, together with the original scenes for comparison, when the input SNR is 20 dB. As visualized in the second column of Fig. 4.9b, 2D Symmlet $\bigotimes$ 1D DCT transform provides smoother structure than the convolutional dictionary (in third column). On the other hand, the convolutional dictionary reconstructs the spectrum better than the 2D Symmlet $\bigotimes$ 1D DCT transform. Even though edges are recovered successfully, artifactual structures are observed in the third column of Fig. 4.9b reconstructed with convolutional dictionary. On the other hand, the artifacts on the images are eliminated by exploiting Tikhonov regularization for the convolutional dictionary, which is shown in the last column of Fig. 4.9b. Since Tikhonov regularization does not increase the computational complexity, it is an efficient way to increase image reconstruction performance.

Moreover, the difference between the original image and the reconstructed images are shown in Fig. 4.9c for convolutional dictionary with Tikhonov regularization, convolutional dictionary and 2D Symmlet $\bigotimes$ 1D DCT transform. In the first column of Fig. 4.9c, edges are more distinctive than the second and third columns, which means that the 2D Symmlet $\bigotimes$ 1D DCT transform is unable to recover the edges successfully. Furthermore, we also observe that the intensity in the first column is higher than the second and third columns; and hence 2D Symmlet $\bigotimes$ 1D DCT transform causes smooth intensity loss. We also observe some artifacts in the second column, which displays the images reconstructed with the convolutional dictionary. As we discussed earlier, these artifacts are removed via Tikhonov's regularization. However, the edges observed in the third column are more distinctive than the second column. Consequently, while Tikhonov regularization removes the artifacts from the reconstructed images, it also slightly deteriorates the performance of the convolutional dictionary for the edge reconstruction. This is an expected result as we inherently enforce smoother reconstructions.

With increasing SNR level, e.g., 30 and 40 dBs, we observe fewer artifacts on the images reconstructed with the convolutional dictionary in Fig. B.6a and Fig. B.7a. Here, the spectral reconstruction performance of the regularization with the convolutional dictionary is clearly superior to the transform. These reconstructions are presented in Appendix B.


(a)



(b)

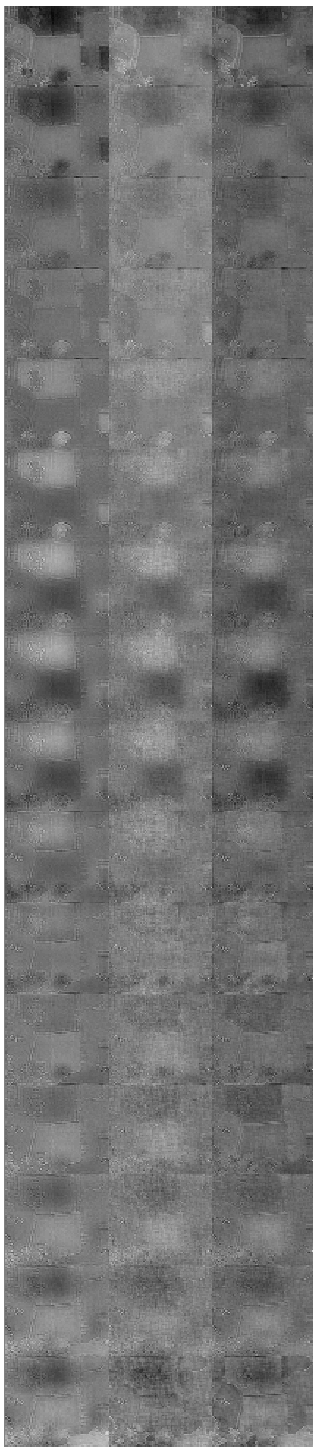

(c)

Figure 4.9: (a) Measured images (b) Left-to-right: Original images, Reconstructed spectral images with 2D Symmlet $\bigotimes$ 1D DCT transform, convolutional dictionary, convolutional dictionary with Tikhonov regularization under 20 dB SNR (c) The difference between original image and reconstructed images

Furthermore, to demonstrate successful recovery along the spectral dimension, we also display the spectrum of the point at $\mathbf{P}_1$ and $\mathbf{P}_2$ in Figure 4.10a and 4.10b for Objects Data respectively. The reconstructed spectra at these points are plotted with the spectra of the original image. It can be seen that the spectrum is reconstructed better with a convolutional dictionary than the 2D Symmlet $\bigotimes$ 1D DCT transform. We also observe a slight increase in the performance of the convolutional dictionary with Tikhonov regularization.

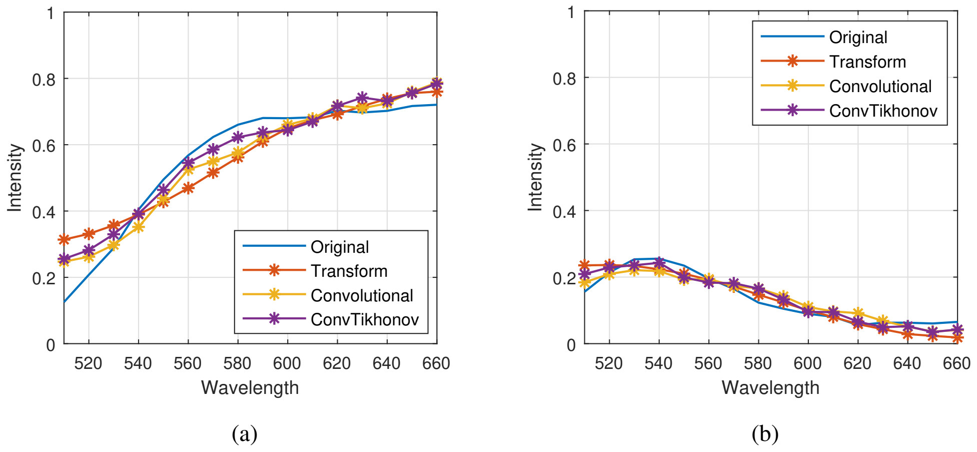
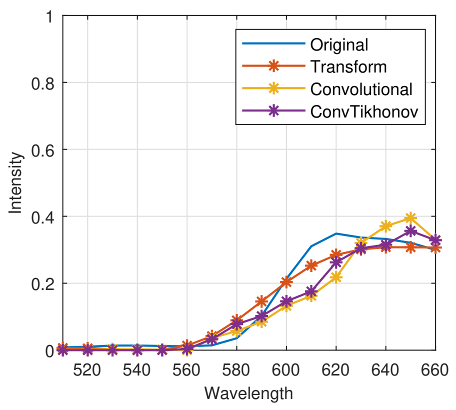
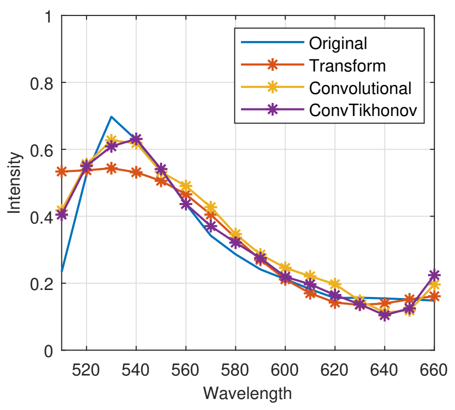


Figure 4.10: (a) Spectrum for Point P1, (b) Spectrum for Point P2 with 20 dB SNR for Objects

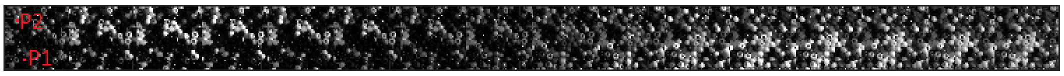
As the Beads data in Fig. 4.11c involves a more complicated structure than the Objects data; both priors can not perfectly reconstruct the spectrum, as shown in Fig. 4.11a and 4.11b. However, we observe a smooth reconstructed spectrum with 2D Symmlet $\bigotimes$ 1D DCT. On the other hand, in Fig. 4.11b, convolutional dictionaries are adaptive to changes in the spectral dimension between $510-540$ nm, while 2D Symmlet $\bigotimes$ 1D DCT transform is poor. Furthermore, in Fig. 4.12a,4.12b, 4.13a,4.13b, 4.14a,4.14b, we again observe a smooth reconstructed spectrum with 2D Symmlet $\bigotimes$ 1D DCT transform reconstructions for Flowers, Pompoms and Threads data which are shown in 4.12c, 4.13c and 4.14c. However, as the convolutional dictionary methods (with and without Tikhonov regularization) are data-adaptive, they mostly reconstruct the spectrum better, but some artifacts are observed in the spectrum.



(a)

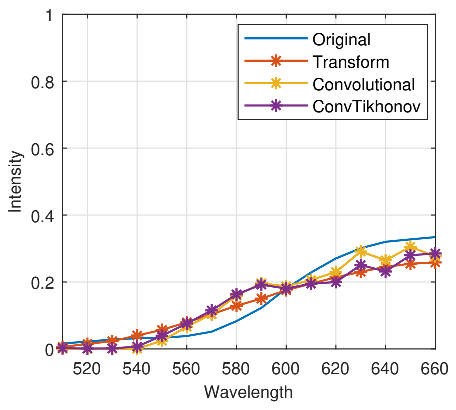


(b)

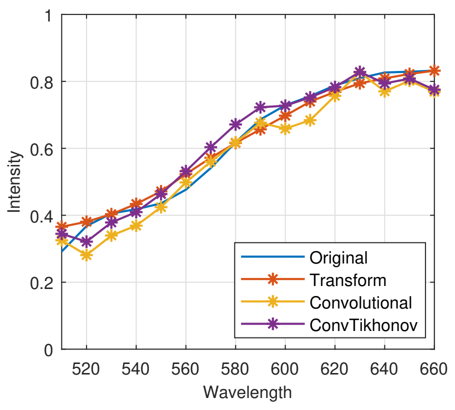


(c)

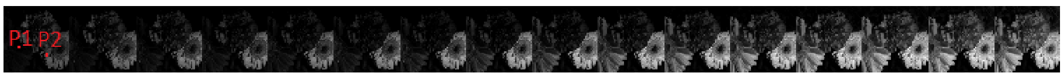
Figure 4.11: (a) Spectrum for Point P1, (b) Spectrum for Point P2 with 20 dB SNR, and (c) Beads data



(a)

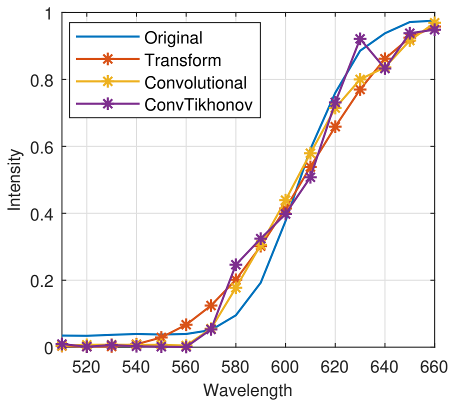


(b)

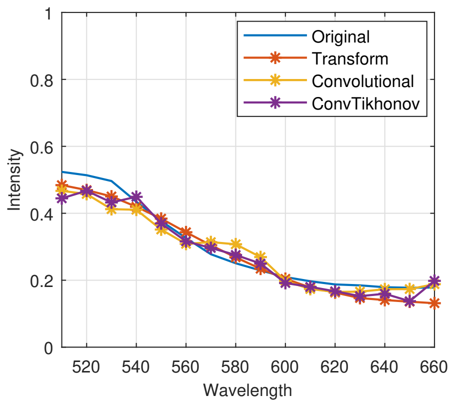


(c)

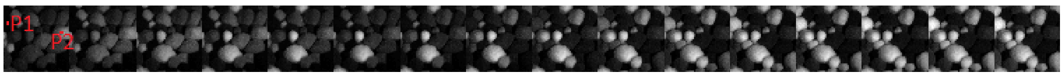
Figure 4.12: (a) Spectrum for Point P1, (b) Spectrum for Point P2 with 20 dB SNR, and (c) Flowers data



(a)

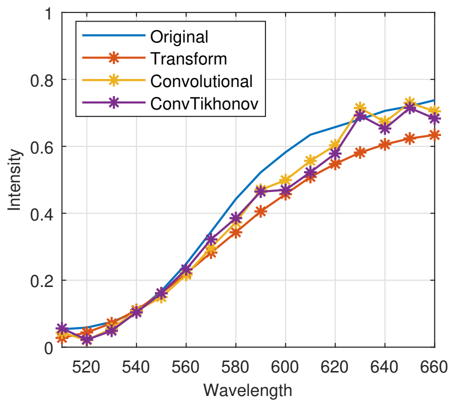


(b)

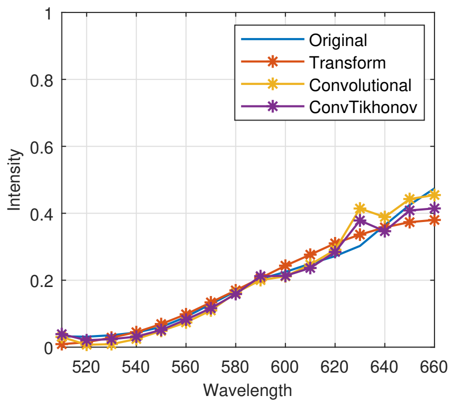


(c)

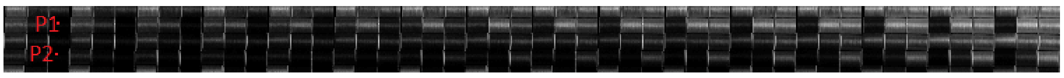
Figure 4.13: (a) Spectrum for Point P1, (b) Spectrum for Point P2 with 20 dB SNR, and (c) Pompoms data



(a)



(b)



(c)

Figure 4.14: (a) Spectrum for Point P1, (b) Spectrum for Point P2 with 20 dB SNR, and (c) Threads data

Dictionary Size Selection for Convolutional Dictionary In this section, we will present how we selected dictionary size for the convolutional dictionary. Here, we also enforce the Tikhonov regularization for the gradient of the sparse codes. Previously, it is mentioned that dictionary size is selected as $32 \times 32 \times 5$ and number of filters is set to 6. The table in 4.18 demonstrates the image reconstruction quality in PSSI problem, for changing size and number of the dictionary filters under 20 dB SNR for the average of 5 Monte Carlo results. Increasing the number of filters causes a huge increase in computation time, however, does not enhance the performance significantly. Best image reconstruction quality in terms of PSNR is attained at dictionary of size $32 \times 32 \times 5$ at 9 filters. The dictionary of size $32 \times 32 \times 5$ and with 6 filters provides better SAM and SSIM quality and its image reconstruction time is substantially lower than the dictionary with 9 filters. Therefore, the most advantegous dictionary in terms of both image reconstruction quality and image reconstruction time is of the size $32 \times 32 \times 5$ and filters 6.

Table 4.18: Reconstruction PSNR, SAM, SSIM, and implementation time with different size and number of dictionary filters for 20 dB SNR

NOF	$16 \times 16 \times 5$	$32 \times 32 \times 5$	$64 \times 64 \times 5$	Rec. Time
3	$25.54/11.68°/0.79$	$25.75/11.14°/0.80$	$26.18/10.67°/0.81$	980 seconds
6	$26.27/10.93°/0.77$	$26.82/9.64°/0.84$	$26.02/10.69°/0.83$	2150 seconds
9	$26.25/10.78°/0.77$	$26.95/9.81°/0.82$	$26.72/9.92°/0.81$	3100 seconds
12	$26.45/10.54°/0.78$	$26.63/9.94°/0.82$	$26.80/9.72°/0.82$	4500 seconds

Note that Table 4.18 only includes the comparison of dictionary size in the spatial dimension. To better observe the effect of dictionary size in the spectral dimension, we also compare different sizes of dictionaries. For constant spatial size 32×32 and 6 dictionary filters, we changed the dictionary size in the spectral dimension from 3 to 9 for odd numbers. Similarly, we observe that the highest image reconstruction quality is also attained with the dictionary of size $32 \times 32 \times 5$ and the number of filters of

Table 4.19: Reconstruction PSNR, SAM, SSIM, and implementation time with different dictionary sizes for 20 dB SNR

$32 \times 32 \times 3$	$32 \times 32 \times 5$	$32 \times 32 \times 7$	$32 \times 32 \times 9$
$25.81/10.99°/0.82$	$26.82/9.64°/0.84$	$26.26/10.67°/0.81$	$26.30/10.17°/0.84$

6. As our reconstructed data cubes have 16 bands in the spectral dimension, 5 bands for dictionary size is a reasonable choice. Dictionary size is significantly smaller than the image size, but it is not too low for the representation of the images.

CHAPTER 5

CONVOLUTIONAL INVERSE PROBLEMS WITH COMPRESSIVE SETTING

5.1 Introduction

Conventional imaging techniques rely on a scanning process to build-up the multi-dimensional (generally 3D) data cube from a series of 2D measurements. It renders these methods inefficient in terms of spatial, temporal, and spectral resolutions and data acquisition time [2]. In order to overcome these limitations, computational 3D image reconstruction emerges as an effective approach. In this approach, 3D image data is reconstructed from indirect two-dimensional measurements [6, 8–14].

Compressive 3D imaging provides an effective way to overcome these limitations by passing on some burden to a computational system. Image reconstruction of an entire data cube is enabled only with few multiplexed measurements. This is made possible by compressive sensing (CS), which relies on two principles: sparsity of the 3D images in some transform domain or sparse representation of images in terms of elements of a dictionary and incoherence of the measurements. It is widely known that 3D images exhibit three-dimensional correlations, which allow three-dimensional sparse representations [13, 67]. Coded compressive imaging systems can provide the incoherence of the measurements.

In this chapter, we develop image reconstruction methods with different priors for compressive image reconstruction and compare their performance. First, the forward

Some parts of this chapter have been submitted for publication [49].

problem is introduced, which includes a coded aperture with the imaging system. As a sample imaging system, compressive spectral imaging with diffractive lenses improved in [13] has been selected. Then, inverse problem is formulated referring the previous discussions in Chapter 2 and 3. The resulting image reconstruction problem is solved with different analysis and synthesis priors. Since we already explained image reconstruction updates for the non-compressive scenario in detail, we only emphasize the differences occurring in the image update step. Then, we evaluate the numerical performance of the developed methods with a sparsifying transform and convolutional dictionary for a compressive spectral imaging problem.

5.2 Forward Problem

In compressive imaging problems, the measurements can be modeled in the following general convolutional form:

$$y_k[n_1, n_2] = \sum_{s=1}^{S} (x_s[n_1, n_2] c_s[n_1, n_2]) * h_{k,s}[n_1, n_2] + w_k[n_1, n_2], \quad 1 \le k \le K \tag{5.1}$$

where n_1,n_2 = $-N/2, ..., N/2 - 1$. Here y_k's denote different measurements, x_s's represents different unknown images to be reconstructed, $h_{k,s}$ denotes the blur acting on the sth image, x_s, for the kth measurement, y_k. Moreover, we have $c_s[n_1, n_2]$ which denotes sth coded aperture with entries 1 or 0 and modulates the corresponding spectral components. If the components are modulated differently, the coded aperture becomes "colored".

$$\mathbf{y} = \mathbf{HCx} + \mathbf{w} \tag{5.2}$$

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_{1,1} & \cdots & \mathbf{H}_{1,S} \\ \vdots & \ddots & \vdots \\ \mathbf{H}_{K,1} & \cdots & \mathbf{H}_{K,S} \end{pmatrix}, \mathbf{C} = \begin{pmatrix} diag(c_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & diag(c_s) \end{pmatrix}, \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ : \\ \mathbf{y}_K \end{bmatrix},$$

$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ : \\ \mathbf{x}_S \end{bmatrix}$. Here $\mathbf{y}_k \in \mathbb{R}^{N^2}$ is lexicographically ordered noisy kth measurement vector, $\mathbf{y}$ is the overall noisy measurement vector by vertically concatenating all the

K measurements. Similarly, $\mathbf{x}_s \in \mathbb{R}^{N^2}$ denotes sth unknown image and the vector $\mathbf{x}$ is obtained by combining unknown images. Here, the matrix $\mathbf{H}_{k,s} \in \mathbb{R}^{N^2 \times N^2}$ is a circular convolution matrix which corresponds to the convolution operation with $h_{k,s}[n_1, n_2]$. The diagonal matrix $\mathbf{C} \in \mathbb{R}^{KN^2 \times SN^2}$ represents the overall coding operation, and has values 0 or 1 along its diagonal. $\mathbf{w}_k$ represents additive white Gaussian noise vector $\mathbf{w}_k \sim N(0, \sigma_k^2)$ and concatenated vertically as $\mathbf{w} = [\mathbf{w}_1^T ... \mathbf{w}_K^T]^T$. In this chapter, the number of measurements (K) is smaller than the number of unknown images (S), which results in an under-determined system of equations.

5.2.1 Compressive Spectral Imaging System with Diffractive Lenses

A sample system which has a forward problem in Eq. (5.1) is compressive spectral imaging system with diffractive lenses (CSID). Here, two different variants of the system with the moving detector and fixed detector is presented.

5.2.1.1 Moving Detector

In this system, first, the image of the scene is coded with a coded aperture, and it passes through a diffractive lens such as photon sieve [11–13]. Each spectral component is focused on different planes, as the diffractive lens has a wavelength-dependent focal length. Then, the measurements are taken at a few measurement planes using a moving detector. Note that each measurement is a superposition of differently blurred and coded spectral bands. This system is a modified and generalized version of the photon sieve spectral imaging system presented in Chapter 2 with a coded aperture. Here coded aperture enables image reconstruction from only "few" multiplexed measurements instead of a full measurement set.

5.2.1.2 Fixed Detector

A very similar system is improved with a fixed detector instead of a moving detector in [13]. In this system, first, the image of the scene is coded with a coded aperture, and it passes through a diffractive lens such as photon sieve [11–13]. Instead of taking

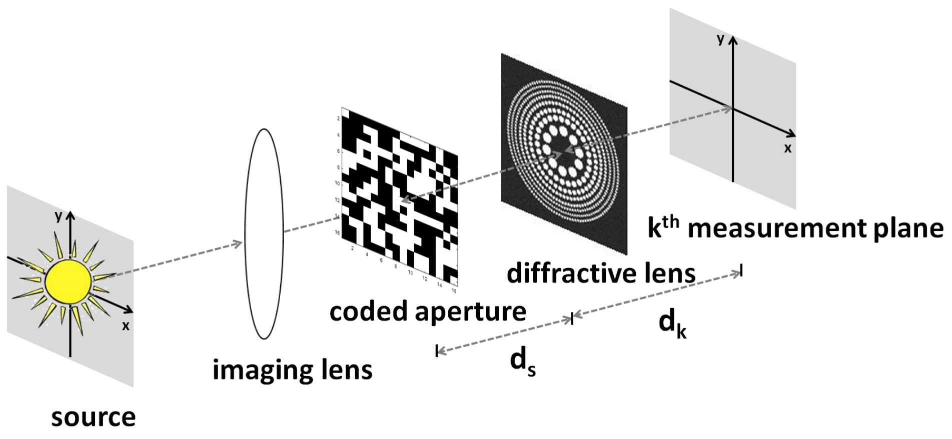


Figure 5.1: Compressive diffractive imaging system with moving detector

measurements at different planes, measurements are taken at a fixed plane with a fixed detector. The focusing behavior of the diffractive lens is adapted for K measurements by changing the photon-sieve diameter in each shot. Such few measurements can be obtained in different ways, such as with programmable diffractive lens, or in a snapshot using multiple diffractive lenses. This system is more time-efficient as it does not require moving the detector. In this chapter, implementations are based on this imaging modality.

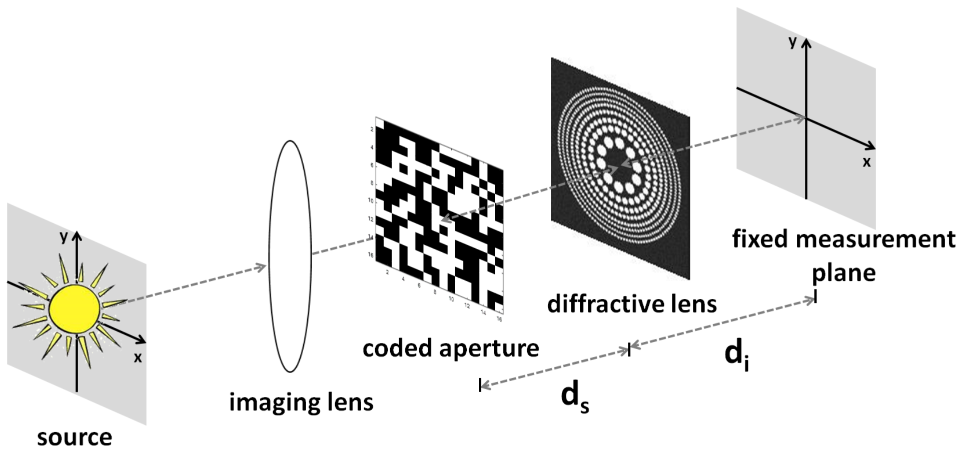


Figure 5.2: Compressive diffactive imaging system with fixed detector

5.3 Inverse Problem

In the inverse problem, the goal is to reconstruct the unknown images $\mathbf{x}$ from their compressive and superimposed measurements, $\mathbf{y}$, which involve their coded and blurred versions. This problem is highly ill-posed. Previously proposed approaches solving convolutional inverse problems are applicable for the compressive scenario. By incorporating the prior information available for the images, we formulate the problem as follows:

$$\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y} - \mathbf{HCx}||_2^2 + \mathcal{R}(\mathbf{x}) \tag{5.3}$$

This is a regularized least squares problem, which can also be related to maximum posterior estimation (MAP). Here the first term controls data fidelity, whereas the second term $\mathcal{R}(\mathbf{x})$ controls how well the reconstruction matches our prior knowledge of the solution, with the scalar parameter β trading off between these two terms.

To solve the inverse problem, the sparsity of natural multidimensional image data in another domain is enforced via $\mathcal{R}(\mathbf{x})$. In the scope of the present work, both analysis and synthesis priors are exploited for $\mathcal{R}(\mathbf{x})$. Since compressive scenario use the correlation information in the third-dimension, only transforms and dictionaries has to be three-dimensional.

5.3.1 Analysis Prior

In this case, sparsifying analysis prior is expressed as

$$\mathcal{R}(\mathbf{x}) = \mathbf{\Phi}(\mathbf{Tx}) \tag{5.4}$$

where $\mathbf{T}$ represents a fixed sparsifying transform matrix. There are popular and powerful choices of the regularizer $\mathbf{\Phi}(.)$. One popular choice is $\mathbf{\Phi}(\mathbf{Tx})$ = $||\mathbf{Tx}||_1$, whose solution is obtained by nonlinear optimization techniques.

5.3.2 Synthesis Prior

For the sake of brevity, we express dictionary-based synthesis prior in a following generalized expression as

$$\mathcal{R}(\mathbf{x}) = \min_{\mathbf{z},\mathbf{D}} ||\mathbf{Dz} - \mathbf{Px}||_{\mathbf{2}} + \lambda||\mathbf{z}||_{\mathbf{1}}. \tag{5.5}$$

where $\mathbf{D}$ denotes the dictionary matrix and $\mathbf{z}$ stands for the sparse code. In the patch-based dictionary model, synthesis prior in Eq. (5.5) is simplified version of patch-based model whose representation is

$$\min_{\mathbf{z}_j,\mathbf{D}_L} \sum_{j=1}^{N^2S}(||\mathbf{D}_L\mathbf{z}_j - \mathbf{P}_j\mathbf{x}||_2^2 + \lambda||\mathbf{z}_j||_1). \tag{5.6}$$

The relation between Eq. (5.5) and (5.6) can be inferred using the following equalities $\mathbf{D}$ = $\mathbf{I}_{SN^2} \bigotimes \mathbf{D}_L$, $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_{N^2S} \end{bmatrix}$, and $\mathbf{P} = \begin{bmatrix} \mathbf{P}_1 \\ \vdots \\ \mathbf{P}_{N^2S} \end{bmatrix}$. Here, $\mathbf{P}_j \in \mathbb{R}^{n^2p \times N^2S}$ extracts vectorized patch of size $n \times n \times p$ from vectorized image $\mathbf{x}$ of size $N \times N \times S$. Also, $\mathbf{z}_j$ denotes sparse code of jth patch, and $\mathbf{D}_L$ represents local dictionary commonly used by all N^2S patches. However, we presented that usage of a 3D patch-based dictionary is computationally inefficient for 3D images.

On the other hand, when $\mathbf{D}$ is the concatenation of circular convolution matrices, synthesis-prior in Eq. (5.5) corresponds to the simplified version of convolutional prior which is expressed as

$$\min_{\mathbf{z}_m,\mathbf{d}_m} ||\sum_{m=1}^{M} \mathbf{d}_m * \mathbf{z}_m - \mathbf{x}||_2^2 + \lambda \sum_{m=1}^{M} ||\mathbf{z}_m||_1. \tag{5.7}$$

By inserting, $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_M \end{bmatrix}$ and $\mathbf{D}$ = [$\mathbf{D}_1$ ... $\mathbf{D}_M$] where $\mathbf{D}_m$ is circular convolution matrix of dictionary filter $\mathbf{d}_m$, generalized synthesis prior in Eq. (5.5) is obtained. Here, $\mathbf{d}_m \in \mathbb{R}^{L^2R}$ is vectorized mth dictionary filter of size $L \times L \times R$ and $\mathbf{z}_m \in \mathbb{R}^{N^2S}$ is the corresponding vectorized sparse code of size $N \times N \times S$. The choice of the dictionary size is significantly smaller than the image size as $L << N$ and $R << S$ to provide circular boundary conditions [30].

5.4 Image Reconstruction Method

The resulting optimization problems for analysis and synthesis approaches are solved for the unknown image $\mathbf{x}$, the sparse coefficient vectors $\mathbf{z}_m$'s and dictionary filters $\mathbf{d}_m$'s by using Alternating Direction Method of multipliers (ADMM) [60]. ADMM is an optimization algorithm that is used in many signal and image reconstruction problems [32–37,40,41]. ADMM solves distributed unconstrained optimization problems by splitting them into sub-problems, which are solved alternatingly. In this section, the solutions are presented for three-dimensional transforms and dictionaries.

5.4.1 Analysis Prior

In this part, we solve the regularized least-square problem in Eq. (5.3) by inserting analysis prior in Eq. (2.7) for regularization term $\mathcal{R}(\mathbf{x})$. Then, the objective function is formulated using augmented Lagrangian in ADMM framework as

$$\min_{\mathbf{x},\mathbf{t}} \frac{\beta}{2}||\mathbf{y}-\mathbf{HCx}||_2^2 + \lambda\Phi(\mathbf{Tx}) \quad \text{s.t.} \quad \mathbf{Tx}=\mathbf{t}. \tag{5.8}$$

This problem is minimized with respect to variables $\mathbf{x}$ and $\mathbf{t}$. ADMM steps for this alternating minimization process have the following form:

$$\mathbf{x}^{l+1} = \arg\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y}-\mathbf{HCx}||_2^2 + \frac{\rho}{2}||\mathbf{Tx}-\mathbf{t}^l+\mathbf{u}^l||_2^2 \tag{5.9}$$

$$\mathbf{t}^{l+1} = \arg\min_{\mathbf{t}} \lambda\Phi(\mathbf{t}) + ||\mathbf{Tx}^{l+1}-\mathbf{t}+\mathbf{u}^l||_2^2. \tag{5.10}$$

$$\mathbf{u}^{l+1} = \mathbf{u}^l + \mathbf{Tx}^{l+1} - \mathbf{t}^{l+1}. \tag{5.11}$$

Here, $\mathbf{u}$ is defined as dual variable of $\mathbf{t}$ and Eq. (5.11) represents the update of the dual variable. Note that, Eq. (5.10) is same update with the update of the non-compressive scenario in Chapter 3. We now explain how to efficiently solve the problem in Eq. (5.9) which we refer as image $\boldsymbol{x}$ update.

5.4.1.1 Image Update

For image update, we need to solve the problem in Eq. (3.5). Since this problem is a least-squares problem, it has a closed-form solution as

$$(\rho\mathbf{I} + \beta\mathbf{C}^H\mathbf{H}^H\mathbf{H}\mathbf{C})\mathbf{x} = (\beta\mathbf{C}^H\mathbf{H}^H\mathbf{y} + \rho\mathbf{T}^H(\mathbf{t} - \mathbf{u})). \tag{5.12}$$

A direct matrix inversion approach for solving the linear system in Eq. 5.12 is not feasible for large-scale data cubes. Furthermore, it should be noted that the block recursive matrix inversion approach that we utilized in the non-compressive scenario is not applicable as well. **C** matrix removes the benefits of the convolution matrices. Consequently, we solve this problem iteratively using the conjugate-gradient method. Similar to previous discussions, forming any of the matrices is not required, which provides huge savings for the computation time.

We first compute $\beta\mathbf{C}^H\mathbf{H}^H\mathbf{y}$ term. 2D Fourier transform of $h_{k,s}[n_1, n_2]$ is computed for $k = 1, 2, .., K$ and $s = 1, 2, .., S$. Next, we take the FFT of $y_k[n_1, n_2]$ for $k = 1, ..., K$. Spesifically, the operations required to form $\mathbf{H}^H\mathbf{y}$ corresponds to summation of element-wise multiplications and taking block inverse fourier transform. $\mathbf{C}^H$ performs elementwise multiplications with coded aperture function $c_s^*[n_1, n_2]$. The second term, $\mathbf{T}^H(\mathbf{t} - \mathbf{u})$, is updated in each ADMM iteration as follows: applying $\mathbf{T}^H$ corresponds to taking the inverse 3D transform of $(t[n_1, n_2, s] - u[n_1, n_2, s])$. Third, $(\rho\mathbf{I} + \beta\mathbf{C}^H\mathbf{H}^H\mathbf{H}\mathbf{C})$x computed. First, $\mathbf{H}^H\mathbf{H}$ is performed in the frequency domain which corresponds to summation of elementwise multiplications. Second, **Cx** performs elementwise multiplications with coded aperture function $c_s[n_1, n_2]$. Then, fourier transform of **Cx** is obtained and $\mathbf{H}^H\mathbf{H}\mathbf{C}\mathbf{x}$ computed via summation of elementwise multiplications in the frequency domain. After going back to time domain, we again perform elementwise multiplications for the multiplication with $\mathbf{C}^H$. Finally, Eq. (5.12) is computed iteratively with conjugate-gradient method.

5.4.2 Synthesis Prior

In this part, we present the regularized least-square problem in Eq. (5.3) by inserting synthesis prior in Eq. (5.5) for regularization term $\mathcal{R}(\mathbf{x})$. Then, the objective function is formulated using augmented Lagrangian in ADMM framework as

$$\begin{aligned}&\min_{\mathbf{x},\mathbf{z},\mathbf{t}} \frac{\beta}{2}||\mathbf{y}-\mathbf{HCx}||_2^2 + \frac{1}{2}||\mathbf{Dz}-\mathbf{Px}||_2 + \lambda||\mathbf{t}||_1 \\ &\text{s.t.} \quad \mathbf{z}=\mathbf{t}.\end{aligned} \tag{5.13}$$

This problem is alternatingly solved with respect to variable $\mathbf{x}$, $\mathbf{z}$ and $\mathbf{t}$. ADMM iterations for this problem are slightly different than the analysis approach in Eq. (3.4). ADMM steps are expressed as

$$\mathbf{x}^{l+1} = \arg\min_{\mathbf{x}} \frac{\beta}{2}||\mathbf{y}-\mathbf{HCx}||_2^2 + \frac{1}{2}||\mathbf{Dz}^{l+1}-\mathbf{Px}||_2 \tag{5.14}$$

$$\mathbf{z}^{l+1} = \arg\min_{\mathbf{t}} \frac{1}{2}||\mathbf{Dz}^{l+1}-\mathbf{Px}||_2||_2^2 + \frac{\rho}{2}||\mathbf{z}-\mathbf{t}^l+\mathbf{u}^l||_2^2 \tag{5.15}$$

$$\mathbf{t}^{l+1} = \arg\min_{\mathbf{t}} \lambda||\mathbf{t}||_1 + ||\mathbf{z}^{l+1}-\mathbf{t}+\mathbf{u}^l||_2^2 \tag{5.16}$$

$$\mathbf{u}^{l+1} = \mathbf{u}^l + \mathbf{z}^{l+1} - \mathbf{t}^{l+1}. \tag{5.17}$$

Here, $\mathbf{u}$ is defined as dual variable of $\mathbf{t}$ and Eq. 5.17 represents its update. These updates except the image update in Eq. (5.14) are exactly same with the updates of non-compressive scenario presented in Chapter 3 for both patch-based and convolutional dictionaries. Note that, online dictionary learning can be performed by following the dictionary learning steps presented in Chapter 3 (Eq. (3.27), (3.28) and (3.44) for patch-based and Eq. (3.58), (3.59) and (3.86) for convolutional dictionary learning). However, they are not repeated for the sake of brevity.

5.4.2.1 Image Update

For image update, we need to solve the problem in Eq. (5.14). As image update is a least-squares problem, it has a closed-form solution:

$$(\mathbf{P}^H\mathbf{P} + \beta\mathbf{C}^H\mathbf{H}^H\mathbf{HC})\mathbf{x} = (\beta\mathbf{C}^H\mathbf{H}^H\mathbf{y} + \mathbf{P}^H\mathbf{Dz}) \tag{5.18}$$

For patch-based dictionary, referring to the patch-based dictionary in Eq. (5.6), $\mathbf{P^HP}$ = $\sum_{j=1}^{N^2S}\mathbf{P}_j^H\mathbf{P}_j$. Here, we consider a $P_j \in \mathbb{R}^{n^2p\times N^2S}$ extracting vectorized patch of size $n \times n \times p$ from vectorized image of size $N \times N \times S$ which makes $\mathbf{P}^H\mathbf{P} = n^2p\mathbf{I}$ is of size N^2S. For convolutional dictionary, $\mathbf{P} = \mathbf{I}$ and $\mathbf{P}^H\mathbf{P} = \mathbf{I}$ of size N^2S. As a result, $\mathbf{P}^H\mathbf{P}$ is multiplication of $\mathbf{I}$ matrix by a constant for both dictionaries.

The computation of the term on the left-hand side of Eq. (5.18) and $\mathbf{C}^H\mathbf{H}^H\mathbf{y}$ are same in image update step of analysis case. The only difference occurs in the computation of $\mathbf{P}^H\mathbf{D}\mathbf{z}$.

For patch-based dictionary, $\mathbf{P}^H\mathbf{D}\mathbf{z} = \sum_{j=1}^{N^2S} \mathbf{P}_j^H\mathbf{D}_L\mathbf{z}_j$ based on the patch-based dictionary in Eq. (5.6) and is computed in the following order: First, local dictionary matrix $\mathbf{D}$ and each sparse vector $\mathbf{z_j}$ is multiplied. Next, to apply $\mathbf{P_j}^H$ operation, each $\mathbf{D}_L\mathbf{z}_j$ vector is reshaped as a 3D patch, inserted into their corresponding patch indices and summed for N^2S patches. Resulting $\mathbf{P}^H\mathbf{D}\mathbf{z}$ is a 3D data cube having same size as the image $x[n_1, n_2, s]$.

For convolutional dictionary, $\rho\mathbf{P}^H\mathbf{D}\mathbf{z} = \mathbf{D}\mathbf{z}$ is computed differently. 3D Fourier transform of $d_m[n_1, n_2, s]$ is computed for $m = 1, 2, .., M$ and $s = 1, 2, .., S$. Next, we take the FFT of $z_m[n_1, n_2, s]$ for $m = 1, ..., M$. Specifically, the operations required to form $\mathbf{D}\mathbf{z}$ corresponds to summation of element-wise multiplications and taking block inverse fourier transform.

5.4.3 Computational Complexity

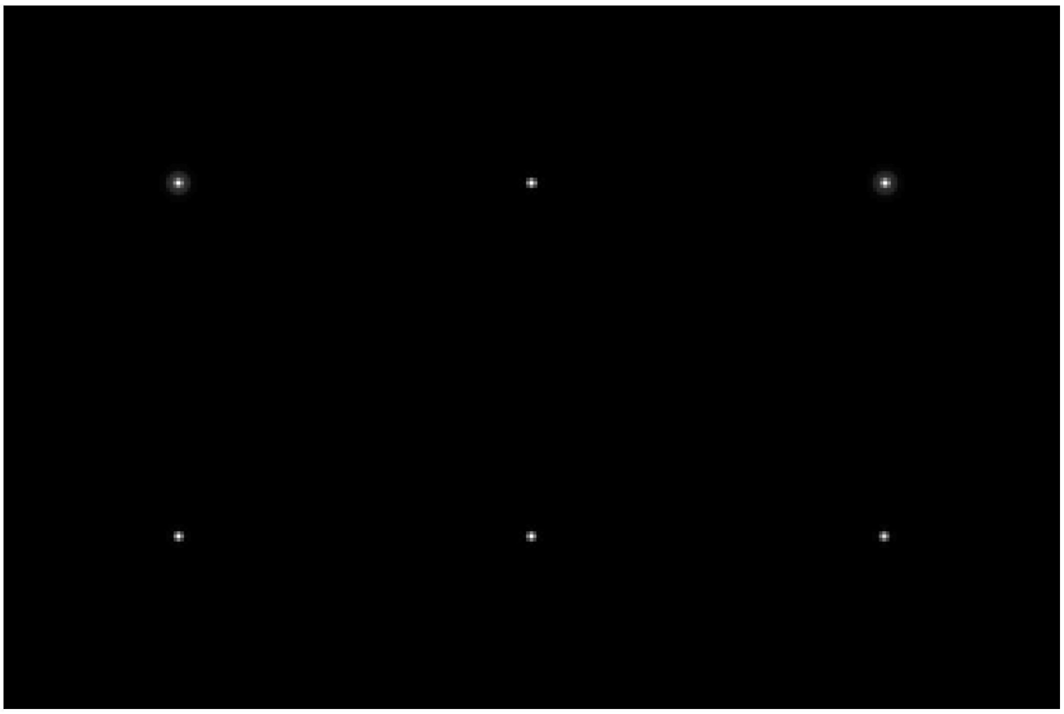
The computational cost of these algorithms is very similar to the algorithms for the non-compressive case. The only difference occurs in the image update step. In the non-compressive case, we take advantage of recursive block inversion algorithms [61]. However, due to the addition of the coded-aperture matrix, the conjugate-gradient algorithm is used instead. Therefore, the computational cost of the conjugate-gradient algorithm $\mathcal{O}_{CG}$ is added to the image update step presented in Chapter 3, which is shown in Table 3.5.

5.5 Numerical Results

Here, reconstruction results are presented for a realistic scenario in visible bands. We consider a dataset of size $256 \times 256 \times 16$ (16 wavelengths from $510 - 660$ nm with 10 nm spacing) in the visible band that was obtained from an online hyperspectral image database referred as Objects [69]. The compressive diffractive imaging system with a fixed detector is shown in Fig. 5.2 is used in the simulations.

A photon sieve is used as a diffractive lens with a fixed measurement plane. For the photon sieve, we use two different designs. These designs are named Design I and Design II for the rest of the paper. For Design I, the smallest hole diameter of $\delta = 8$ μm. The pixel size of the detector is chosen as 4 μm to satisfy the spectral resolution. Photon sieves are used with the detector to diffractive lens distance $f_0 = 2.56$ cm, and the design is changed to focus a different wavelength at this distance. For this purpose, the outer diameter of the sieve is changed as $D_k = \lambda_k f_0/\delta$. For example, if the $\lambda_k = 580$ nm, the diameter $D_k = 1.856$ mm. The expected spectral resolution is $4\delta^2/f_0 = 10$ nm, which matches with the sampling interval. This design is previously exploited in [13]. In order to improve Design I, the second design is considered. For this Design II, the smallest hole diameter of $\delta = 16$ μm. The pixel size of the detector is chosen as 8 μm to satisfy the spectral resolution. Photon sieves are used with the detector to diffractive lens distance $f_0 = 2.56$ cm, and the design is changed to focus a different wavelength at this distance. For this purpose, the outer diameter of the sieve is changed as $D_k = \lambda_k f_0/\delta$. For example, if the $\lambda_k = 580$ nm, the diameter $D_k = 0.92$ mm. The expected spectral resolution is greater than the minimum spectral resolution as $4\delta^2/f_0 = 40$ nm.

The main difference between Design I and Design II, the point spread functions $h_{k,s}[n_1, n_2]$ in Design II cause less blurry measurements than Design I. These PSFs $h_{k,s}[n_1, n_2]$ in the range of $510 - 660$ nm with 10 nm intervals, are shown in Figure B.16a, B.16b, B.17a and B.17b for 3 and 4 measurements. As shown in Fig. B.16b and B.17b, the point spread functions in Design II provide less blurry measurements than Design I, which are shown in Fig. B.16a and B.17a. Namely, the point spread functions of Design II provide more spectral information from the measured images than Design I for both 3 and 4 measurements. For example, the PSF for the measurement focused on 560 nm, the adjacent PSFs of 550 and 570 nm creates more blurry measurements in Design I than Design II, as shown in Fig. 5.3. However, the spatial resolution of Design II is two times worse than Design I. As a result, there is a trade-off between spatial and spectral resolutions of Design I and Design II.

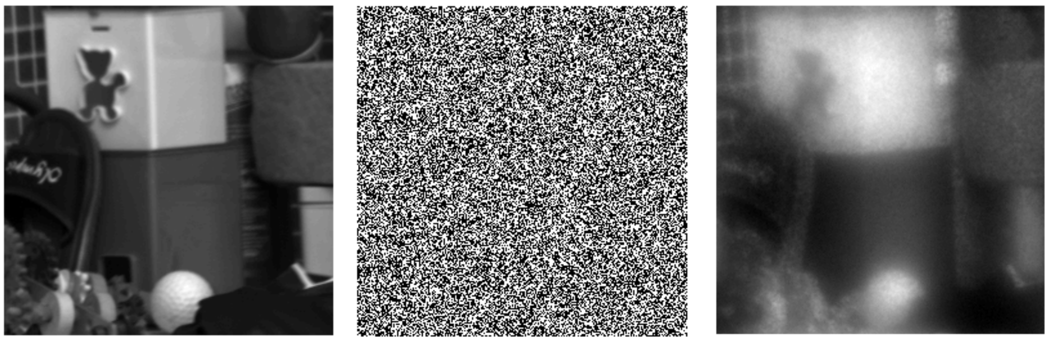


(a)

Figure 5.3: The PSFs at $550, 560, 570$ nm for the measurement focused at 560 nm. Top line: For Design I, Bottom line: For Design II

The compressive measurements are simulated using the forward problem in Eq (5.2) with additive white Gaussian noise. In each measurement, the system applies the same masking operation to each spectral band using a traditional block-unblock mask. The entries of the mask are drawn from a Bernoulli distribution, which is shown in Fig. 5.4b. After the coded field passes through the same plane by changing the outer diameter of the sieve. Superimposed true image along the spectral dimension, a sample mask, and a blurry compressive measurement are displayed in Figure 5.4a, 5.4b, and 5.4c. In this measurement, 560 nm is focused by the sieve onto the detector plane, while all other spectral components are defocused.

We consider different compressive scenarios with 3 and 4 measurements taken at



(a) (b) (c)

Figure 5.4: (a) Superimposed true image, (b) Coding mask pattern, (c) A sample compressive measurements

different planes with equidistant wavelengths from the spectral range 510-660 nm. More specifically, the chosen wavelengths are $\{510, 560, 610, 660\}$ nm for the $K = 4$ and $\{530, 580, 630\}$ nm for $K = 3$. These cases with $K = 4$ and $K = 3$ correspond the compression levels of $100 \times (1 - K/S)$, of 81.25% and 75%. These are equivalent to reconstructing spectral cube from 18.75% and 25% data.

To analyze medium to low noise cases, different SNR levels are considered. The measurements are generated using the forward problem in Eq. (2.1) with signal-to-noise ratios (SNRs) of 20, 30 and 40 dBs. Our goal is to observe the effect of these designs and the number of measurements with different priors. Similar to non-compressive scenario, we first exploit a 3D transform for the image reconstructions: a Kronecker basis $\mathbf{T} = \mathbf{T_1} \bigotimes \mathbf{T_2}$ where $\mathbf{T_1}$ is the 2D Symmlet-8 basis and $\mathbf{T_2}$ is the 1D cosine basis. Parameter selection is shown in Table 5.1 where $\beta = 1$. Second, the convolutional dictionary with Tikhonov regularization has been exploited instead of the regular one, since we have already shown in Chapter 4 that Tikhonov regularization improves the performance of the convolutional dictionary. Here, the convolutional dictionary in the image reconstruction algorithm is initialized with pre-trained dictionaries, which is shown in Fig. 4.8b. We use the same pre-trained dictionary of size $32 \times 32 \times 5$ with filter size $M = 6$ for both compressive and non-compressive scenario. The parameter selection of the algorithm is shown in Table 5.2. Note that the patch-based dictionaries are not exploited for the 3D case due to their substantial computational complexity. Here, the algorithm with the sparsifying transform takes around 1100 seconds, and the algorithm with a convolutional dictionary takes around 2400 seconds on the average for various SNR levels.

Table 5.1: Parameter Selection for various SNRs with Sparsifying Transform

Updates	20 dB	30 dB	40 dB
λ	0.03	0.01	0.005
ρ	0.5	1	2

Table 5.2: Parameter Selection for various SNRs with Convolutional Dictionary

Updates	20 dB	30 dB	40 dB
λ	0.005	0.005	0.005
β	$[1, 2]$	$[2, 5]$	$[5, 10]$
ρ	50	50	50
σ	1000	1000	1000
μ	$[10, 20]$	$[10, 20]$	$[10, 20]$

Table 5.3 and 5.4 show the average reconstruction PSNR, SAM and SSIM values under 20 dB, 30 dB and 40 dB SNRs for 3 and 4 measurements. Each table includes the comparison of 2D Symmlet $\bigotimes$ 1D DCT transform and convolutional dictionary with Tikhonov regularization for Design I and Design II. These values are obtained as taking an average of 10 Monte-Carlo results for Objects data.

As shown in tables, for Design I, 2D Symmlet $\bigotimes$ 1D DCT have a better performance than the convolutional dictionary in terms of PSNR, SAM, and SSIM metrics. Specifically, for $K = 3$, there is a substantial difference between performances. Since the convolutional dictionary is adaptively learned from the measurements, it is affected by the decreasing number of the measurements. On the other hand, for Design II, there is an enhancement in the performance of the convolutional dictionary. Nonetheless, 2D Symmlet $\bigotimes$ 1D DCT has similar performance with the convolutional dictionary for $K = 3, 4$ and different SNRs. Since Design II provides more knowledge about the spectral dimension, the performance of the convolutional dictionary increased as well. By optimizing the size of the dictionary and choosing better point spread functions, the performance of the convolutional dictionary can be further improved.

Table 5.3: Image reconstruction performance comparison in terms of PSNR, SAM and SSIM for 2D Symmlet $\bigotimes$ 1D DCT transform and convolutional dictionary (Tikhonov) for Objects data with 4 measurements

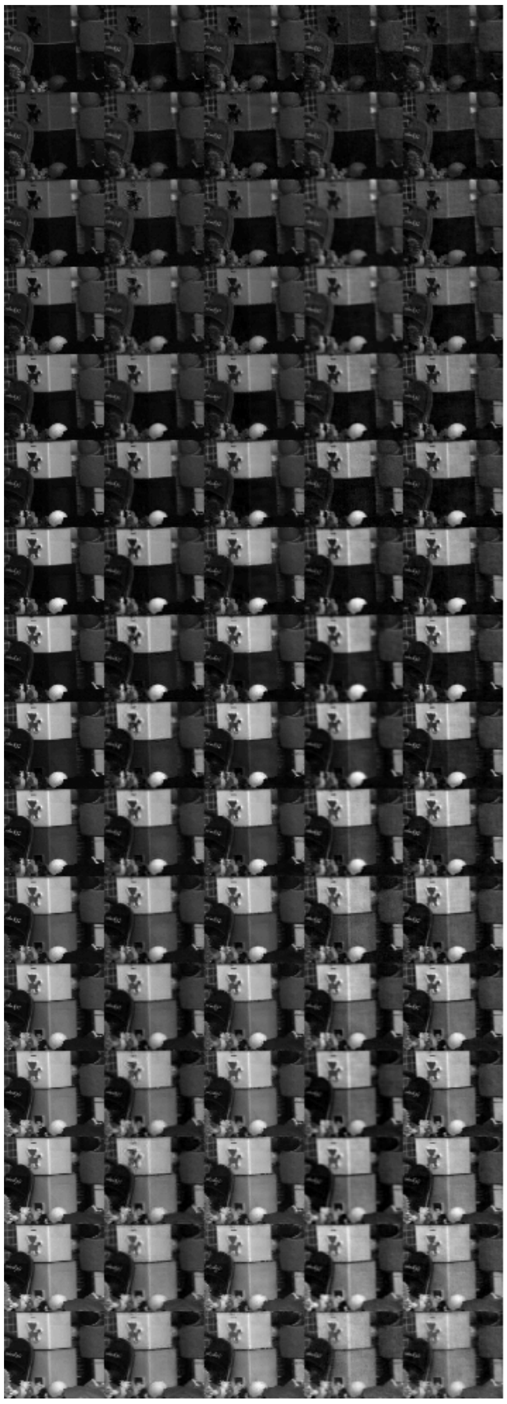
Dataset	SNR	Design I		Design II	
	(dB)	Transform	ConvDic	Transform	ConvDic
	20	$26.88/9.42°/0.78$	$26.54/10.08°/0.79$	$26.92/8.96°/0.82$	$26.96/9.92°/0.79$
Objects	30	$28.98/6.80°/0.84$	$28.24/8.54°/0.84$	$29.95/6.15°/0.88$	$30.85/6.30°/0.89$
	40	$29.48/6.75°/0.87$	$28.78/8.18°/0.88$	$32.82/5.04°/0.93$	$33.22/4.75°/0.93$

Table 5.4: Image reconstruction performance comparison in terms of PSNR, SAM and SSIM for 2D Symmlet $\bigotimes$ 1D DCT transform and convolutional dictionary (Tikhonov) for Objects data with three measurements

Dataset	SNR	Design I		Design II	
	(dB)	Transform	ConvDic	Transform	ConvDic
	20	$26.25/9.73°/0.76$	$24.11/13.95°/0.73$	$26.31/9.28°/0.82$	$26.20/11.01°/0.79$
Objects	30	$28.18/7.79°/0.84$	$25.69/11.90°/0.81$	$28.33/7.00°/0.87$	$29.13/7.38°/0.89$
	40	$28.80/7.56°/0.86$	$26.82/10.38°/0.85$	$32.10/5.39°/0.93$	$31.80/5.47°/0.92$

The reconstructed images using 2D Symmlet $\bigotimes$ 1D DCT transform and convolutional dictionary for Design I and Design II are shown in Fig. 5.5a for the sixteen sources, together with the original scenes for comparison, when the input SNR is 30 dB and 4 measurements are available. As visualized in the second and the third column of Fig. 5.5a, 2D Symmlet $\bigotimes$ 1D DCT transform provides smoother structure than the convolutional dictionary. Furthermore, we also observe that Design II (in Column II and Column IV) provides remarkably better visual quality than Design I (in Column I and Column III) for both 2D Symmlet $\bigotimes$ 1D DCT and convolutional dictionary.

Moreover, the difference between the original image and the reconstructed images are shown in Fig. 5.5b. Convolutional dictionary with Design I (Column III) clearly fails as it could not recover specific spectral bands ($530-540$, $580-590$, $630-640$ nm). As shown in Fig. B.16a, the point spread functions do not provide sufficient contribution from the measurements for these bands. Moreover, we also clearly observe that 2D Symmlet $\bigotimes$ 1D DCT transform (Column I and Column II) causes smooth loss from the recovered image. On the other hand, we still observe small artifacts on the images reconstructed with the convolutional dictionary (Column III and Column IV).


(a)

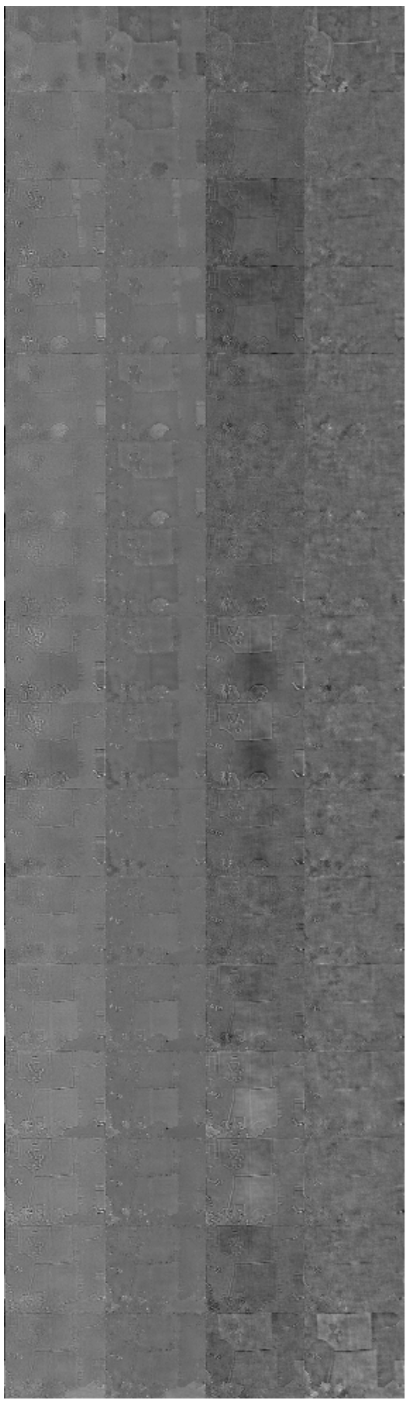
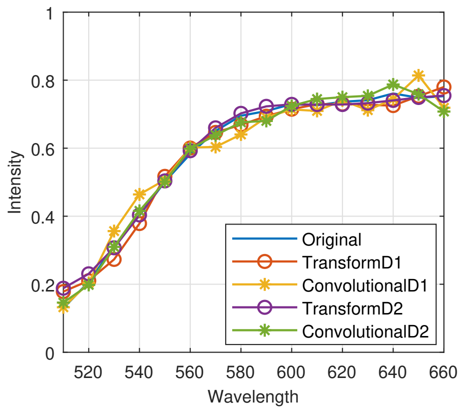
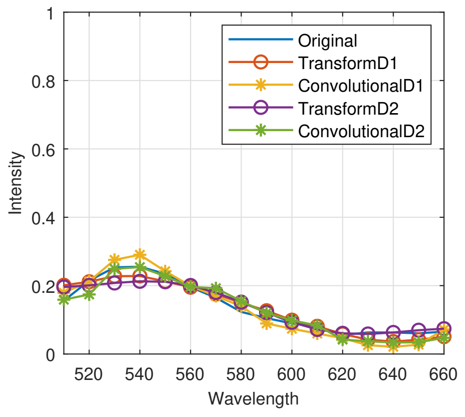

(b)

Figure 5.5: (a) Left-to-Right: Original images, Reconstructed spectral images with 2D Symmlet $\bigotimes$ 1D DCT transform in Design I and Design II, with convolutional dictionary (Tikhonov) in Design I and Design II under 30 dB SNR (b) The difference between original image and reconstructed images

Furthermore, to demonstrate successful recovery along the spectral dimension, we also display the spectrum of the point at $\mathbf{P1}$ and $\mathbf{P2}$ in Figure 5.6a and 5.6b for Objects Data respectively. The reconstructed spectra at these points are plotted with the spectra of the original image. It can be seen that the convolutional dictionary in Design I recovers the spectrum poorly, and other counterparts have better performances than Design I. Consequently, using Design II substantially enhances the spectral reconstruction performance of the convolutional dictionary; however, it does not provide a huge performance gain for the sparsifying transform.



(a)



(b)

Figure 5.6: (a) Spectrum for Point P1, (b) Spectrum for Point P2 with 30 dB SNR in Objects

5.6 Conclusion

In this section, we have introduced convolutional inverse problems with a compressive setting and develop image reconstruction algorithms using a sparsifying transform and convolutional dictionary. We demonstrate the numerical performance of the developed methods in a compressive spectral imaging application with diffractive lenses. Considering a different number of measurements and different designs, we evaluate the image reconstruction performance of the sparsity priors. We observe that, when the system allows more contributions from the data, the convolutional dictionary generally outperforms the sparsifying transform. Since we exploit a data-adaptive model for the convolutional dictionary, the result is expected. Furthermore, with a better pre-trained dictionary and size optimization of the dictionary for a particular imaging system, image reconstruction performance of the convolutional dictionary can be further improved.

CHAPTER 6

CONCLUSION

In this thesis, we have studied convolutional inverse problems. We first provided a general formulation for these problems, and then develop fast and efficient image reconstruction algorithms that exploit sparse models in analysis and synthesis forms. These priors involve sparsifying transforms or data-adaptive dictionaries that are patch-based and convolutional. The developed methods are based on the alternating direction method of multipliers (ADMM), which is a state-of-the-art problem-solving technique for signal and image recovery problems. The numerical performance of the developed algorithms is evaluated for a three-dimensional image reconstruction problem in spectral imaging. The results demonstrate the superiority of the convolutional dictionary prior to others. The developed algorithms are also extended to the compressive setting with compressed convolutional measurements.

In Chapter 2, we introduced the convolutional forward problem by considering various imaging problems. These include image deconvolution, multi-frame image deconvolution, and examples from spectral and depth imaging. The convolutional inverse problem was also formulated with various priors involving sparsifying transform, patch-based dictionary, convolutional dictionary, and convolutional dictionary with Tikhonov regularization. This prior enforces the Gradient minimization of the sparse codes to eliminate the artifacts caused by the convolutional dictionary. Lastly, the merits and drawbacks of these priors have been discussed based on the literature.

In this chapter, we also mentioned that the reconstructed data cube might be correlated or uncorrelated in the third dimension. Therefore, the inverse problem is modeled with both two and three-dimensional priors. If the data is correlated along the third dimension, three-dimensional priors have been exploited. Otherwise, two-dimensional priors are sufficient to represent the data cube.

In Chapter 3, we developed image reconstruction methods for the presented priors. For the first algorithm, fixed transforms are exploited for convolutional inverse problems. Second, we developed an image reconstruction algorithm exploiting patch-based sparse representations. An extension with online dictionary learning was also included to improve image reconstruction quality. Third, we developed an image reconstruction method with the convolutional dictionary. This algorithm also involves online dictionary learning. Lastly, we presented an extension of the convolutional sparse prior with Tikhonov regularization. These methods are presented for three-dimensional image reconstruction problems for various cases with or without correlation along the third-dimension. Moreover, all methods are based on the alternating direction method of multipliers (ADMM).

In this chapter, we have presented the efficient implementation of the proposed methods in detail, which does not require matrix-vector multiplications. We have exploited the property that convolution matrices are diagonalizable by the frequency matrices, and element-wise multiplications are performed for efficient computation. Lastly, we analyzed the computational complexity of developed methods. As fixed transforms provide fast implementations, this method has lower computational complexity than the other counterparts. Furthermore, the convolutional dictionary provides more cost-efficient solutions than the patch-based dictionary.

In Chapter 4, the numerical performance of the developed algorithms has been evaluated for a three-dimensional image reconstruction problem in spectral imaging for discrete and continuous spectrum. For the discrete spectrum case, the data cube is uncorrelated along the third dimension. Hence two-dimensional priors are exploited. On the other hand, the reconstructed data cube is correlated along all three dimensions for the continuous spectrum case; therefore, three-dimensional priors are enforced. For the discrete spectrum case, we analyzed the image reconstruction quality of the improved methods in terms of PSNR and SSIM quality metrics. Here, both patch-based and convolutional dictionary with online dictionary learning has an advantage over the sparsifying transform in terms of image reconstruction quality. On the other hand, both the sparsifying transform and convolutional dictionary provides substantially faster solutions than the patch-based dictionary. Namely, the convolutional dictionary combines higher quality solutions with fast implementations. Furthermore, the performance of the convolutional dictionary has been improved with the Tikhonov regularization of the sparse codes. Consequently, the best prior has been selected as a convolutional dictionary with online learning and Tikhonov regularization.

For continuous spectrum case, the numerical performance of the developed algorithms is evaluated for a three-dimensional denoising and PSSI problem in spectral imaging. Due to the enormous implementation cost of the patch-based method, we only assessed the performance of the fixed transform and convolutional dictionary with online dictionary learning. In addition to PSNR and SSIM, we also used another quality metric SAM, which evaluates the reconstruction quality in the spectral dimension. We observed that the convolutional dictionary results in higher image reconstruction performance than the sparsifying transform in terms of PSNR and SAM; however, it causes some artifacts on the images as can be understood from the visual inspection. On the other hand, by adding Tikhonov regularization of the sparse codes, artifacts on the images have been eliminated for the convolutional dictionary. Therefore, PSNR, SAM, and SSIM levels have been enhanced for almost all images and all SNR levels. However, the algorithm with the convolutional dictionary has a $2\times$ slower convergence rate than the one with the sparsifying transform.

In Chapter 5, the developed algorithms are also extended to the compressive setting with compressed convolutional measurements. The three-dimensional data cube is reconstructed from highly compressed coded measurements using analysis and synthesis priors. These algorithms are also based on ADMM, and the only difference with the non-compressive case occurs in the computation of the image update. Furthermore, we illustrated the performance of developed methods using a sparsifying transform and convolutional dictionary for two different photon sieve designs for different number of measurements. We observe that the performance of the priors varies with respect to the design of the imaging system. When the imaging system causes less blurry measurements, the convolutional dictionary has promising results, and its performance can be further improved. On the other hand, when the imaging system results in highly blurry measurements, the sparsifying transform is more successful than the convolutional dictionary.

As future work, in all of the above discussion, instead of optimization-based learning methods, deep learning methods can improve the image reconstruction performance in multidimensional imaging problems. Another critical part is the optimization of the design of the imaging system, which will enable better system matrices satisfying CS requirements. By combining a well-improved system with convenient priors, image reconstruction performance can be enhanced remarkably.

Appendix A

MATHEMATICAL DERIVATIONS

A.1 Diagonal Block Linear Systems

Given sets of vectors $\mathbf{a}_m \in \mathbb{C}^{N^2S}$ $\mathbf{b}_m \in \mathbb{C}^{N^2S}$ and $\mathbf{c}_m \in \mathbb{C}^{N^2S}$ and unknown vectors $\mathbf{z}_m \in \mathbb{C}^{N^2S}$ Assuming $\mathbf{A}_m$ = $diag(\mathbf{a}_m)$, $\mathbf{B}_m$ = $diag(\mathbf{b}_m)$ and $\mathbf{A}$ = $[\mathbf{A}_1 ... \mathbf{A}_M]$ and

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{B}_M \end{pmatrix}, \mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ : \\ \mathbf{z}_M \end{bmatrix}, \text{and } \mathbf{c} = \begin{bmatrix} \mathbf{c}_1 \\ : \\ \mathbf{c}_M \end{bmatrix}.$$

The size of $\mathbf{A}_m$ is $N^2 \times N^2$ and since $\mathbf{A}$ contains M blocks its size is $N^2 \times MN^2$. In the following discussion, $\mathbf{a}*$ which is conjugate of $\mathbf{a}$. $\mathbf{A}^H$ denotes the Hermitian transpose (conjugate transpose) of $\mathbf{A}$. In order to solve the linear system $(\mathbf{A}^H\mathbf{A}+\mathbf{B})\mathbf{z}$ = $\mathbf{c}$. The expanded version is

$$\begin{pmatrix} \mathbf{A}_1^H\mathbf{A}_1 + \mathbf{B}_1 & \cdots & \mathbf{A}_1^H\mathbf{A}_M \\ \vdots & \ddots & \vdots \\ \mathbf{A}_M^H\mathbf{A}_1 & \cdots & \mathbf{A}_M^H\mathbf{A}_M + \mathbf{B}_M \end{pmatrix} \begin{bmatrix} \mathbf{z}_1 \\ : \\ \mathbf{z}_M \end{bmatrix} = \begin{bmatrix} \mathbf{c}_1 \\ : \\ \mathbf{c}_M \end{bmatrix}. \tag{A.1}$$

Here, $\mathbf{A}_m$'s are diagonal, hence the multiplications in the blocks above are also diagonal. Which means $\mathbf{A}_i^H\mathbf{A}_j$'s are also diagonal for $i, j = 1, 2, ..., M$. By closely looking at the multiplications considering elements, each element of $\mathbf{c}_m[n]$ where n represents (n_1, n_2, s) pixels is computed as follows:

$$\sum_{i=1}^{M}(\mathbf{a}_m[n]^*\mathbf{a}_i[n]\mathbf{z}_i[n]) + \mathbf{b}_m[n]\mathbf{z}_m[n] = \mathbf{c}_m[n] \tag{A.2}$$

Then, we rearrange the places of the indices as follows:

$$\begin{aligned}\tilde{\mathbf{a}}_n[m] &= \mathbf{a}_m[n]*,\\ \tilde{\mathbf{b}}_n[m] &= \mathbf{b}_m[n],\\ \tilde{\mathbf{c}}_n[m] &= \mathbf{c}_m[n],\\ \tilde{\mathbf{z}}_n[m] &= \mathbf{z}_m[n]\end{aligned} \tag{A.3}$$

by inserting these equations to Eq. (A.2) we obtain

$$\sum_{i=1}^{M}(\tilde{\mathbf{a}}_n[m]\tilde{\mathbf{a}}_n[i]^*\tilde{\mathbf{z}}_n[i]) + \tilde{\mathbf{b}}_n[m]\tilde{\mathbf{z}}_n[m] = \tilde{\mathbf{c}}_n[m]. \tag{A.4}$$

Here, indices of the elements are changed. This results in the following system

$$(\tilde{\mathbf{a}}_n\tilde{\mathbf{a}}_n^H + diag(\tilde{\mathbf{b}}_n))\tilde{\mathbf{z}}_n = \tilde{\mathbf{c}}_n. \tag{A.5}$$

where each $\tilde{\mathbf{a}}_n$, $\tilde{\mathbf{b}}_n$, $\tilde{\mathbf{c}}_n$, $\tilde{\mathbf{x}}_n$is of size M. After these replacements of the indices Eq. (A.1) is expressed as

$$\begin{pmatrix} \tilde{\mathbf{a}}_{1,1,1}\tilde{\mathbf{a}}_{1,1,1}^H + diag(\tilde{\mathbf{b}}_{1,1,1}) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \tilde{\mathbf{a}}_{N,N,S}\tilde{\mathbf{a}}_{N,N,S}^H + diag(\tilde{\mathbf{b}}_{N,N,S}) \end{pmatrix} \begin{bmatrix} \tilde{\mathbf{z}}_{1,1,1} \\ : \\ \tilde{\mathbf{z}}_{N,N,S} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{c}}_{1,1,1} \\ : \\ \tilde{\mathbf{c}}_{1,1,1} \end{bmatrix}. \tag{A.6}$$

Here, linear systems Eq. (A.1) and (A.6) are inherently same. Only difference is changing indices of the elements. Therefore, size of $MN^2S \times MN^2S$ system in Eq. (A.1) is replaced by N^2S systems each is of size $M \times M$. Each one consists of a rank one plus a diagonal component which can be solved by Sherman-Morrison formulation.

A.2 Sherman-Morrison Solution

A linear system of the form

$$(\mathbf{q}\mathbf{q}^H + \mathbf{B})\mathbf{z} = \mathbf{c} \tag{A.7}$$

can be solved using Sherman-Morrison formula. Taking inverse with Sherman-Morrison formula is as follows:

$$(\mathbf{u}\mathbf{v}^H + \mathbf{B})^{-1} = \mathbf{B}^{-1} - \frac{\mathbf{B}^{-1}\mathbf{u}\mathbf{v}^H\mathbf{B}^{-1}}{1 + \mathbf{u}^H\mathbf{B}^{-1}\mathbf{v}} \tag{A.8}$$

Hence, using Sherman-Morrison formula $\mathbf{z}$ can be obtained from Eq. (A.7) as follows:

$$\mathbf{z} = (\mathbf{B}^{-1} - \frac{\mathbf{B}^{-1}\mathbf{a}\mathbf{a}^H\mathbf{B}^{-1}}{1+\mathbf{a}^H\mathbf{B}^{-1}\mathbf{a}})\mathbf{c} \tag{A.9}$$

Here, $\mathbf{a}^h\mathbf{B}^{-1}\mathbf{c}$ is scalar and $\mathbf{a}(\mathbf{a}^h\mathbf{B}^{-1}\mathbf{c})$ = $\mathbf{a}^h\mathbf{B}^{-1}\mathbf{c}\mathbf{a}$. Hence, Eq. (A.9) is modified as

$$\mathbf{z} = \mathbf{B}^{-1}(\mathbf{c} - \frac{\mathbf{a}^H\mathbf{B}^{-1}\mathbf{c}\mathbf{a}}{1+\mathbf{a}^H\mathbf{B}^{-1}\mathbf{a}}) \tag{A.10}$$

In our case $\mathbf{B}$ = ρ and Eq. (A.10) becomes

$$\mathbf{z} = \rho^{-1}(\mathbf{c} - \frac{\mathbf{a}^H\mathbf{c}\mathbf{a}}{\rho+\mathbf{a}^H\mathbf{a}}) \tag{A.11}$$

A.3 Multiple Diagonal Block Linear Systems

Now, assume that we define vectors $\mathbf{a}_{p,m} \in \mathbb{C}$ and $\mathbf{A}_{p,m}$ = $diag(\mathbf{a}_{p,m})$ and $\mathbf{A}_p$ = $[\mathbf{A}_{p,1}...\mathbf{A}_{p,M}]$. If the linear system being solved is in the form of $(\sum_{p=1}^{P}\mathbf{A}_p^H\mathbf{A}_p+\mathbf{B})\mathbf{z}$ = $\mathbf{c}$ which is also expressed as

$$\begin{pmatrix} \sum_{p=1}^{P}\mathbf{A}_{p,1}^H\mathbf{A}_{p,1}+\mathbf{B}_1 & \cdots & \sum_{p=1}^{P}\mathbf{A}_{p,1}^H\mathbf{A}_{p,M} \\ \vdots & \ddots & \vdots \\ \sum_{p=1}^{P}\mathbf{A}_{p,M}^H\mathbf{A}_{p,1} & \cdots & \sum_{p=1}^{P}\mathbf{A}_{p,M}^H\mathbf{A}_{p,M}+\mathbf{B}_M \end{pmatrix} \begin{bmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_M \end{bmatrix} = \begin{bmatrix} \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_M \end{bmatrix}. \tag{A.12}$$

Now the replacement of the indices is as follows:

$$\begin{aligned} \tilde{\mathbf{a}}_{p,n}[m] &= \mathbf{a}_{p,m}[n]^*, \\ \tilde{\mathbf{b}}_{p,n}[m] &= \mathbf{b}_{p,m}[n], \\ \tilde{\mathbf{c}}_{p,n}[m] &= \mathbf{c}_{p,m}[n], \\ \tilde{\mathbf{z}}_{p,n}[m] &= \mathbf{z}_{p,m}[n] \end{aligned} \tag{A.13}$$

Now, using these equations we acquire the following linear system:

$$\sum_{i=1}^{M}(\sum_{p=1}^{P}\tilde{\mathbf{a}}_{p,n}[m]\tilde{\mathbf{a}}_{p,n}[i]^*\tilde{\mathbf{z}}_{p,n}[i]) + \tilde{\mathbf{b}}_n[m]\tilde{\mathbf{z}}_{p,n}[m] = \tilde{\mathbf{c}}_n[m]. \tag{A.14}$$

the following form is acquired with vector products

$$(\sum_{p=1}^{P}\tilde{\mathbf{a}}_{p,n}\tilde{\mathbf{a}}_{p,n}^* + diag(\tilde{\mathbf{b}}_n))\tilde{\mathbf{x}}_{p,n} = \tilde{\mathbf{c}}_n. \tag{A.15}$$

Now, the left hand-side term is rank-N and efficiently solved by iterated Sherman-Morrison formula.

Algorithm 4 Iterated Sherman-Morrison

```
Require: $\mathbf{a}_p$: Measured image, $\rho$: Parameter
 1: $\alpha$ = $\rho^{-1}\mathbf{a}_1$ and $\beta$ = $\rho^{-1}\mathbf{b}$
 2: for do$p \in \{1, 2, ..., P\}$
 3:     $\gamma_{p-1} = \frac{\alpha}{1+\mathbf{a}_{p-1}^H\alpha}$
 4:     $\beta = \beta - \gamma_{p-1}\mathbf{a}_{p-1}^H\beta$
 5:     if then$k \leq K-1$ $\alpha$ = $\rho^{-1}\mathbf{a}_p$
 6:         for do$l \in \{1, 2, ..., p\}$
 7:             $\alpha = \alpha - \gamma_{l-1}\mathbf{a}_{l-1}^H\alpha$
 8:         end for
 9:     end if
10: end for
```

A.4 Iterated Sherman-Morrison Solution

A linear system of the form

$$(\sum_{p=1}^{P}\mathbf{q}_p\mathbf{q}_p^H + \mathbf{B})\mathbf{z} = \mathbf{c} \tag{A.16}$$

By defining $\mathbf{A}_1$ = $\mathbf{B}$, $\mathbf{A}_{p+1}$ = $\mathbf{A}_p + \mathbf{a}_p\mathbf{a}_p^H$. With the Sherman-Morrison formula

$$\mathbf{A}_{p+1}^{-1} = \mathbf{A}_p^{-1} - \frac{\mathbf{A}_p^{-1}\mathbf{a}_p\mathbf{a}_p^H\mathbf{A}_p^{-1}}{1+\mathbf{a}_p^H\mathbf{A}_p^{-1}\mathbf{a}_p} \tag{A.17}$$

By defining $\alpha_{l,p}$ = $\mathbf{A}_l^{-1}\mathbf{a}_p$ and β_p = $\mathbf{A}_p^{-1}\mathbf{c}$. Next, $\alpha_{1,p}$ = $\mathbf{B}^{-1}\mathbf{a}_p$ and β_0 = $\mathbf{B}^{-1}\mathbf{c}$

$$\beta_{p+1} = \mathbf{A}_{p+1}^{-1}\mathbf{c} \tag{A.18}$$

$$\beta_{p+1}^{-1} = \mathbf{A}_p^{-1}\mathbf{c} - \frac{\mathbf{A}_p^{-1}\mathbf{a}_p\mathbf{a}_p^H\mathbf{A}_p^{-1}\mathbf{c}}{1+\mathbf{a}_p^H\mathbf{A}_p^{-1}\mathbf{a}_p} \tag{A.19}$$

$$\beta_{p+1} = \beta_p - \frac{\alpha_{p,p}\mathbf{a}_p^H\beta_p}{1+\mathbf{a}_p^H\mathbf{A}_p^{-1}\mathbf{a}_p} \tag{A.20}$$

and

$$\alpha_{l+1,p} = \mathbf{A}_{l+1}^{-1}\mathbf{a}_p \tag{A.21}$$

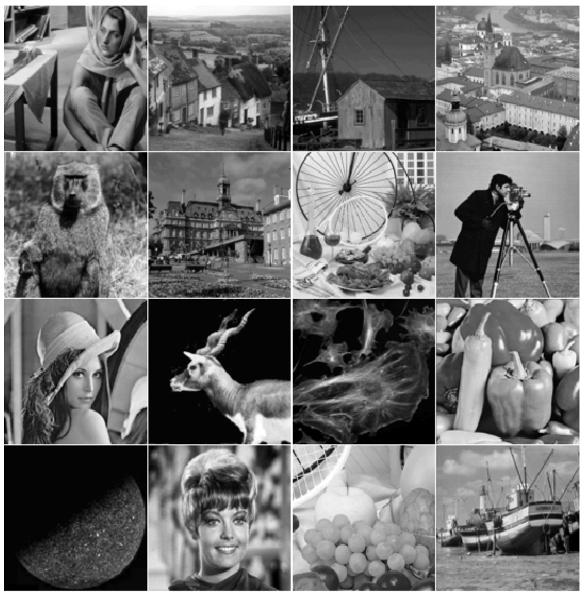
$$\alpha_{l+1,p} = \mathbf{A}_l^{-1}\mathbf{a}_p - \frac{\mathbf{A}_l^{-1}\mathbf{a}_l\mathbf{a}_l^H\mathbf{A}_l^{-1}\mathbf{a}_p}{1+\mathbf{a}_l^H\mathbf{A}_l^{-1}\mathbf{a}_l} \tag{A.22}$$

$$\alpha_{l+1,p} = \alpha_{l,p} - \frac{\alpha_{l,l}\mathbf{a}_l^H \alpha_{l,p}}{1 + \mathbf{a}_l^H \alpha_{l,l}} \tag{A.23}$$

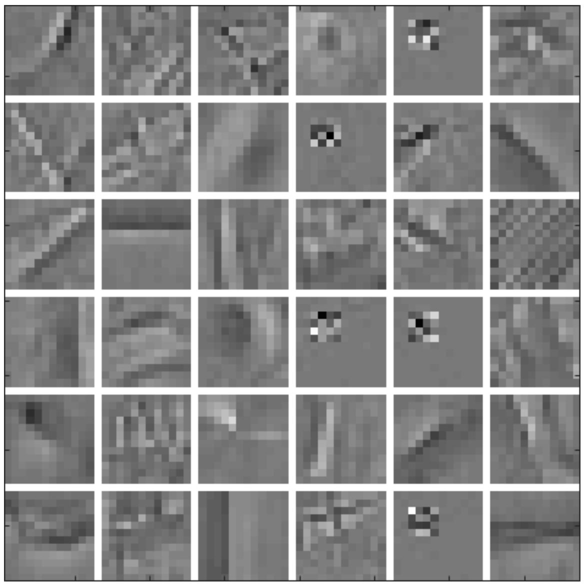
An iterative algorithm computing the solution of the linear system given in Eq. (A.15) is easily derived from the equations from Eq. (A.16) to (A.23).

Appendix B

RESULTS

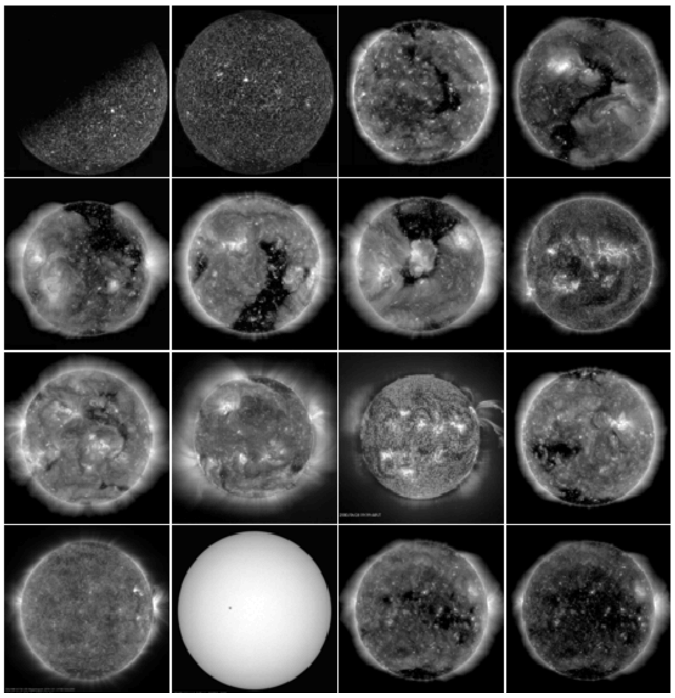


(a)



(b)

Figure B.1: (a) MATLAB Standard Images, (b) 36 Dictionary Filters Trained with Matlab Standard Images


(a)

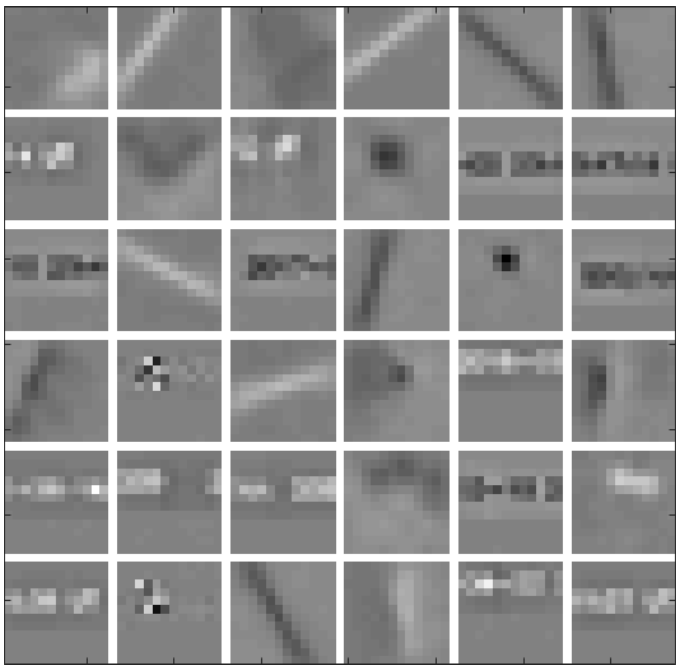
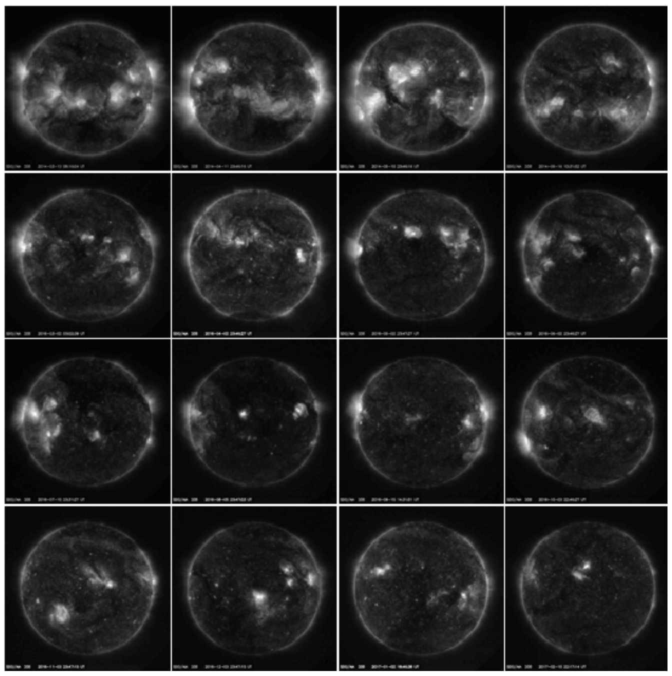
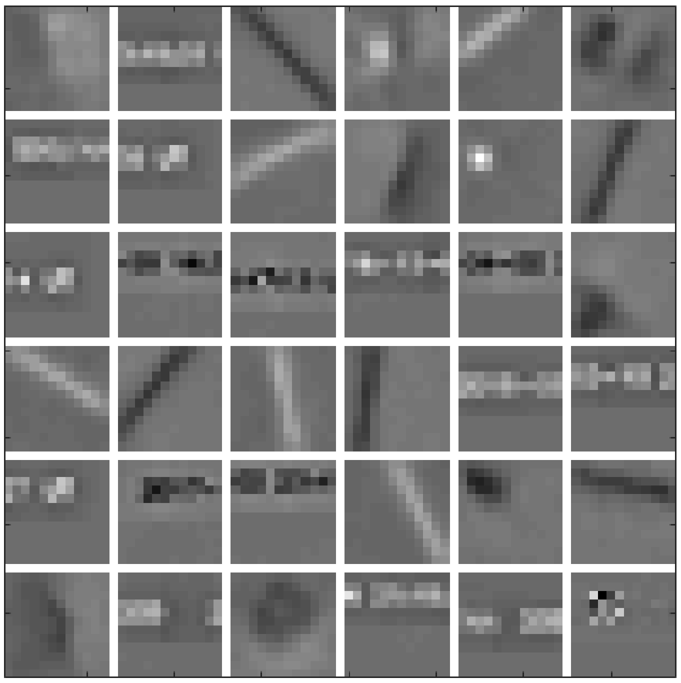

(b)

Figure B.2: (a) Random solar images, (b) 36 dictionary filters trained with random solar images



(a)



(b)

Figure B.3: (a) Solar images from telescope AIA335, (b) 36 dictionary filters trained with solar images from telescope AIA335

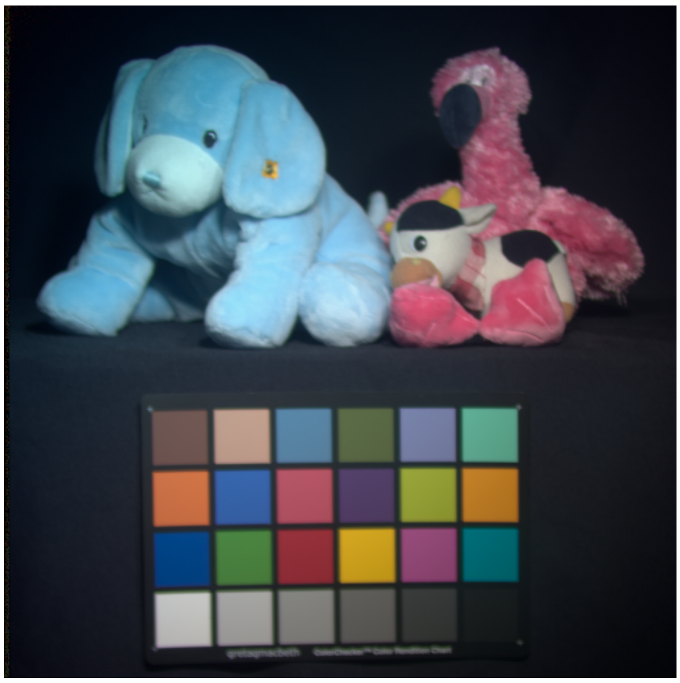


Figure B.4: Training images are cropped from this image

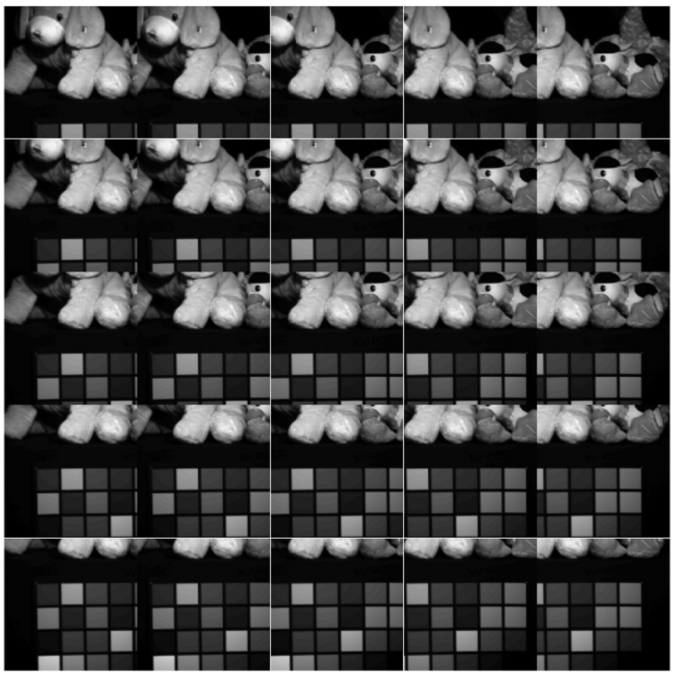
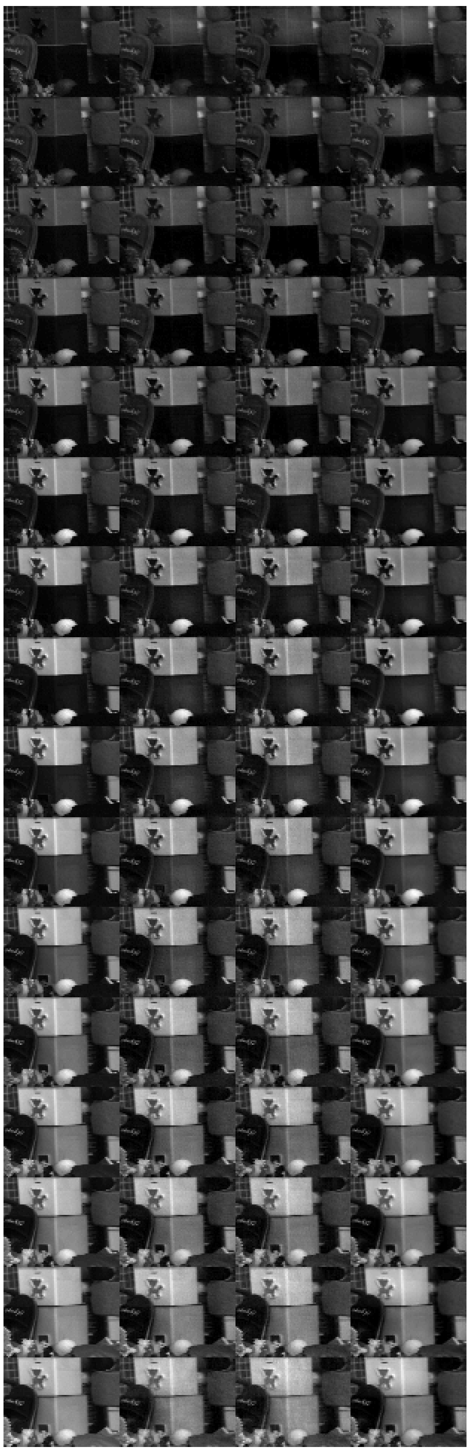
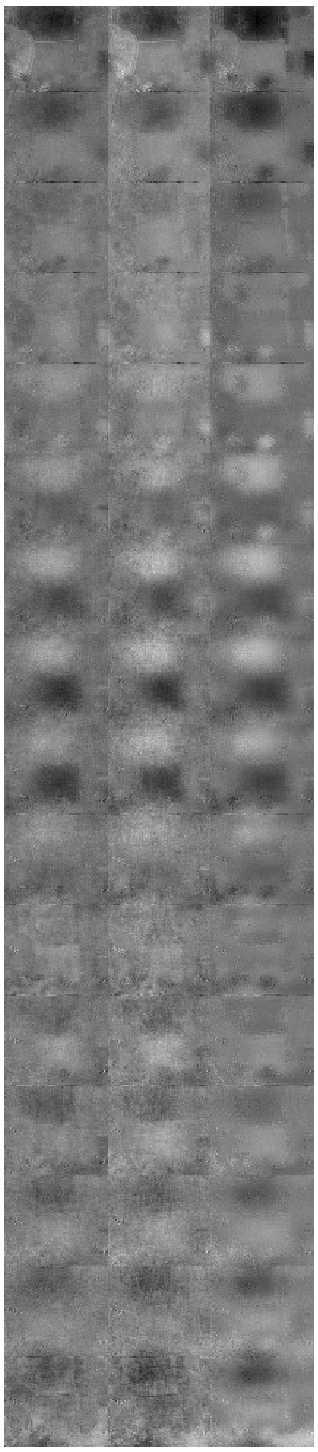


Figure B.5: Cropped training images for a single band

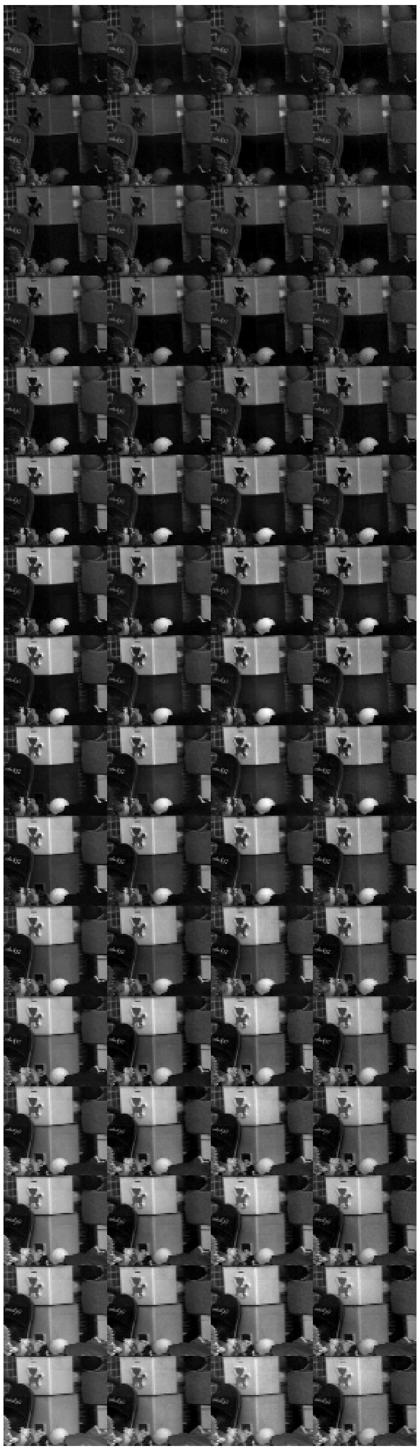


(a)



(b)

Figure B.6: (a) Left-to-right: Original images (Objects), Reconstructed spectral images with 2D Symmlet + 1D DCT (Kronecker) transform, convolutional dictionary, convolutional dictionary with Tikhonov regularization under 30 dB SNR (b) The difference between original image and reconstructed images


(a)

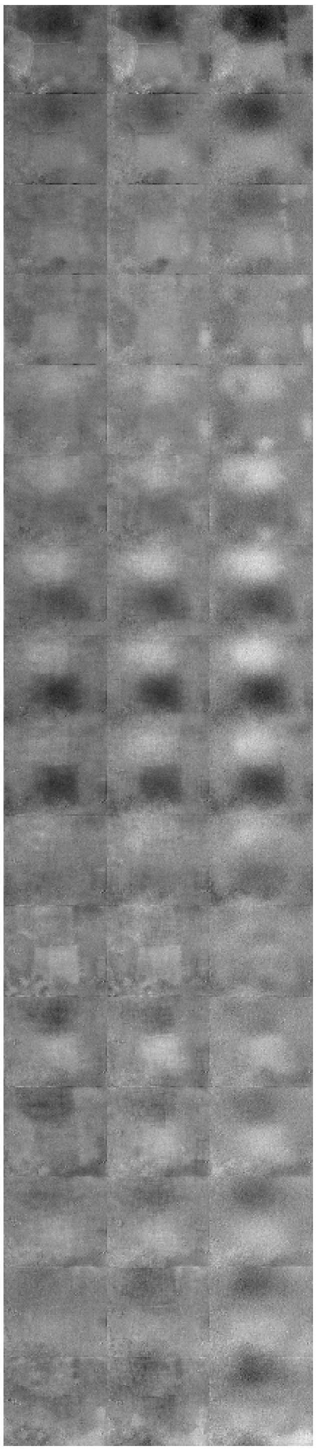

(b)

Figure B.7: (a) Left-to-right: Original images (Objects), Reconstructed spectral images with 2D Symmlet + 1D DCT (Kronecker) transform, convolutional dictionary, convolutional dictionary with Tikhonov regularization under 40 dB SNR (b) The difference between original image and reconstructed images

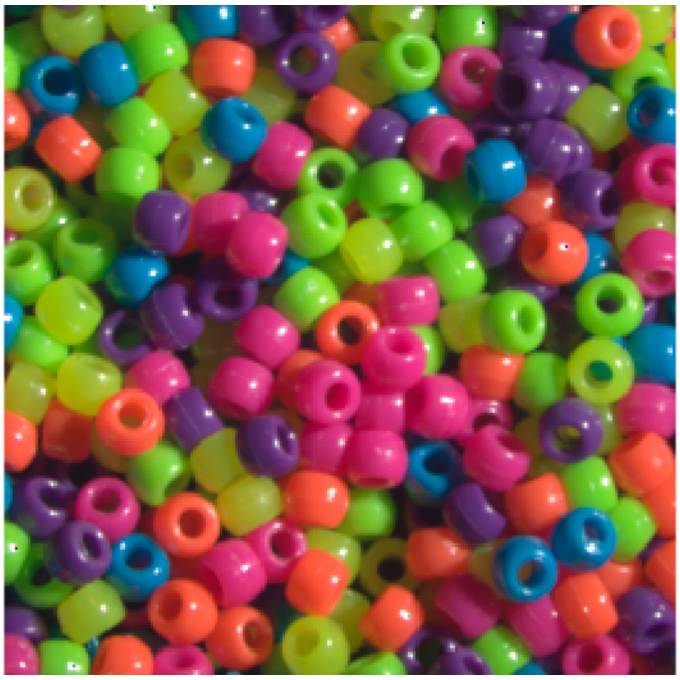


Figure B.8: Beads data colored

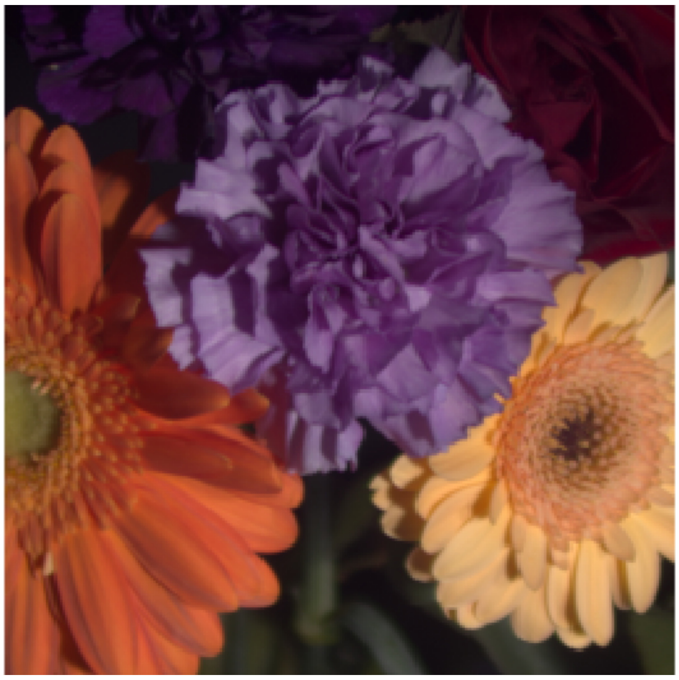


Figure B.9: Flowers data colored

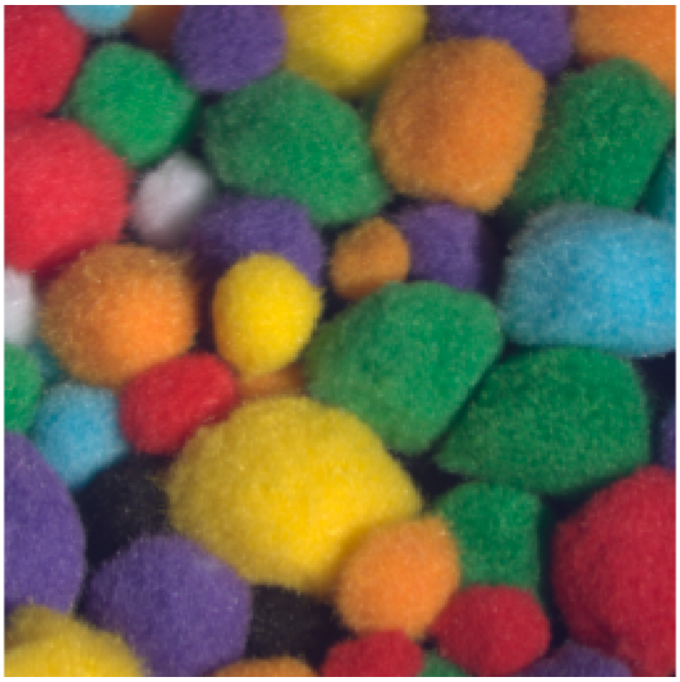


Figure B.10: Pompoms data colored

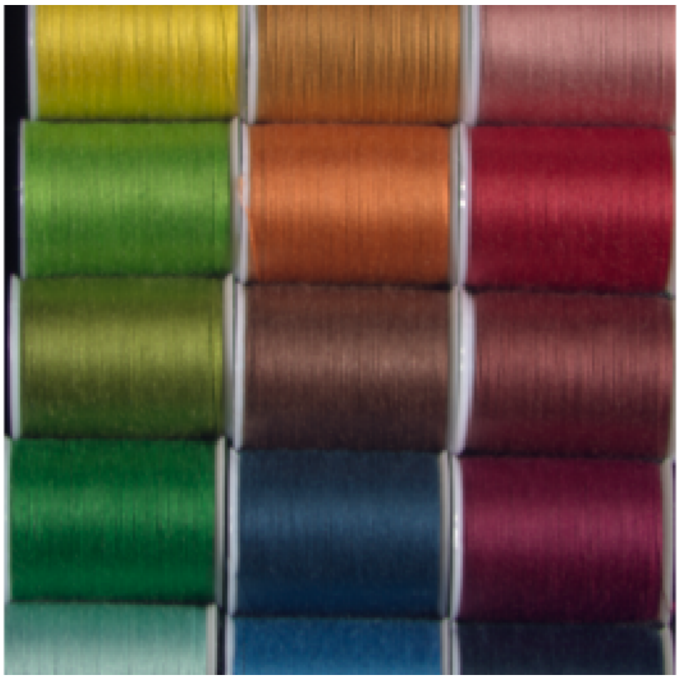
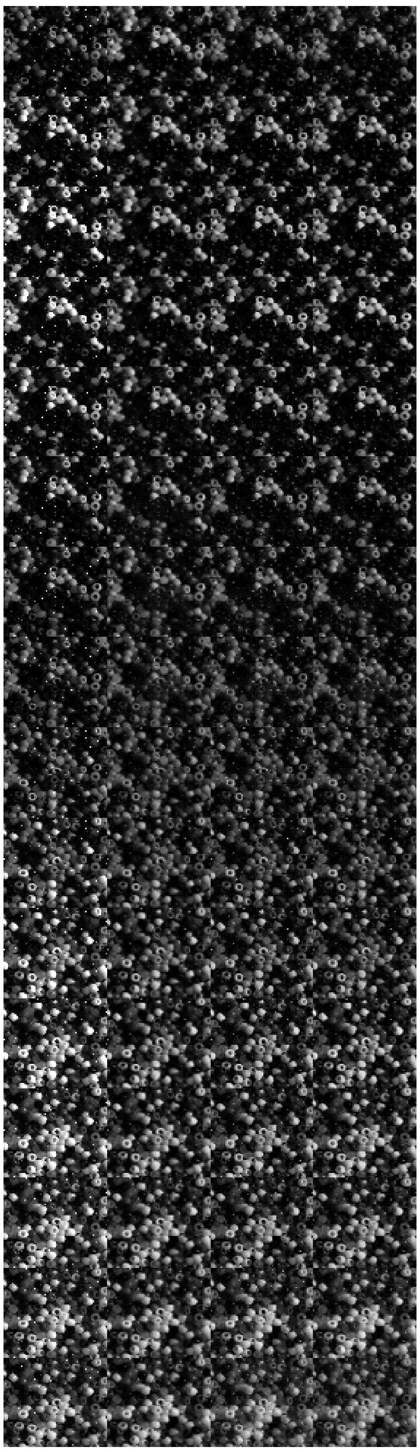


Figure B.11: Threads data colored


(a)

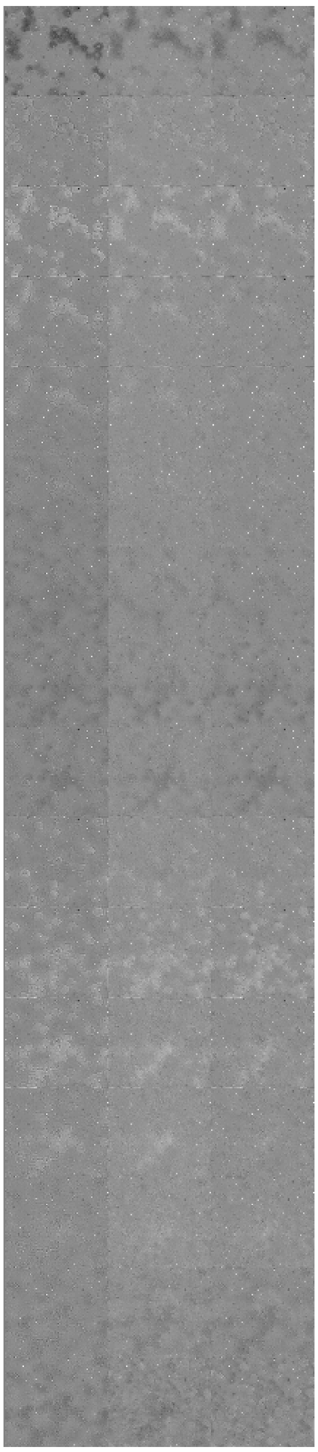
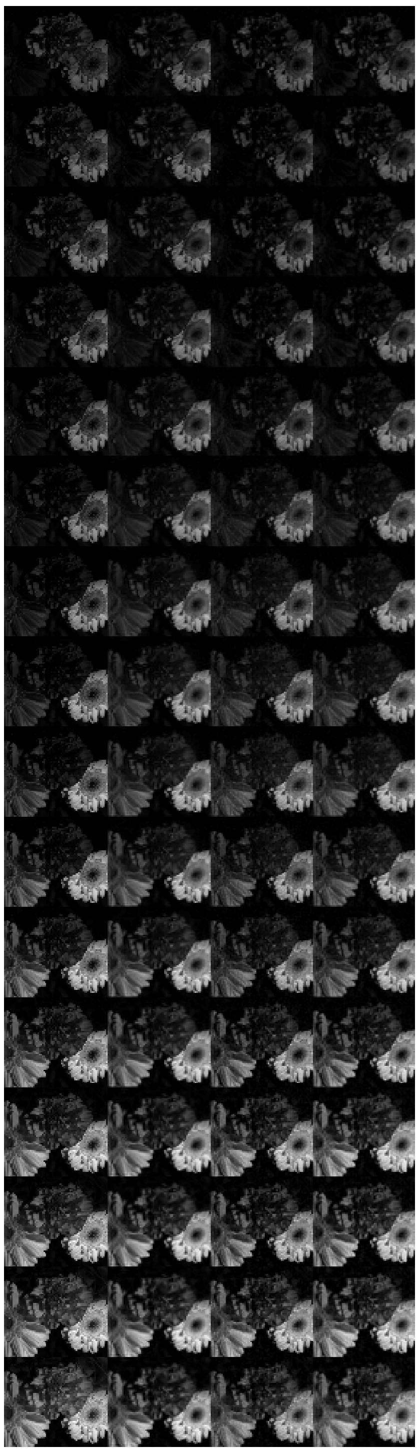

(b)

Figure B.12: (a) Left-to-right: Original images (Beads), Reconstructed spectral images with 2D Symmlet + 1D DCT (Kronecker) transform, convolutional dictionary, convolutional dictionary with Tikhonov regularization under 20 dB SNR (b) The difference between original image and reconstructed images


(a)

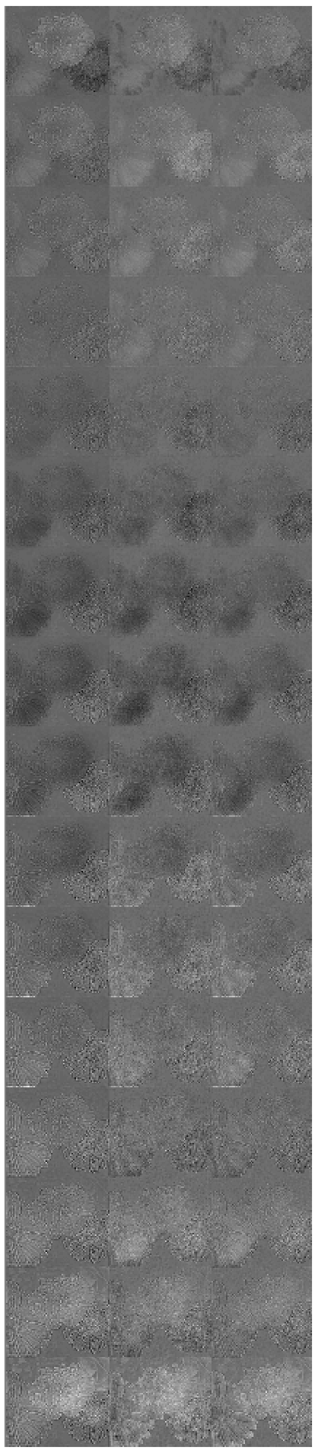
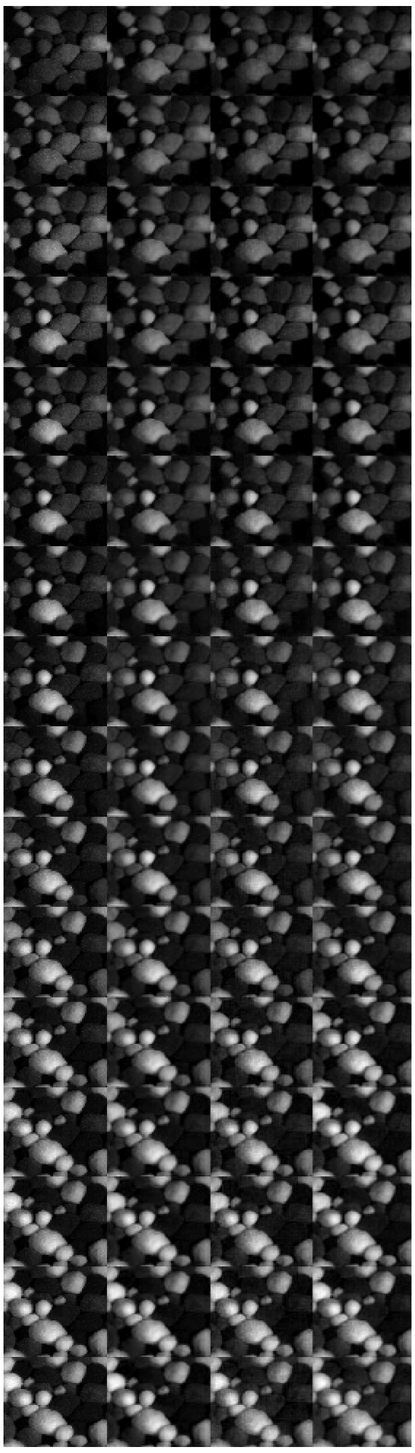

(b)

Figure B.13: (a) Left-to-right: Original images (Flowers), Reconstructed spectral images with 2D Symmlet + 1D DCT (Kronecker) transform, convolutional dictionary, convolutional dictionary with Tikhonov regularization under 20 dB SNR (b) The difference between original image and reconstructed images


(a)

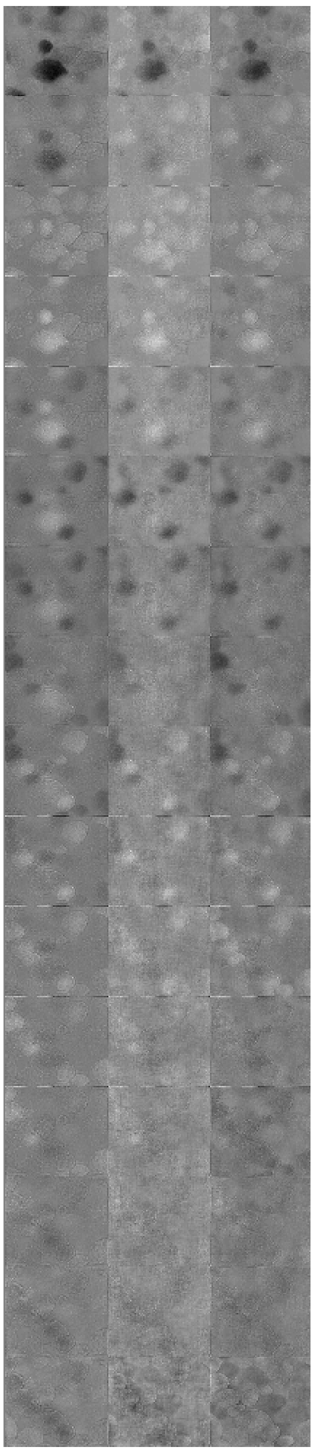
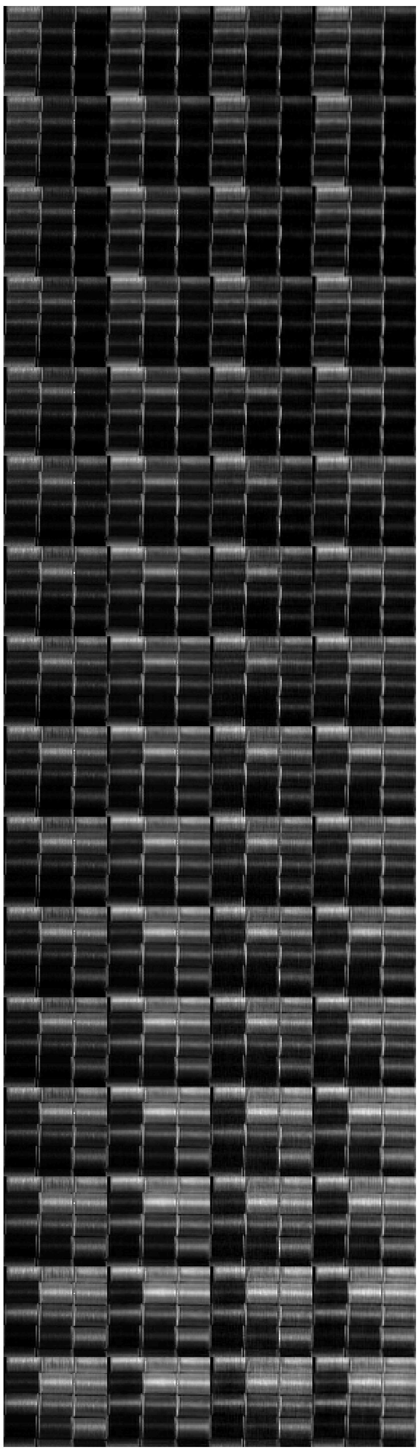

(b)

Figure B.14: (a) Left-to-right: Original images (Pompoms), Reconstructed spectral images with 2D Symmlet + 1D DCT (Kronecker) transform, convolutional dictionary, convolutional dictionary with Tikhonov regularization under 20 dB SNR (b) The difference between original image and reconstructed images


(a)

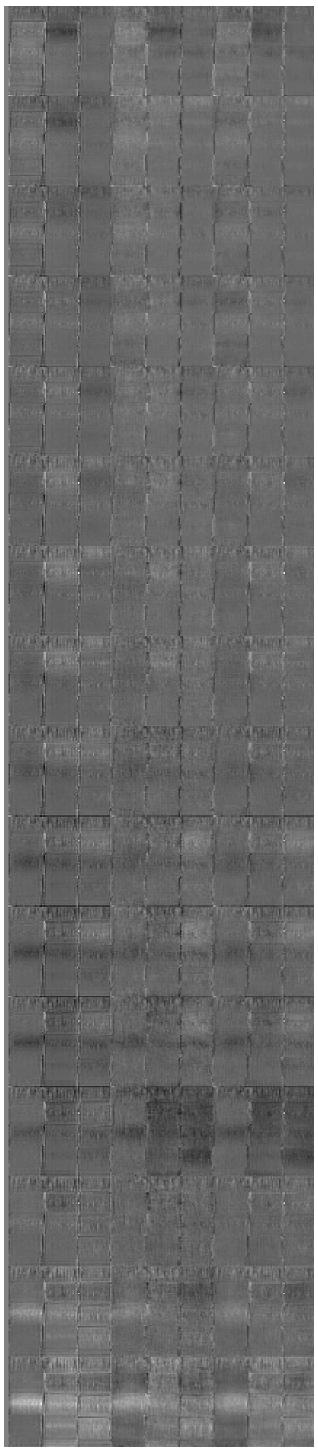
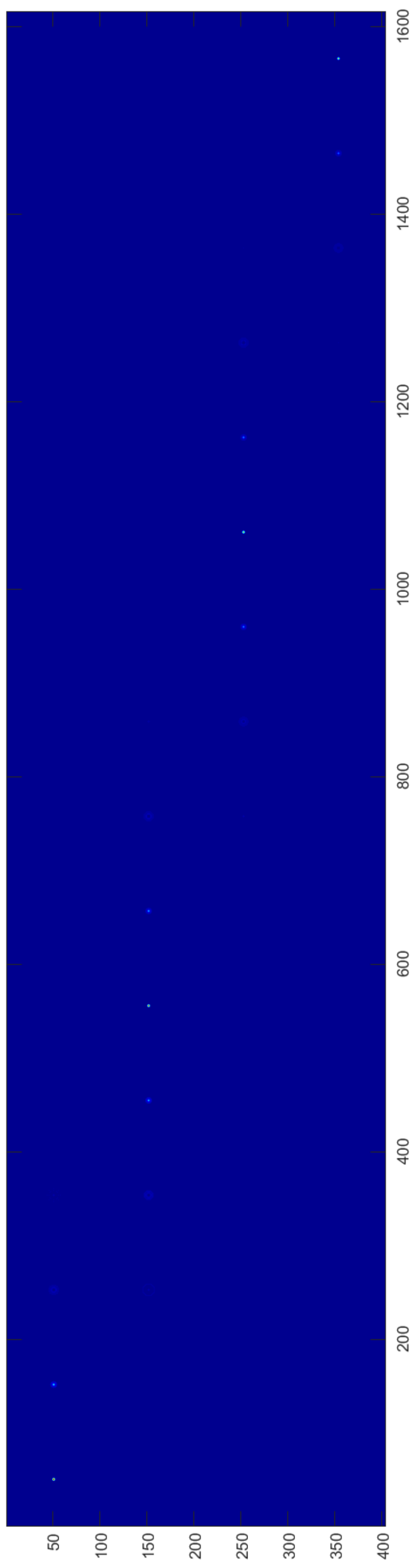
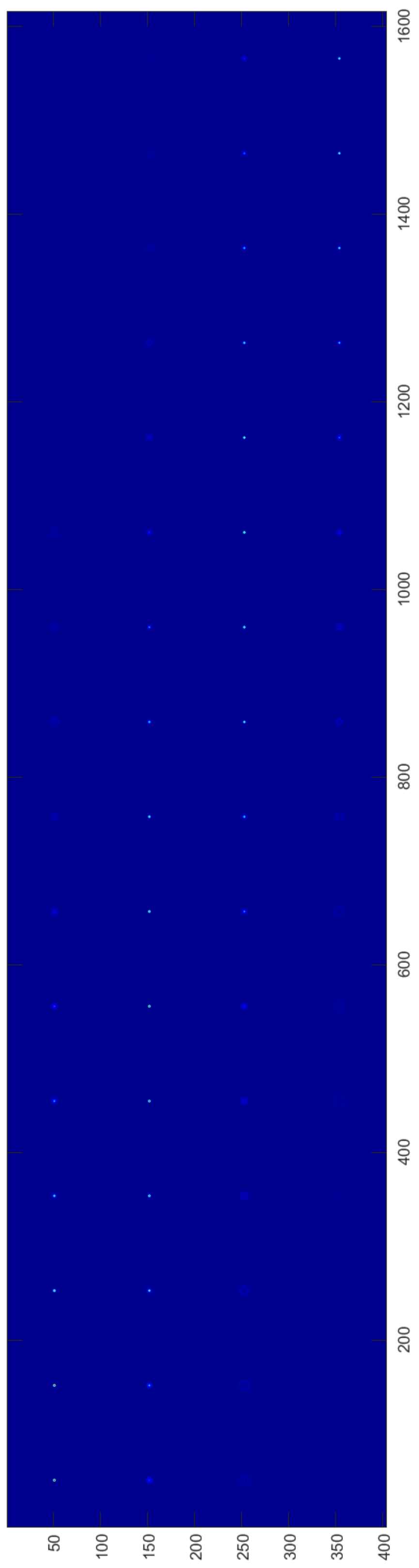

(b)

Figure B.15: (a) Left-to-right: Original images (Threads), Reconstructed spectral images with 2D Symmlet + 1D DCT (Kronecker) transform, convolutional dictionary, convolutional dictionary with Tikhonov regularization under 20 dB SNR (b) The difference between original image and reconstructed images

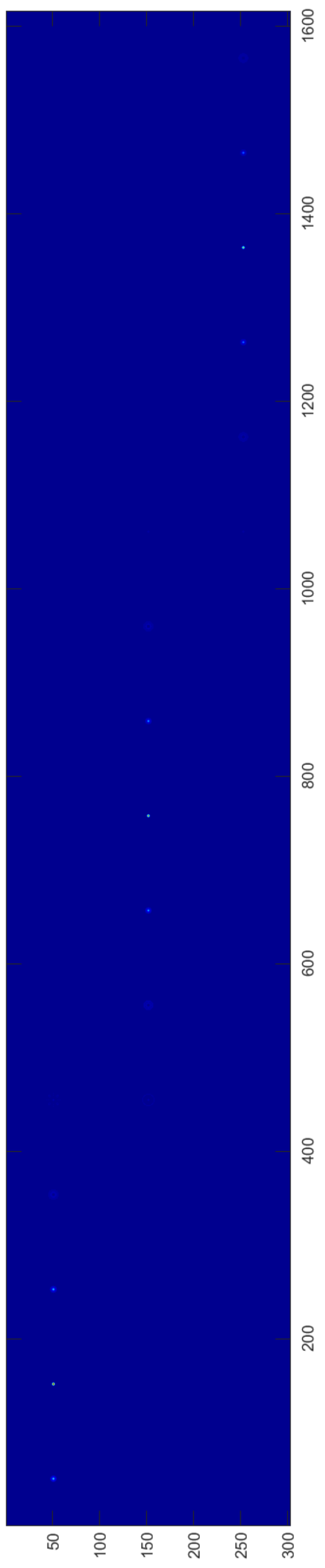


(a)

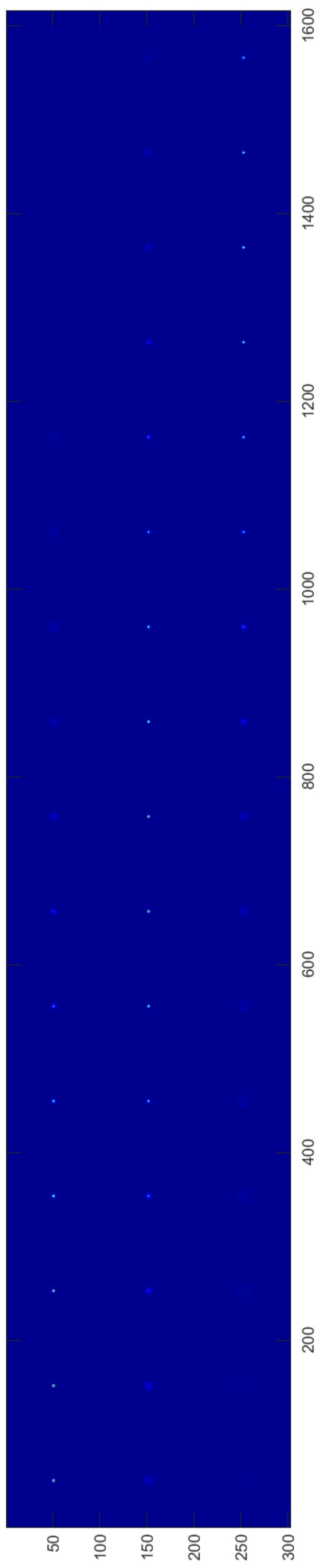


(b)

Figure B.16: (a) Concatenated PSFs for 4 measurements in Design I (b) Concatenated PSFs for 4 measurements in Design II



(a)



(b)

Figure B.17: (a) Concatenated PSFs for 3 measurements in Design I (b) Concatenated PSFs for 3 measurements in Design II

REFERENCES

[1] F. S. Oktem, G. Liang, and F. Kamalabadi. “Computational spectral and ultrafast imaging via convex optimization ,”*Handbook of Convex Optimization Methods in Imaging Science* pp. 105-127, 2018.

[2] L. Gao, and V. W. Lihong “A review of snapshot multidimensional optical imaging: measuring photon tags in parallel,” *Physics reports* vol. 616, pp. 1–37, 2016.

[3] L. Gao, L. Jinyang, L. Chiye, and V. W. Lihong “Single-shot compressed ultrafast photography at one hundred billion frames per second,” *Nature*, vol. 516, no. 7529, pp. 74–77, 2014.

[4] X. Cao, T. Yue, X. Lin, S. Lin, X. Yuan, Q. Dai, L. Carin, and D. J. Brady, “Computational snapshot multispectral cameras: Toward dynamic capture of the spectral world,” *IEEE Signal Processing Magazine*, vol. 33, no. 5, pp. 95–108, 2016.

[5] J. Schlemper, J. Caballero, J. V. Hajnal, A. N. Price, and D. Rueckert, “A deep cascade of convolutional neural networks for dynamic MR image reconstruction,” *IEEE transactions on Medical Imaging*, vol. 37 no. 2, 491–503, 2017.

[6] N. Antipa, G. Kuo, R. Heckel, B. Mildenhall, E. Bostan, R. Ng, and L. Waller, "DiffuserCam: lensless single-exposure 3D imaging.” *Optica*, vol.5. no.1 pp. 1–9, 2018.

[7] W. Dong, L. Zhang, G. Shi, and X. Wu, “Image denoising via sparse and redundant representations over learned dictionaries,” *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[8] W. Dong, L. Zhang, G. Shi, and X. Wu, “Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization,” *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 1838–1857, 2011.

[9] H. Zhang, J. Yang, Y. Zhang, and T. S. Huang, “Sparse representation based blind image deblurring,” in *2011 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, pp. 1–6, 2011.

[10] G. Harikumar and Y. Bresler. Exact image deconvolution from multiple FIR blurs. *IEEE Transactions on Image Processing*, vol. 8, no. 6, pp. 846–862, 1999.

[11] F. S. Oktem, F. Kamalabadi, and J. M. Davila, “High-resolution computational spectral imaging with photon sieves,” *IEEE Int. Conf. on Image Processing (ICIP)*,IEEE, pp. 5122–5126, 2014.

[12] U. Kamaci, F. C. Akyon, T. Alkanat, and F. S. Oktem, "Efficient sparsity-based inversion for photon-sieve spectral imagers with transform learning." In *Proc. of IEEE GlobalSIP Conference*, pp. 1225–1229, 2017.

[13] O. F. Kar and F. S. Oktem, “Compressive spectral imaging with diffractive lenses,” *Optics Letters* 44, 4582–4585, 2019.

[14] D. Dogan and F. S. Oktem, “Convolutional Inverse Problems in Imaging with Convolutional Sparse Models,” *OSA Imaging and Applied Optics Congress*, Munich, Germany, 2019.

[15] S. Ravishankar and Y. Bresler, “Closed-form solutions within sparsifying transform learning,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP),* IEEE, pp. 5378–5382, 2013.

[16] M. F. Duarte, and R. G. Baraniuk,“Kronecker compressive sensing,” *IEEE Transactions on Image Processing*, vol. 21, no. 8, pp. 494–504, 2011.

[17] S. Ravishankar and Y. Bresler, “Efficient blind compressed sensing using sparsifying transforms with convergence guarantees and application to magnetic resonance imaging,” *SIAM Journal on Imaging Sciences*, vol. 8, no. 4, pp. 2519–2557, 2015.

[18] M. Aharon, M. Elad, and A. Bruckstein, “K–SVD: An Algorithm for Designing Overcomplete Dictionaries for Spaese Representation,” *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[19] S. Ravishankar and Y. Bresler, "MR image reconstruction from highly undersampled k-space data by dictionary learning," *IEEE Transactions on Medical Imaging*, vol. 30, no. 5, pp. 1028–1041, 2011.

[20] J. Caballero, A. N. Price, D. Rueckert, and J. V. Hajnal. "Dictionary learning and time sparsity for dynamic MR data reconstruction," *IEEE Trans. Med. Imaging,* vol. 33 pp. 979–994, 2014.

[21] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE J. Sel. Topics Appl. Earth Observ. in Remote Sens.,* vol. 5, no. 2, pp. 354–379, 2012.

[22] Z. Xing, M. Zhou, A. Castrodad, G. Sapiro, and L. Carin, "Dictionary learning for noisy and incomplete hyperspectral images," *SIAM Journal on Imaging Sciences,* vol. 5, no. 1, pp. 33–56, 2012.

[23] A. S. Charles, B. A. Olshausen, and C. J. Rozell, "Learning sparse codes for hyperspectral imagery," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 5, pp. 963–978, 2011.

[24] O. Lorintiu, H. Liebgott, M. Alessandrini, O. Bernard, and D. Friboulet, "Compressed Sensing Reconstruction of 3D Ultrasound Data Using Dictionary Learning and Line-Wise Subsampling,"*in IEEE Transactions on Medical Imaging*, vol. 34, no. 12, pp. 2467-2477, 2015.

[25] A. Rajwade, D. Kittle, T. Tsai, D. Brady, and L. Carin, "Coded hyperspectral imaging and blind compressive sensing," *SIAM Journal on Imaging Sciences*, vol. 6, no. 2, pp. 782–812, 2013.

[26] M. Elad, M. A.T. Figueiredo, and Y. Ma, "On the role of sparse and redundant representations in image processing," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 972–982, 2010.

[27] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-laplacian priors," in *Advances in Neural Information Processing Systems*, pp. 1033–1041, 2009.

[28] H. Bristow, A. Eriksson and S. Lucey, "Fast Convolutional Sparse Coding," *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Portland, OR, pp. 391–398, 2013.

[29] F. Heide, W. Heidrich and G. Wetzstein, "Fast and flexible convolutional sparse coding," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, pp. 5135–5143, 2015.

[30] B. Wohlberg. "Efficient algorithms for convolutional sparse representations." *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 301–315, 2016.

[31] C. Garcia-Cordona and B. Wohlberg. "Convolutional dictionary learning: A comparative review and new algorithms." *IEEE Transactions on Image Processing*, vol. 4, no. 3, pp. 366–381, 2018.

[32] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang, "Convolutional sparse coding for image super-resolution," *IEEE International Conference on Computer Vision*, pp. 1–9, 2015.

[33] H. Wu, S. Zhao, J. Zhang, and C. Lu "Remote sensing image sharpening by integration multispectral image super-resolution and convolutional sparse representation fusion," , vol. 7, 2019.

[34] Y. Liu, X. Chen, R. K. Ward, and Z. J. Wang "Image fusion with convolutional sparse representation," *IEEE Signal Processing Letters*, vol. 23, no. 12, pp. 1882–1886, 2016.

[35] H. Zhang and M. Patel "Convolutional sparse and low-rank coding-based rain streak removal," *2017 IEEE Winter Conference on Applications of Computer Vision*, pp. 1–9, 2017.

[36] X. Hu, F. Heide, Q. Dai and G. Wetzstein. "Convolutional sparse coding for RGB+NIR imaging," *IEEE Transactions on Image Processing*, vol. 27, no. 4, pp. 1611–1625, 2018.

[37] C. G. Cardona and B. Wohlberg. "Convolutional dictionary learning for multi-channel signals," *Asilomar Conference on Signals, Systems, and Computers* , pp. 335–342, 2018.

[38] K. Degraux, U. S. Kamilov, P. T. Boufounos and D. Liu, “Online convolutional dictionary learning for multimodal imaging, ” *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 1617–1621, 2017.

[39] T. M. Quan and W. Jeong, “Compressed sensing reconstruction of dynamic contrast-enhanced MRI using GPU-accelerated convolutional sparse coding,” *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, pp. 518-521, 2016.

[40] C. Barajas-Solano, J. M. Ramirez, H. Garcia, and H. Arguello, “Tridimensional Convolutional Sparse Coding of Spectral Images,” in *Optical Sensors and Sensing Congress*, Optical Society of America, 2019.

[41] T. N. Duc and W. K. Jeong, “Compressed sensing dynamic MRI reconstruction using multi-scale 3D convolutional sparse coding with elastic net regularization," *2018 IEEE International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 1–4, 2018.

[42] D. Nguyen, T. M. Quan, W. K. Jeong, “Frequency-splitting Dynamic MRI Reconstruction using Multi-scale 3D Convolutional Sparse Coding and Automatic Parameter Selection,” *Medical Image Analysis,* vol 53, 2019.

[43] A. N. Tikhonov and V. I. Arsenin, *Solutions of ill-posed problems*, vol. 14. Vh Winston, 1977.

[44] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of inverse problems*, vol. 375. Springer Science & Business Media, 1996.

[45] M. Bertero and P. Boccacci, *Introduction to inverse problems in imaging*. CRC press, 1998.

[46] J. Kaipio and E. Somersalo,*Statistical and computational inverse problems* , vol. 160. Springer Science & Business Media, 2006.

[47] S. Geman and D. Geman, “Stochastic relaxation, gibbs distributions, and the bayesian restoration of images,” in *Readings in Computer Vision*, pp. 564–584, Elsevier, 1987.

[48] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 681–695, 2011.

[49] D. Dogan and F. S. Oktem, "Model-based Inversion Methods for Compressive Spectral Imaging with Diffractive Lenses," submitted

[50] D. Dogan and F. S. Oktem, "Comparison of Dictionary-based Sparse Reconstruction Algorithms for Inverse Problems," *Accepted for 26th Signal Processing and Communications Applications Conference (SIU),* pp. 1–4, IEEE, 2020.

[51] D. Dogan and F. S. Oktem, "Efficient Algorithms for Convolutional Inverse Problems in Multidimensional Imaging," in preparation

[52] A. Chambolle, "An algorithm for total variation minimization and applications,"*Journal of Mathematical Imaging and Vision*, vol. 20, no. 1-2, pp. 89–97, 2004.

[53] S. V. Venkatakrishnan and B. Wohlberg, "Convolutional Dictionary Regularizers for Tomographic Inversion," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7820–7824, 2019.

[54] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Transactions on Image Processing*, vol. 19, no. 9, pp. 2345–2356, 2010.

[55] A. Bruckstein, D. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM Rev.*, vol. 51, no. 1, pp. 34–81, 2009.

[56] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," *In Proceedings of the 26th Annual International Conference on Machine Learning,* 689–696.

[57] A. Beck and M. Teboulle, "Gradient-based algorithms with applications to signal recovery," *Convex optimization in signal processing and communications,* pp. 42–88, 2009.

[58] J. A. Tropp and S. J. Wright, "Computational methods for the sparse solution of linear inverse problems," *Proceedings of the IEEE,* vol. 98, no. 6, pp. 948–958, 2010.

[59] J. A. Tropp and S. J. Wright, "Working locally thinking globally - Part I: Theoretical guarantees for convolutional sparse coding," 2017.

[60] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[61] B. Noble and J. W. Daniel, *Applied linear algebra*, vol. 3, Prentice-Hall New Jersey, 1988.

[62] W. Dong, L. Zhang, G. Shi, and X. Li, "Nonlocally centralized sparse representation for image restoration," *IEEE Transactions on Image Processing*, vol. 22, no. 4, pp. 1620–1630, 2013.

[63] K. Skretting, and K. Engan "Recursive Least Squares Dictionary Learning Algorithm ,"*in IEEE Transactions on Signal Processing*, vol. 58 no. 4, pp. 2121–2130, 2015.

[64] B. Wohlberg, "Convolutional Sparse Representations as an Image Model for Impulse Noise Restoration," *in Proceedings of the IEEE Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, 2016

[65] S. V. Venkatakrishnan and B. Wohlberg, "Convolutional Dictionary Regularizers for Tomographic Inversion", *in Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 7820–7824, 2019

[66] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity, "*IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[67] G. Arce, D. Brady, L. Carin, H. Arguello, and D. Kittle, "Compressive coded aperture spectral imaging: An introduction,"*IEEE Signal Processing Magazine* , vol. 31, no. 1, pp. 105–115, 2014.

[68] D. Pesnell and K. Addison, “Solar Dynamics Observatory: Data,” Accessed on: January. 5, 2020. [Online]. Available: https://sdo.gsfc.nasa.gov/data/aiahmi/

[69] S. M. Nascimento, F. P. Ferreira, and D. H. Foster, “Statistics of spatial coneexcitation ratios in natural scenes,”*JOSA A*, vol. 19, no. 8, pp. 1484–1490, 2002.

[70] F. Yasuma, T. Mitsunaga, D. Iso, and S. K. Nayar, “Generalized assorted pixel camera: postcapture control of resolution, dynamic range, and spectrum,”*IEEE Transactions on Image Processing*, vol. 19, no. 9, pp. 2241–2253, 2010.