# Deep Tensor ADMM-Net for Snapshot Compressive Imaging

Jiawei Ma[†] Columbia University
Xiao-Yang Liu[†] Columbia University
Zheng Shou Columbia University
Xin Yuan Nokia Bell Labs
{jm4743, xl2427, zs2262}@columbia.edu xyuan@bell-labs.com

## Abstract

*Snapshot compressive imaging (SCI) systems have been developed to capture high-dimensional ($\geq 3$) signals using low-dimensional off-the-shelf sensors,* i.e.*, mapping multiple video frames into a single measurement frame. One key module of a SCI system is an accurate decoder that recovers the original video frames. However, existing model-based decoding algorithms require exhaustive parameter tuning with prior knowledge and cannot support practical applications due to the extremely long running time. In this paper, we propose a deep tensor ADMM-Net for video SCI systems that provides high-quality decoding in seconds. Firstly, we start with a standard tensor ADMM algorithm, unfold its inference iterations into a layer-wise structure, and design a deep neural network based on tensor operations. Secondly, instead of relying on a pre-specified sparse representation domain, the network learns the domain of low-rank tensor through stochastic gradient descent. It is worth noting that the proposed deep tensor ADMM-Net has potentially mathematical interpretations. On public video data, the simulation results show the proposed method achieves average $0.8 \sim 2.5$ dB improvement in PSNR and $0.07 \sim 0.1$ in SSIM, and $1500\times \sim 3600\times$ speedups over the state-of-the-art methods. On real data captured by SCI cameras, the experimental results show comparable visual results with the state-of-the-art methods but in much shorter running time.*

## 1. Introduction

Inspired by compressive sensing [4, 6, 7], various computational imaging systems [1] have been built, which employ low-dimensional sensors to capture high-dimensional signals by generating *compressed* measurements. One important branch of computational imaging with promising applications is the snapshot compressive imaging (SCI) [19, 25], which utilized a two-dimensional camera to capture the 3D video or spectral data. Different from conventional cameras, such imaging systems adopt *sampling* on a set of consecutive images–from sequenced temporal channels (*i.e.*, CACTI [19, 34]) or with multiple spectral variations (*i.e.*, CASSI [26])–in accordance with the sensing matrix and *summing up* those sampled signals along time or spectrum to obtain the compressed measurements. With this technique, SCI systems [8, 11, 22, 25, 26, 34] can capture the high-speed motion and high-resolution spectral information but with low memory footprints.

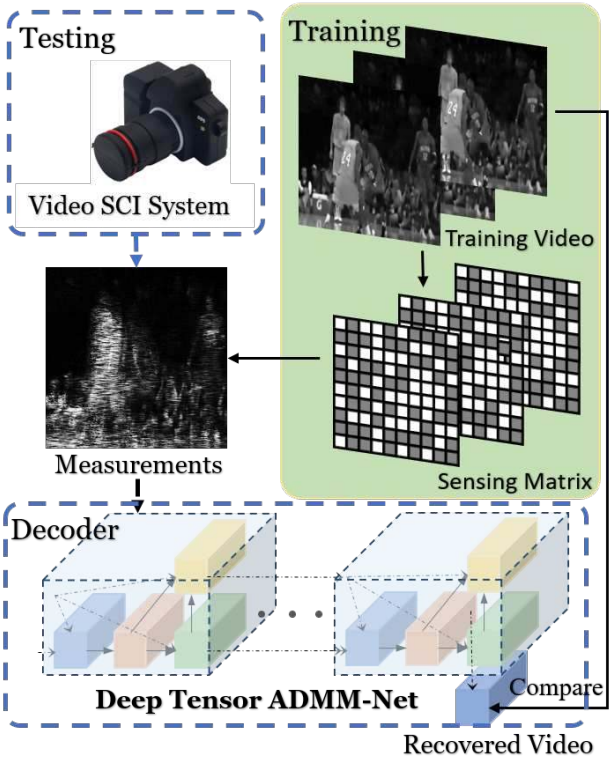


Figure 1. Overview of the deep **tensor ADMM-Net** for SCI systems. The measurement by the video SCI camera [19] is modeled as linear measurements with prior known sensing matrices. We use video samples to train a neural net as a decoder.

In this paper, we focus on the video SCI systems (Fig. 1). Different from other available high-speed cameras that are suffering from great expense and bandwidth cost [23], the video SCI systems [19, 34] enable high-resolution shooting,

[†]Equal contribution. The code is available at `https://github.com/Phoenix-V/tensor-admm-net-sci`.

*e.g.*, NBA slow motion in sports and vehicle crash test in manufacturing with low cost and low bandwidth. Besides, these systems enable the long-time shooting, *e.g.*, aerial photography in topographic survey whose filming duration is greatly limited by the memory size, and traffic monitoring in road networks for which a deluge of video data (*e.g.*, 5 TB per sensor per day) are generated.

In addition to the hardware design, one key module of a SCI system is an accurate decoder, usually called "reconstruction algorithm", that recovers the original video frames. The prior knowledge of desired videos, *e.g.*, sparsity [20], low total variance [33] and low rank, is employed as a regularizer in most of the state-of-the-art video decoding algorithms. However, these decoding algorithms model the videos as a set of matrices and relies on the pre-set knowledge, making it hard to fully exploit the spatial-temporal correlations in video data. The most recently proposed DeSCI algorithm [18] extends the idea of rank minimization by integrating weighted nuclear norm minimization [9] with the alternating direction method of multipliers (ADMM) framework [2] to achieve state-of-the-art results on both video and spectral SCI. Even though the joint models like DeSCI are developed, redundant patch extraction always leads to exhausted processing time.

By contrast, in this paper, intuitively, we view the gray/color video with multiple frames as a *3D/4D tensor* and generalize the techniques in matrix to tensor such that the video decoding task is modeled as a tensor recovery problem from random linear measurements. Inspired by [10, 24], we unfold the algorithm, *tensor nuclear norm minimization using ADMM (TNN-ADMM)*, into a layer-wise deep neural network. As depicted in Fig. 2, each iteration is mapped to a stage of neural network and in every stage each transformation matrix (learned through network training) is regarded as a *pattern*. Thus, a deep network structure is developed by connecting multiple stages in sequence. The output of each reconstruction layer can all be viewed as the recovered signal and compared with the ground truth to calculate the training loss to accelerate its convergence. Specifically, we propose a *deep tensor ADMM-Net* for video SCI systems to provide high-quality decoding within seconds. Our contributions are summarized as follows:

(i) Motivated by the standard tensor ADMM algorithm [36], we generalize it to our video SCI decoding task and propose a deep learning-based but potentially interpretable tensor reconstruction scheme, termed as *tensor ADMM-Net*. We unfold the inference iterations into a novel layer-wise structure that automatically learns the sparse representation domain through network training.

(ii) We exploit the block diagonal structure of sensing matrices in SCI, resulting from the tensor product of the transformation matrix, and deliberately propose a computationally efficient method.

(iii) We design the multi-layer loss minimization and residual structure to capture more details in our tensor ADMM-Net.

(iv) Extensive experiments on both simulation and real-world SCI camera data demonstrate the effectiveness and high-speed of our algorithm. On these simulation videos, the proposed method achieves an average improvement of $0.8 \sim 2.5$ dB in PSNR and $0.07 \sim 0.1$ in SSIM, and $1500\times \sim 3600\times$ speedups over DeSCI. On real-world SCI data experiments, we achieve comparable visual results with the state-of-the-art work but in a much shorter running time.

The remainder of the paper is organized as follows. Section 2 presents related works on SCI. Section 3 briefly introduces the video SCI system and tensor operations. Section 4 provides implementation details of the proposed Tensor ADMM-Net. Section 5 and Section 6 presents the experimental results. Section 7 concludes the paper.

## 2. Related Works

SCI systems have been developed to capture videos [11, 19, 22, 34], 3D spectral images [27, 28, 29, 30], dynamic range images, depth and polarization images, *etc*. From the algorithm side, in addition to the conventional sparsity based algorithms, Gaussian Mixture Model (GMM) in [32] exploits the sparsity of patches and assumes the pixels within a spatial-temporal patch are drawn (at once) from a GMM. GAP-TV model in [33] adopts the idea of total variance minimization under the generalized alternating projection (GAP) framework. Sparse coding [30] has also been developed. As mentioned before, most recently, DeSCI proposed in [18] to reconstruct videos or hyperspectral images in SCI has led to state-of-the-art results. However, most of these algorithms treat video and hyperspectral images in SCI as matrices, while these 3D/4D data indeed is a *tensor*. Therefore, for the first time in the literature, we aim to exploit the tensor based algorithm in SCI reconstruction. The TNN-ADMM algorithm was proposed in [36] and turned to be effective for recovering corrupted images. Nevertheless, TNN-ADMM was only tested on corrupted images without any compressive sensing.

Inspired by recent advances of deep learning on image restoration, researchers have started using deep learning in computational imaging. Some algorithms have been proposed for SCI reconstruction [21]. The models in [24] and [35] successfully unfolded the convex algorithms (ISTA and ADMM) for MRI image reconstruction into deep neural networks. Different from these methods, in this work, we integrate the tensor low-rank model into ADMM-net [24] for SCI reconstruction.

## 3. Video SCI Systems and Tensor Model

We briefly summarize the underlying principle of the video SCI system in Section 3.1. Typical tensor operations are provided in Section 3.2, which are used for problem formulation in Section 3.3.

### 3.1. Snapshot Compressing Imaging System

In the video SCI system (*e.g.*, CACTI) depicted in Fig. 1, consider that a $B$-frame video $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times B}$ is modulated and compressed by $B$ sensing matrices (masks) $\mathcal{C} \in \mathbb{R}^{n_1 \times n_2 \times B}$, and the measurement frame $\boldsymbol{Y} \in \mathbb{R}^{n_1 \times n_2}$ can be expressed as [19, 34]

$$\boldsymbol{Y} = \sum_{b=1}^{B} \mathcal{C}^{(b)} \odot \mathcal{X}^{(b)} + \boldsymbol{N}, \tag{1}$$

where $\boldsymbol{N} \in \mathbb{R}^{n_1 \times n_2}$ denotes the noise; the *frontal slices* $\mathcal{C}^{(b)} = \mathcal{C}(:,:,b)$ and $\mathcal{X}^{(b)} = \mathcal{X}(:,:,b) \in \mathbb{R}^{n_1 \times n_2}$ represent the $b$-th sensing matrix and the corresponding video frame, and $\odot$ denotes the Hadamard (element-wise) product.

Mathematically, the measurement in (1) can be expressed by the following linear equation:

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x} + \boldsymbol{n}, \tag{2}$$

where $\boldsymbol{y} = \text{Vec}(\boldsymbol{Y}) \in \mathbb{R}^{n_1 n_2}$ and $\boldsymbol{n} = \text{Vec}(\boldsymbol{N}) \in \mathbb{R}^{n_1 n_2}$. Correspondingly, the video signal $\boldsymbol{x} \in \mathbb{R}^{n_1 n_2 B}$ is

$$\boldsymbol{x} = \text{Vec}(\mathcal{X}) = [\text{Vec}(\mathcal{X}^{(1)})^\top, ..., \text{Vec}(\mathcal{X}^{(B)})^\top]^\top. \tag{3}$$

Unlike traditional compressive sensing [4, 5, 7], the sensing matrix $\boldsymbol{\Phi} \in \mathbb{R}^{n_1 n_2 \times n_1 n_2 B}$ in video SCI is sparse and exhibits a block diagonal structure as follows

$$\boldsymbol{\Phi} = [\text{diag}(\text{Vec}(\mathcal{C}^{(1)})), ..., \text{diag}(\text{Vec}(\mathcal{C}^{(B)}))]. \tag{4}$$

Consequently, the *sampling rate* here is equal to $1/B$. It has been proved recently in [13] that the reconstruction error of SCI is bounded even when $B > 1$.

As video is a sequence of frames along time, it is intuitively suitable to represent a video as a 3D/4D array (tensor) and treat the video SCI decoding tasks as *reconstructing a third-order tensor from random linear measurements*.

### 3.2. Tensor Operations

Aiming at reconstructing tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times B}$, our SCI decoder rests on the following definition of *Tensor Nuclear-Norm* in Def. 1, and the details of basic tensor operations for drawing TNN are provided in the Supplementary Materials (SM). A 3D tensor $\mathcal{X}$ can be basically viewed as an $n_1 \times n_2$ matrix of tubes lying in the third-dimension. We denote tensor $\widetilde{\mathcal{X}}$ as the frequency domain representation of $\mathcal{X}$, obtained by taking Fourier transform on each tube, *i.e.*, $\widetilde{\mathcal{X}}(i,j,:) = \text{fft}(\mathcal{X}(i,j,:))$.

**Definition 1.** ***Tensor Nuclear-Norm (TNN)*** *[15, 14, 36]. The tensor nuclear norm of a tensor $\mathcal{T}$ is defined as $\|\mathcal{T}\|_{TNN} = \|\overline{\mathcal{T}}\|_*$, where $\|\cdot\|_*$ denotes the matrix nuclear norm,* i.e.*, the sum of singular values of all the frontal slices $\widetilde{\mathcal{T}}^{(b)}$ for $b \in \{1, \ldots, B\}$, and $\overline{\mathcal{T}}$ denotes the* ***diagonal block form*** *of the third-order tensor $\widetilde{\mathcal{T}}$,*

$$\overline{\mathcal{T}} \triangleq \begin{bmatrix} \widetilde{\mathcal{T}}^{(1)} & & & \\ & \widetilde{\mathcal{T}}^{(2)} & & \\ & & \ddots & \\ & & & \widetilde{\mathcal{T}}^{(B)} \end{bmatrix} \in \mathbb{C}^{n_1 B \times n_2 B}. \tag{5}$$

It has been proved that the TNN is the tightest convex relaxation to the $\ell_1$-norm of the tensor multi-rank [36]. By generalizing the Fourier transform to other full-rank time-frequency domain transformation, we denote $\|\mathcal{T}\|_{\Lambda,\text{TNN}} = \|\overline{\mathcal{T}}\|_{\Lambda,*}$ as the TNN of $\mathcal{T}$ under the *transformation* $\Lambda$ [17].

### 3.3. Problem Formulation

In this work, we model the reconstruction task in video SCI system as a tensor recovery problem, and we use the TNN minimization under multiple transform domains as a constraint. It is worth noting that rather than imposing the low-rank property on non-local similar patch groups as in [18], we impose the low-rank property on the entire video (tensor) but in different transform domains [17]. Towards this end, the problem of video SCI reconstruction is formulated as a weighted convex optimization problem in multiple transform domains,

$$\begin{aligned} &\underset{\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times B}}{\text{argmin}} \sum_{f=1}^{F} w_f \|\mathcal{X}\|_{\Lambda_f,\text{TNN}} \\ &\text{s.t. } \boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x} + \boldsymbol{n}, \end{aligned} \tag{6}$$

where $w_f$ denotes the weight corresponding to the transformation $\Lambda_f$ for $f \in \{1, ..., F\}$. Here, in total, we have $F$ transforms. By adopting the general form of transformation matrices, the optimization problem in (6) is equivalent to

$$\underset{\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times B}}{\text{argmin}} \sum_{f=1}^{F} w_f \|\overline{\mathcal{X}_f}\|_{\Lambda_f,*} + \mathbb{1}_{\boldsymbol{y}=\Phi\boldsymbol{x}+\boldsymbol{n}}, \tag{7}$$

where $\Lambda = [\Lambda_1, \ldots, \Lambda_F]$ denotes the transform matrices, $\overline{\mathcal{X}}$ is constructed from $\widetilde{\mathcal{X}_f}$ [17] (which is further constructed from $\mathcal{X}$) as in (5) and $\mathbb{1}$ denotes the indicator function.

For video SCI decoding, we may derive the iterative algorithm for problem (7) and run it dozens of iterations to get a satisfactory reconstruction. However, setting the hyper parameters, *e.g.*, transformation matrix $\Lambda_f$ and related weight $w_f$, is challenging and tuning these parameters for different scenarios is nontrivial. To exploit the learning ability of neural networks, as depicted in Fig. 2, we develop a

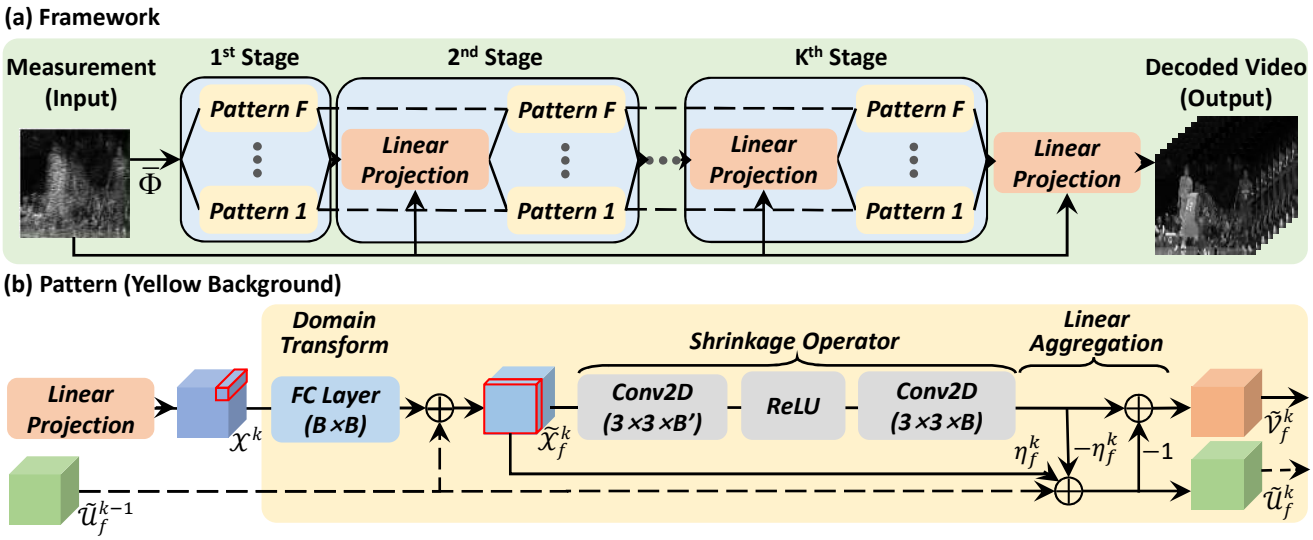


Figure 2. Data flow graph of our proposed Tensor ADMM-Net: (a) The general framework is constructed by connecting each **Stage** in a sequence order. In each **Stage**, multiple transformation **Patterns** are processed in parallel. (b) Details of each **Pattern**. The dashed line denotes the data flow between two consecutive stages while the solid line denotes the data flow inside the same stage.

layer-wise structure based mechanism inside each iteration (stage). Instead of relying on a pre-specified sparse representation domain knowledge, we untie the model parameters across layers to obtain a novel network structure and train the model using the stochastic gradient descent method. In this way, the transformations and weights can be learned in a discriminate manner.

## 4. Deep Tensor ADMM-Net

We first derive the basic formulation of the TNN-ADMM algorithm in Section 4.1 to solve the optimization problem in (7). Then the structure details of our *deep tensor* ADMM-Net is presented in Section 4.2.

### 4.1. TNN-ADMM Algorithm

By using the ADMM framework [2, 3] and introducing auxiliary variables $\widetilde{\mathcal{Z}} = [\widetilde{\mathcal{Z}}_1, ..., \widetilde{\mathcal{Z}}_f]$, (7) can be written as :

$$\begin{aligned} \underset{\mathcal{X},\ \widetilde{\mathcal{Z}}}{\operatorname{argmin}} & \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}\|_2^2 + \sum_{f=1}^{F} w_f \|\overline{\mathcal{Z}}_f\|_{\Lambda_f,*} \\ & \text{s.t. } \widetilde{\mathcal{X}}_f = \widetilde{\mathcal{Z}}_f, \end{aligned} \tag{8}$$

where $\overline{\mathcal{Z}}$ is constructed from $\widetilde{\mathcal{Z}}$ similar to $\mathcal{X}$. This problem can be solved by the following subproblems

$$\begin{aligned} \mathcal{X}^k = \underset{\mathcal{X}}{\operatorname{argmin}} \Bigg\{ & \sum_{f=1}^{F} \left\langle \widetilde{\mathcal{U}}_f^{k-1}, \widetilde{\mathcal{X}}_f - \widetilde{\mathcal{Z}}_f^{k-1} \right\rangle \\ & + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}\|_F^2 + \sum_{f=1}^{F} \frac{\rho_f}{2} \|\widetilde{\mathcal{X}}_f - \widetilde{\mathcal{Z}}_f^{k-1}\|_F^2 \Bigg\}, \end{aligned} \tag{9}$$

$$\begin{aligned} \widetilde{\mathcal{Z}}_f^k = \underset{\widetilde{\mathcal{Z}}_f}{\operatorname{argmin}} \Bigg\{ & \left\langle \widetilde{\mathcal{U}}_f^{k-1}, \widetilde{\mathcal{X}}_f^k - \widetilde{\mathcal{Z}}_f \right\rangle + w_f \|\overline{\mathcal{Z}}_f\|_* \\ & + \frac{\rho_f}{2} \|\widetilde{\mathcal{X}}_f^k - \widetilde{\mathcal{Z}}_f\|_F^2 \Bigg\}, \end{aligned} \tag{10}$$

$$\widetilde{\mathcal{U}}_f^k = \widetilde{\mathcal{U}}_f^{k-1} + \eta_f (\widetilde{\mathcal{X}}_f^k - \widetilde{\mathcal{Z}}_f^k), \tag{11}$$

where $\widetilde{\mathcal{U}} = [\widetilde{\mathcal{U}}_1, ..., \widetilde{\mathcal{U}}_F]$ and $\rho = [\rho_1, ..., \rho_F]$ denote the multipliers and the coefficients of Lagrange expansion in the ADMM framework, respectively, and $\eta_f$ is a constant to determine the step size. Since the transformation matrices are set independently, the updates in (10)-(11) for different transform domain can be processed in parallel and independently in the same iteration. The detailed solutions for (9) to (11) are provided in the SM. As mentioned before, this optimization based algorithm, though may lead to good results, will cost a long time due to the large computational workload. In the following, we employ a deep network to solve this problem, dubbed *deep tensor ADMM-Net*.

### 4.2. Pipeline Design for Tensor ADMM-Net

Derived from (9) to (11), Fig. 2 shows the stage-wise deep model structure. In each stage, we first aggregate the outputs from previous stage by the *Linear Projection* module and then feed the output to another parallel *Patterns* for further processing.

#### 4.2.1 Linear Projection

By adopting the following equation derived from (9), our *Linear Projection* module aggregates the measurement and the outputs from all patterns of previous stage and aims to

reconstruct the desired signal

$$\begin{aligned}\mathcal{X}^k &= \boldsymbol{S}^k(\boldsymbol{\Phi}^\top \boldsymbol{y} + \sum_{f=1}^{F} \rho_f^{k+1} \Pi_f^k (\widetilde{\mathcal{Z}}_f^{k-1} - \widetilde{\mathcal{U}}_f^{k-1})),\\ \boldsymbol{S}^k &= (\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \sum_{f=1}^{F} \rho_f^k \Pi_f^k \Pi_f^{k\top})^{-1},\\ \Pi_f^k &= \Lambda_f^k \otimes \boldsymbol{I}_{n_1 n_2},\end{aligned} \tag{12}$$

where $\boldsymbol{I}_{n_1 n_2} \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$ denotes an identity matrix, $\otimes$ represents the Kronecker (tensor) product and $\Lambda^k = [\Lambda_1^k, \ldots, \Lambda_F^k]$ is the generalized transformation matrices in the $k$-th stage, which are learned during model training as well as the relative parameters $\rho^k = [\rho_1^k, \ldots, \rho_F^k]$. By setting $\widetilde{\mathcal{Z}}^0$ and $\widetilde{\mathcal{U}}^0$ as zero matrices, we define

$$\overline{\boldsymbol{\Phi}} = \boldsymbol{S}^1 \boldsymbol{\Phi}^\top, \tag{13}$$

which will be used to initialize the input of our network (please refer to the input in Fig. 2(a)). Nevertheless, $\boldsymbol{S}^{k+1} \in \mathbb{R}^{n_1 n_2 B \times n_1 n_2 B}$ in each stage will occupy an astounding amount of memory and is not applicable in the gradient calculation. Inspired by the diagonal block structure of $\boldsymbol{\Phi}$ in (4) and tensor-based domain transform, we further investigate the inner structure and reduce the processing complexity.

**Rectangular Diagonal Block (RDB)** structure is basically a $B \times B$ matrix of $n$-by-$n$ symmetric diagonal matrices lying on the plane to form an $nB \times nB$ rectangular matrix. The matrix with RDB structure is supposed to be full-rank and invertable and each symmetric diagonal matrix is termed as a *block*. As shown in Fig. 3, for the matrix $\boldsymbol{W}$ with RDB structure, the green background indicates one block. Instead of calculating $\boldsymbol{W}^{-1}$ directly, the inverse matrix is calculated according to the following steps:

(1) *Split* all $B \times B$ blocks: the elements of the same position are aggregated to build the corresponding smaller matrices and there are in total $n \times n$ *elemental matrix*, *e.g.*, the elements with blue (orange) outline are extracted to build $\boldsymbol{W}_{\text{diag}(1)}$ ($\boldsymbol{W}_{\text{diag}(2)}, \ldots$) individually.

(2) Calculate the *inverse* matrix of all these $n \times n$ *elemental matrix* individually (potentially in parallel), *e.g.*, calculate $\boldsymbol{W}_{\text{diag}(1)}^{-1}$ and $\boldsymbol{W}_{\text{diag}(2)}^{-1}, \ldots$.

(3) *Assemble* the $n \times n$ inverse matrices back to the matrices of RDB structure, *e.g.*, according the position of $\boldsymbol{W}_{\text{diag}(1)}$ ($\boldsymbol{W}_{\text{diag}(2)}, \ldots$) in $\boldsymbol{W}$, $\boldsymbol{W}_{\text{diag}(1)}^{-1}$ and $\boldsymbol{W}_{\text{diag}(2)}^{-1}, \ldots$ are aggregated to draw out $\boldsymbol{W}^{-1}$.

This calculation scheme is theoretically equivalent to the direct matrix inverse computation with detailed derivations provided in the SM.

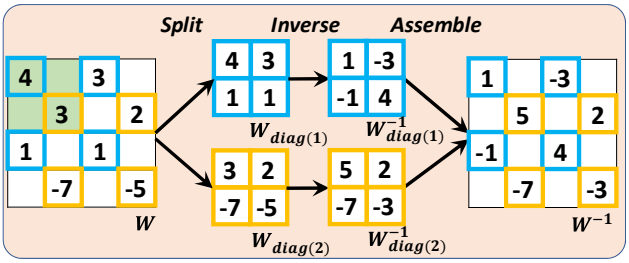


Figure 3. Example of fast inverse calculation (The empty area indicates zero).

According to (4), $\boldsymbol{\Phi}^\top \boldsymbol{\Phi}$ is of the RDB structure. As the time-frequency transform operation is essentially an invertable linear transform, the transform matrix $\Lambda_f^k$ is invertable and thus of full rank. Consequently, the product $\Pi_f^k \Pi_f^{k\top}$ for $f \in \{1, \ldots, F\}$ are of the full-rank RDB structure. Therefore, the inverse of $\boldsymbol{S}^k$ can be efficiently calculated using this framework. In practice, in case that the summation of multiple full-rank RDB matrix is not always full-rank, we add a scaled identity matrix as noise to avoid the gradient explosion caused by the degenerated matrix.

In this manner, both the forward and backward gradient calculation of $\boldsymbol{S}^k$ can be broken into the calculations of $n_1 \times n_2$ small matrices $\boldsymbol{S}_{\text{diag}(i)}^k$ for $i \in \{1, ..., n_1 n_2\}$ where each matrix can be calculated separately and in parallel. By adopting such a strategy, the memory occupancy is decreased from $(n_1 n_2 B)^2$ to $n_1 n_2 B^2$ and the computation complexity of the matrix inversion in each stage is decreased from $\mathcal{O}(n_1^3 n_2^3 B^3)$ to $n_1 n_2 \mathcal{O}(B^3)$ where $n_1 n_2 = 65536$ in our experiment setup. This memory optimization is one of our contribution in SCI reconstruction and can be generalized to other algorithms.

### 4.2.2 Pattern

Derived from (10)-(11), we design the inner structure of each pattern to perform the update of auxiliary variables. According to (10), the update of $\widetilde{\mathcal{Z}}_f^k$ includes both nuclear-norm minimization and least-square items. Thus, we will first introduce the matrix shrinkage operator in Def. 2 with related theorem and then the solution.

**Definition 2.** ***Singular value shrinkage operator***. *Given the* SVD *of a rank-r matrix* $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ *and* $\boldsymbol{\Sigma} = diag(\{\sigma_i\}_{1 \le i \le r})$, *for each* $\tau \ge 0$, *the soft-thresholding operator is defined as a singular value shrinkage operator*

$$\mathcal{D}_\tau(\boldsymbol{X}) = \boldsymbol{U}\mathcal{D}_\tau(\boldsymbol{\Sigma})\boldsymbol{V}^\top, \tag{14}$$

$$\mathcal{D}_\tau(\boldsymbol{\Sigma}) = diag(\{\max(\sigma_i - \tau, 0)\}_{1 \le i \le r}). \tag{15}$$

**Theorem 1.** *[3] For each* $\tau \ge 0$ *and* $\boldsymbol{M} \in \mathbb{R}^{n_1 \times n_2}$, *the singular value shrinkage operator in Def. 2 satisfies*

$$\mathcal{D}_\tau(\boldsymbol{M}) = \underset{\boldsymbol{X}}{argmin} \left\{ \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{M}\|_F^2 + \tau \|\boldsymbol{X}\|_* \right\}. \tag{16}$$

Soft-thresholding is adopted in singular value shrinkage operator for $\ell_1$-norm minimization and this theorem indicates such an operation on an auxiliary matrix $\boldsymbol{M}$ is equivalent to the minimization consisting of least-square and $\ell_1$-norm items. The solution of (10) can be described as $\widetilde{\mathcal{Z}}^k = \mathcal{D}_{1/\rho}(\widetilde{\mathcal{U}}^{k-1} + \widetilde{\mathcal{X}}^k)$. Since the TNN is defined as the sum of singular values of all the frontal slices, the shrinkage operator can be applied to each frontal slice individually.

**Domain Transformer**: As shown in Fig. 2(b), the fully connected (FC) layer at the beginning of each pattern works as a domain transformer. Each tube $\mathcal{X}_f^k(i,j,:) \in \mathbb{R}^{1\times B}$ is fed into the FC layer in parallel and individually.

**Shrinkage Operator**: The work in [12] declares that multilayer feed forward networks (FFN) are universal approximates for any vector-valued function. In addition, a shrinkage operator is essentially a nonlinear function adopted in each entry of the video and can be described in vector-valued form. As the operator is adopted on images, Thus, we apply 2D Convolution layers on the frontal slices where the number of kernels in each layer is $B' > B$ except for the last layer, *i.e.*, we treat the pixels along third-dimension as a feature vector corresponding to each spatial unit.

**Linear Aggregation**: The linear aggregation of the pattern is derived for the multiplier $\widetilde{\mathcal{U}}_f^k$ update in (11). The constant $\eta_f$ is treated as trainable variable so that the step size for different pattern in different stages varies. Also, for the convenience of calculation in Section 4.2.1, we directly calculate $\widetilde{\mathcal{V}}_f^k = \widetilde{\mathcal{Z}}_f^k - \widetilde{\mathcal{U}}_f^k$ as one of the pattern's output.

# 5. Performance Evaluation on Simulation Data

We first verify the performance of the proposed deep Tensor ADMM-net on the simulation data and then apply it to the real data captured by the SCI cameras [19, 34].

## 5.1. Data Sets

We evaluate the proposed deep *Tensor ADMM-Net* on the simulation data including *Kobe* dataset [32], *Aerial* dataset and *Vehicle Crash* dataset, respectively. We collected *Aerial*, *vehicle*, and basketball shooting (*NBA*) (from YouTube) to train the model. We resize the original images to $256 \times 256$ through down sampling. Following the setting in [18], eight ($B = 8$) consequent frames are modulated by shifting binary masks and then collapsed to a single measurement. Each dataset contains 32 frames and thus 4 measurements.

## 5.2. Compared Methods and Performance Metrics

As mentioned before, various algorithms have been proposed for SCI reconstruction. Within these algorithms, GAP-TV [33] is a good baseline to provide decent results within several seconds and DeSCI has led to state-of-the-art results. Therefore, in the following, we compare our proposed method with these two methods.

**GAP-TV** [33]: The algorithm models the reconstruction of video SCI as a total variation minimization problem. It solves the following problem

$$\widehat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \ \|\mathrm{TV}(\boldsymbol{x})\| \quad \text{s.t. } \boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}, \tag{17}$$

where TV indicates the total variation of the signal.

**DeSCI** [18]: The Decompress SCI algorithm integrates the mathematical model of SCI system with the idea of weighted nuclear norm. By minimizing the nuclear norm of each patch group, the model recovers the signal with the minimized rank. Under the ADMM framework, DeSCI solves the following problem

$$\widehat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \sum_i \|\boldsymbol{Z}_i\|_{w,*} \quad \text{s.t. } \boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}, \tag{18}$$

where $\|\boldsymbol{Z}_i\|_{w,*}$ indicates the matrix nuclear norm weighted by $w$ and each patch group $\boldsymbol{Z}_i$ is extracted from $\boldsymbol{x}$.

We run these three algorithms on 3 simulation data sets and real data sets captured by the video SCI system. Since the number of measurements in each simulation data sets varies, we use the mean value of all testing trials to evaluate the reconstruction performance comprehensively. The following three metrics are employed to compare different methods:

- Peak Signal to Noise Ratio (**PSNR**);
- Structural Similarity Index (**SSIM**) [31];
- **Running time**. We measure the running time of decoding one measurement frame.

## 5.3. Implementation Details

After image resizing, all of the data sets, *i.e.*, *Aerial*, *Vehicle* and *NBA*, contain $500$ frames, and these frames are split for training, validation an testing with the ratio of $85\%, 7.5\%, 7.5\%$, respectively. The model trained by *NBA* is tested on *Kobe* directly. The performance evaluation of the rest data sets is based on the test data respectively.

During our model training, we set the maximum running epoch as 100 and the initial learning rate as 0.01. The rooted-square-mean-error (RMSE) is used as the training loss which is minimized by the Adam optimizer [16]. We implement our model with Tensorflow and conduct experiment on a NVIDIA Tesla V100 GPU, with 32GB device memory and 3072 CUDA cores running at 1.11GHz.

## 5.4. Simulation Results

Table 1. Average PSNR (dB) on different data sets

| Algorithm | Kobe | Aerial | Vehicle |
|---|---|---|---|
| Tensor ADMM-Net | 30.15 | **26.85** | **23.62** |
| GAP-TV | 26.45 | 24.53 | 22.85 |
| DeSCI | **33.25** | 24.95 | 21.16 |

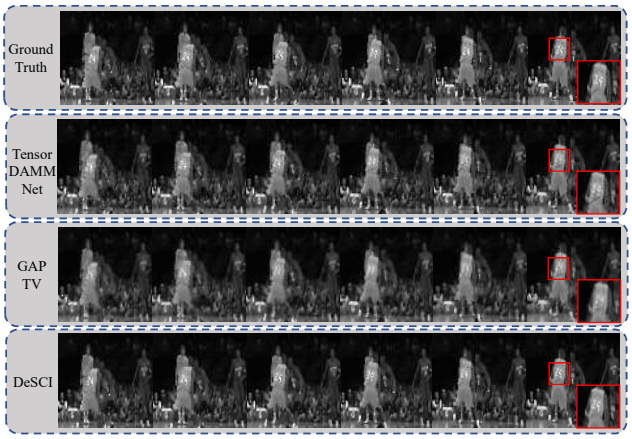


Figure 4. *Kobe*: Results of Tensor ADMM-Net (second row), GAP-TV (third row) and DeSCI (fourth row) compared with ground truth (first row).

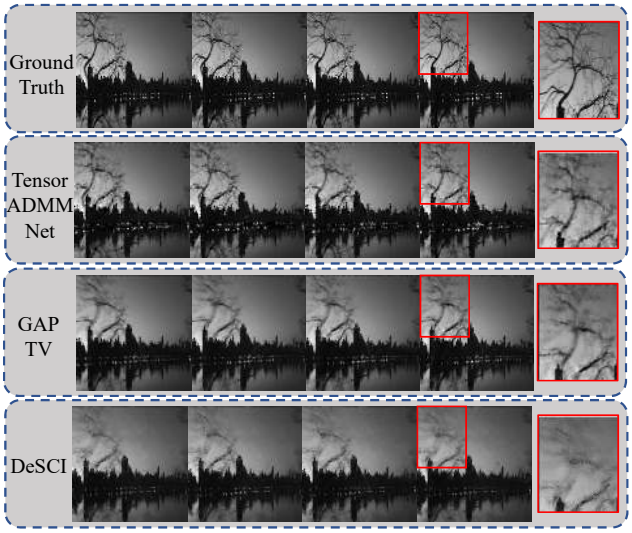


Figure 5. *Aerial*: Results of Tensor ADMM-Net (second row), GAP-TV (third row) and DeSCI (fourth row) compared with ground truth (first row).

Table 2. Average SSIM on different data sets

| Algorithm | Kobe | Aerial | Vehicle |
|---|---|---|---|
| Tensor ADMM-Net | 0.89 | **0.86** | **0.78** |
| GAP-TV | 0.84 | 0.84 | 0.77 |
| DeSCI | **0.95** | 0.80 | 0.70 |

Table 3. Running time (seconds) on different data sets ($B = 8$)

| Algorithm | Kobe | Aerial | Vehicle |
|---|---|---|---|
| Tensor ADMM-Net | **1.9** | **2.4** | **2.1** |
| GAP-TV | 7.9 | 6.9 | 7.2 |
| DeSCI | 6872.9 | 6915.8 | 6823.5 |

Figs. 4-6 show the results of the Tensor ADMM-Net on these three datasets compared with GAP-TV and DeSCI. The corresponding performance comparisons are given in Tables 1-3. It can be observed that though DeSCI leads to the best

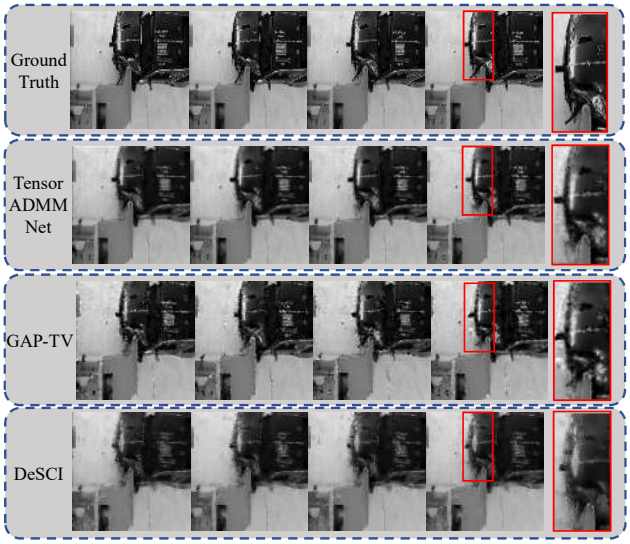


Figure 6. *Vehicle Crash*: Results of Tensor ADMM-Net (second row), GAP-TV (third row) and DeSCI (fourth row) compared with ground truth (first row).

Figure 7. Real video SCI system measurements: *Wheel*, *Balls* and *Hammer*. **Wheel:** Different characters are attached on a fan while the fan is preforming high-speed rotation ($B = 14$ and all frames are shown in Fig. 8). **Ball:** Two plastic balls drop freely and hit the ground. Then, the rotation and rebounding happen on the two objects respectively ($B = 22$ and every other frame is selected and shown in Fig.9). **Hammer:** A hammer, swinging like pendulum, knock down a plastic apple ($B = 22$ and every other frame in shown in Fig. 10).

results on the *Kobe* dataset, our proposed Tensor ADMM-Net provides better results than DeSCI on the *Aerial* and *Vehicle Crash* datasets. This is reasonable since DeSCI relies on the similar patches across the video frames. However, it is challenging to seek similar patches in the latter two videos. Due to this same reason, DeSCI only improved a little bit on PSNR (0.4dB) over GAP-TV for the *Aerial* data while the SSIM is (0.04) lower than GAP-TV. Furthermore, the PSNR and SSIM of DeSCI for the *Vehicle Crash* dataset are both lower than those of GAP-TV. It is worth noting that GAP-TV is used as the initialization of DeSCI. When similar patches cannot be found in the video, DeSCI can not improve the results no matter how long it runs. From Table 3), we see that the proposed Tensor ADMM-Net achieves $1500\times \sim 3600\times$ speedups over DeSCI.

From the visualization results shown in Fig. 5-6, we can observe that DeSCI smooths out the details in the reconstructed video. By contrast, our algorithm provides more

details than the other two methods. One main reason is that our model applies the theory of tensor to capture both the sparsity in transformation domain and the low rank in spectral domain, and thus to capture spatial and temporal relationship in a better way. It is worth noting that instead of hours taken by DeSCI, after training, our Tensor ADMM-Net can provide results within seconds.

## 6. Results on Real SCI Data


Figure 8. Real data *Wheel*: results for Tensor ADMM-Net, GAP-TV and DeSCI.

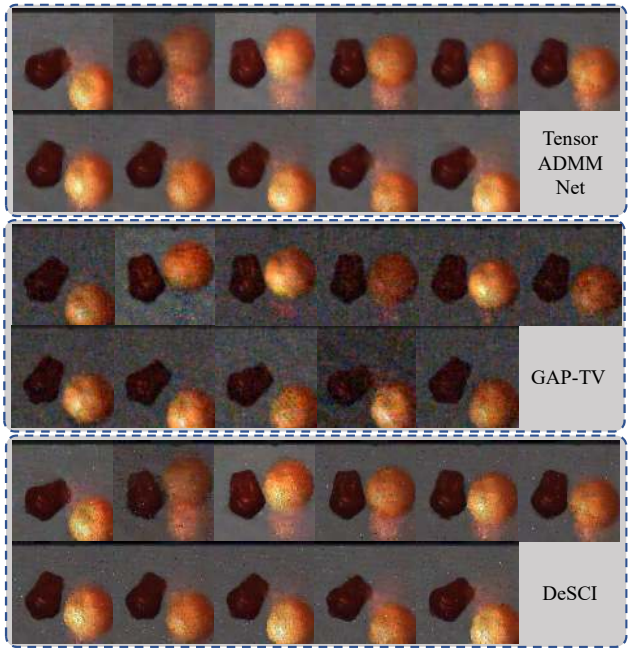


Figure 9. Real data *Balls*: results of Tensor ADMM-Net, GAP-TV and DeSCI.

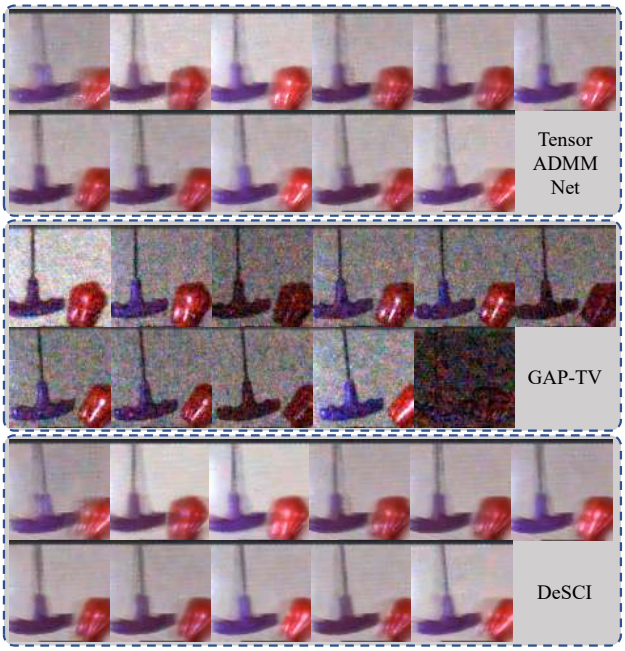


Figure 10. Real data *Hammer*: results for Tensor ADMM-Net, GAP-TV and DeSCI.

We now apply our proposed Tensor ADMM-Net to real data captured by the SCI cameras [19, 34]. Since the real captured data have noise inside, the problem is more challenging. The exposure time of the camera is 33ms and the imaging systems capture a single compressed frame per 33ms (thus 30 fps). With this coded/compressed measurement, we can recover 14 or 22 frames high-speed videos. These real data of SCI system measurements are shown in Fig. 7 and the corresponding reconstructed videos are demonstrated in Figs. 8-10. For gray scale video of **Wheel**, Tensor ADMM-Net generates clear reconstruction with high efficiency and the degree of ghosting is reduced. For color video reconstruction, *i.e.*, **Ball** and **Hammer**, tensor ADMM-Net provides clearer and smoother reconstruction results while much noise exists in the reconstruction of GAP-TV.

In general, the reconstruction quality of our Tensor ADMM-Net is comparable with DeSCI, but the processing of our algorithm is much faster. Therefore, our algorithm is more applicable in real applications.

## 7. Conclusions

In this paper, we have proposed a deep tensor ADMM-Net for snapshot compressive imaging systems that provides high-quality decoding in seconds. We embedded the low-rank tensor model into the ADMM framework and unfolded the iterations into neural network stages, and thus our network enjoys potential mathematical interpretations. Experiments on simulation and real-world SCI camera data demonstrate that the proposed method exhibits superior performance and outperforms current state-of-the-art algorithms.

## References


[1] Yoann Altmann, Stephen McLaughlin, Miles J Padgett, Vivek K Goyal, Alfred O Hero, and Daniele Faccio. Quantum-inspired computational imaging. *Science*, 361(6403), 2018.

[2] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[3] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[4] Emmanuel Candes, Justin Romberg, and Terrence Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[5] Emmanuel Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

[6] Emmanuel Candes and Terrence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 12(52):5406–5425, 2006.

[7] David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[8] Michael Gehm, Renu John, David J Brady, Rebecca Willett, and Timothy Schulz. Single-shot compressive spectral imaging with a dual-disperser architecture. *Optics Express*, 15(21):14013–14027, 2007.

[9] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. Weighted nuclear norm minimization with application to image denoising. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2862–2869, 2014.

[10] John R Hershey, Jonathan Le Roux, and Felix Weninger. Deep unfolding: Model-based inspiration of novel deep architectures. *arXiv preprint arXiv:1409.2574*, 2014.

[11] Yasunobu Hitomi, Jinwei Gu, Mohit Gupta, Tomoo Mitsunaga, and Shree K Nayar. Video from a single coded exposure photograph using a learned over-complete dictionary. In *IEEE International Conference on Computer Vision (ICCV)*, pages 287–294, 2011.

[12] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[13] Shirin Jalali and Xin Yuan. Snapshot compressed sensing: performance bounds and algorithms. *IEEE Transactions on Information Theory*, 2019.

[14] Fei Jiang, Xiao-Yang Liu, Hongtao Lu, and Ruimin Shen. Efficient multi-dimensional tensor sparse coding using t-linear combination. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[15] Misha E Kilmer, Karen Braman, Ning Hao, and Randy C Hoover. Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging. *SIAM Journal on Matrix Analysis and Applications*, 34(1):148–172, 2013.

[16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014.

[17] Xiao-Yang Liu and Xiaodong Wang. Fourth-order tensors with multidimensional discrete transforms. *arXiv preprint arXiv:1705.01576*, 2017.

[18] Yang Liu, Xin Yuan, Jinli Suo, David Brady, and Qionghai Dai. Rank minimization for snapshot compressive imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[19] Patrick Llull, Xuejun Liao, Xin Yuan, Jianbo Yang, David Kittle, Lawrence Carin, Guillermo Sapiro, and David J Brady. Coded aperture compressive temporal imaging. *Optics Express*, 21(9):10526–10545, 2013.

[20] Matteo Maggioni, Giacomo Boracchi, Alessandro Foi, and Karen Egiazarian. Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms. *IEEE Transactions on Image Processing*, 21(9):3952–3966, 2012.

[21] Xin Miao, Xin Yuan, Yunchen Pu, and Vassilis Athitsos. $\lambda$-net: Reconstruct hyperspectral images from a snapshot measurement. In *IEEE Conference on Computer Vision (ICCV)*, 2019.

[22] Dikpal Reddy, Ashok Veeraraghavan, and Rama Chellappa. P2C2: Programmable pixel compressive camera for high speed imaging. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 329–336.

[23] Nirzhar Saha, Md Shareef Ifthekhar, Nam Tuan Le, and Yeong Min Jang. Survey on optical camera communications: challenges and opportunities. *IET Optoelectronics*, 9(5):172–183, 2015.

[24] Jian Sun, Huibin Li, Zongben Xu, et al. Deep admm-net for compressive sensing mri. In *Advances in Neural Information Processing Systems*, pages 10–18, 2016.

[25] Ashwin Wagadarikar, Renu John, Rebecca Willett, and David Brady. Single disperser design for coded aperture snapshot spectral imaging. *Applied Optics*, 47(10):B44–B51, 2008.

[26] Ashwin A Wagadarikar, Nikos P Pitsianis, Xiaobai Sun, and David J Brady. Video rate spectral imaging using a coded aperture snapshot spectral imager. *Optics Express*, 17(8):6368–6388, 2009.

[27] Lizhi Wang, Zhiwei Xiong, Dahua Gao, Guangming Shi, and Feng Wu. Dual-camera design for coded aperture snapshot spectral imaging. *Appl. Opt.*, 54(4):848–858, Feb 2015.

[28] Lizhi Wang, Zhiwei Xiong, Dahua Gao, Guangming Shi, Wenjun Zeng, and Feng Wu. High-speed hyperspectral video acquisition with a dual-camera architecture. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4942–4950, June 2015.

[29] Lizhi Wang, Zhiwei Xiong, Hua Huang, Guangming Shi, Feng Wu, and Wenjun Zeng. High-speed hyperspectral video acquisition by combining nyquist and compressive sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2018.

[30] Lizhi Wang, Zhiwei Xiong, Guangming Shi, Feng Wu, and Wenjun Zeng. Adaptive nonlocal sparse representation for dual-camera compressive hyperspectral imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(10):2104–2111, Oct 2017.

[31] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: From error visibility to

structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[32] Jianbo Yang, Xin Yuan, Xuejun Liao, Patrick Llull, David J Brady, Guillermo Sapiro, and Lawrence Carin. Video compressive sensing using gaussian mixture models. *IEEE Transactions on Image Processing*, 23(11):4863–4878, 2014.

[33] Xin Yuan. Generalized alternating projection based total variation minimization for compressive sensing. In *IEEE International Conference on Image Processing (ICIP)*, pages 2539–2543, 2016.

[34] Xin Yuan, Patrick Llull, Xuejun Liao, Jianbo Yang, David J. Brady, Guillermo Sapiro, and Lawrence Carin. Low-cost compressive sensing for color video and depth. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3318–3325, 2014.

[35] Jian Zhang and Bernard Ghanem. ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1828–1837, 2018.

[36] Zemin Zhang, Gregory Ely, Shuchin Aeron, Ning Hao, and Misha Kilmer. Novel methods for multilinear data completion and de-noising based on tensor-svd. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3842–3849, 2014.