Deep learning for video compressive sensing

Special Collection: Photonics and AI

Mu Qiao, Ziyi Meng, Jiawei Ma, Xin Yuan

APL Photonics 5, 030801 (2020)
https://doi.org/10.1063/1.5140721

Articles You May Be Interested In

High-resolution frequency-comb spectroscopy with electro-optic sampling and instantaneous octave-wide coverage across mid-IR to THz at a video rate
APL Photonics (November 2023)

Invited Article: Mask-modulated lensless imaging with multi-angle illuminations
APL Photonics (June 2018)

Invited Article: Digital refocusing in quantitative phase imaging for flowing red blood cells
APL Photonics (October 2018)
Deep learning for video compressive sensing

Cite as: APL Photon. 5, 030801 (2020); doi: 10.1063/1.5140721
Submitted: 30 November 2019 • Accepted: 18 February 2020 • Published Online: 6 March 2020

Mu Qiao,1 Ziyi Meng,1,a) Jiawei Ma,2 and Xin Yuan3,b)

AFFILIATIONS
1Department of ECE, New Jersey Institute of Technology, Newark, New Jersey 07102, USA
2Department of Computer Science, Columbia University, New York, New York 10027, USA
3Nokia Bell Labs, 600 Mountain Avenue, Murray Hill, New Jersey 07974, USA

Note: This article is part of the APL Photonics Special Topic on Photonics and AI.
a)Also at: State Key Laboratory of Information Photonics and Optical Communications, Beijing University of Posts and Telecommunications, Beijing 100876, China.
b)Author to whom correspondence should be addressed: xyuan@bell-labs.com

ABSTRACT

We investigate deep learning for video compressive sensing within the scope of snapshot compressive imaging (SCI). In video SCI, multiple high-speed frames are modulated by different coding patterns and then a low-speed detector captures the integration of these modulated frames. In this manner, each captured measurement frame incorporates the information of all the coded frames, and reconstruction algorithms are then employed to recover the high-speed video. In this paper, we build a video SCI system using a digital micromirror device and develop both an end-to-end convolutional neural network (E2E-CNN) and a Plug-and-Play (PnP) framework with deep denoising priors to solve the inverse problem. We compare them with the iterative baseline algorithm GAP-TV and the state-of-the-art DeSCI on real data. Given a determined setup, a well-trained E2E-CNN can provide video-rate high-quality reconstruction. The PnP deep denoising method can generate decent results without task-specific pre-training and is faster than conventional iterative algorithms. Considering speed, accuracy, and flexibility, the PnP deep denoising method may serve as a baseline in video SCI reconstruction. To conduct quantitative analysis on these reconstruction algorithms, we further perform a simulation comparison on synthetic data. We hope that this study contributes to the applications of SCI cameras in our daily life.

© 2020 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/). https://doi.org/10.1063/1.5140721

I. INTRODUCTION

Compressive sensing (CS)1–2 has inspired various compressive imaging systems that capture high-dimensional data, such as videos3–11 and hyperspectral images,12–17 in a snapshot fashion. For example, in video CS as shown in Fig. 1, the high-speed frames of a video are modulated at a higher speed than the capture rate of the camera. With knowledge of the modulation, multiple frames can be reconstructed from each single measurement. This type of technique is also termed snapshot compressive imaging (SCI).18 This paper focuses on the video CS, which is also known as the video SCI problem.

SCI systems were originally proposed to capture high-dimensional data using low-dimensional detectors and then employ iterative algorithms to solve such an ill-posed inversion problem, which suffers from long running time. On the one hand, this hardware encoder plus software decoder regime is capable of providing us with more information of the scene. On the other hand, the long running time of the algorithm18 precludes wide applications of SCI since in some cases, a real-time visualization is desired. Thanks to recent advances in deep learning, fast end-to-end reconstruction has been demonstrated in computational imaging (CI).19,20 The deep learning approach first learns an approximate inverse function of the system forward model in training and then provides instantaneous reconstruction by directly estimating outputs from the input measurements. While this approach enjoys the speed advantage, it usually requires a deep model, long training time, and a large amount of training data. Furthermore, it is less flexible than the iteration-based algorithms because the model is trained and then works on the system with determined hyper-parameters such as image size, compression ratio, and coding patterns. Most recently, pre-trained denoising networks have been integrated into the iteration-based algorithms, dubbed Plug-and-Play (PnP) framework,21 to improve the reconstruction speed and image quality. Although the speed of
The principle of video SCI is shown in Fig. 1. A dynamic scene, shown as a sequence of images at different timestamps \( \{t_1, t_2, \ldots, t_n \} \), passes through a dynamic aperture, which imposes individual coding patterns. The coded frames after the aperture are then integrated over time on a camera, forming a single-frame compressed measurement (top-right). In accordance with the measurement and the coding patterns, the iterative algorithms or pre-trained neural networks are used to reconstruct the time series (bottom-left) of the dynamic scene.

This paper aims to validate and compare these two deep learning-based (or related) regimes for video SCI reconstruction. For a fair comparison, we build a video SCI system and capture measurement data with different compression rates. Resting on these data, we develop both end-to-end neural networks and PnP algorithms (with deep denoising priors) to also compare their performance with traditional optimization baseline algorithms, namely, GAP-TV.\(^{18,22}\) We hope this comprehensive study will provide guidance to researchers and engineers to apply SCI cameras in our daily life.

### A. Snapshot video compressive sensing

The sensing process of video SCI is shown in Fig. 1. A dynamic scene, modeled as a time-series of two-dimensional (2D) images, passes through a dynamic aperture which applies timestamp-specified spatial coding. In specific, the value of each timestamp-specified spatial coding is superposed by a random pattern (binary patterns are used in this paper, \( \{0, 1\} \) with 0 denoting blocking the light and 1 meaning the light passing through) and the spatial coding of each two timestamps are different and independent from each other. The coded frames after the aperture are then integrated over time on a camera, forming a compressed coded measurement. Given the coding pattern for each frame, the time series of the scene can be reconstructed from the compressed measurement through iterative algorithms or pre-trained convolutional neural networks (CNNs).

### B. Deep learning for reconstruction

In general, the SCI problem we focus on is developed from the field of computational imaging (CI).\(^{29,30}\) Different from traditional imaging where the users acquire the desired signal directly, in CI, the captured measurement may not be visually interpretable but includes the signal in a carefully designed mechanism. As a result, the reconstruction algorithms are required to recover the signals from the measurement. For SCI problems, the well-established algorithms include TwIST,\(^{17}\) GAP-TV,\(^{24}\) and GMM\(^{25,26}\) based algorithms, where different priors are used. Most recently, the DeSCI algorithm\(^{11}\) has led to the state-of-the-art results of video SCI. DeSCI applies the weighted nuclear norm minimization (WNNM)\(^{10}\) of nonlocal similar patches in the video frames into the alternating direction method of multiplier (ADMM)\(^{10}\) regime.

Inspired by the recent advances of deep learning on image restoration,\(^{12,29}\) researchers have started using deep learning in CI.\(^{18,20,31–38}\) Deep fully connected neural network was used for video CS in Ref. 32, and most recently, a deep tensor ADMM-net was proposed in Ref. 36 for a video SCI problem. A joint optimization and reconstruction network was trained in Ref. 39 for video CS. The coding patterns used in Ref. 32 are a repeated pattern of a small block; this is not practical in real imaging systems, and only simulation results were shown therein. The deep tensor ADMM-net\(^{11}\) employs deep-unfolding technique,\(^{29,30}\) and limited results were shown based on the data in Refs. 5 and 6. In this paper, we build an end-to-end sensing and a reconstruction regime using deep learning.

It is anticipated that an end-to-end convolution neural network (E2E-CNN) can provide excellent results for CI given sufficient training data. Different from the conventional iteration-based algorithms,\(^{18}\) which must implement the iteration and computation for each measurement, the end-to-end CNN conducts optimization only on the training phase and recovers images on the inference phase in an efficient way. Intuitively, the end-to-end network enables a millisecond-level reconstruction for SCI problems. However, one drawback of E2E-CNN is the flexibility, in addition to the training data and time. Specifically, a new network has to be trained from scratch (or at least fine tuning using transfer learning) to perform the inference when the sensing matrices (or the coding patterns) change, which severely limits the application of adaptive sensing.\(^{22}\)

One promising solution to solve this trilemma, i.e., speed, accuracy, and flexibility, is to incorporate the deep learning with optimization-based algorithms. In particular, the deep denoising priors\(^{18}\) can be plugged into the ADMM\(^{24}\) (or other optimization) framework to construct the plug-and-play\(^{41}\) regime for SCI reconstruction. In this case, the denoising network can be pre-trained on regular images, and the repeat training for each individual task is not necessary.

### C. Contributions and organizations of this paper

Bearing the above concerns in mind, this paper makes the following contributions:

- Aiming at a fair comparison between different inversion algorithms on real data, we build a video SCI system using a digital micromirror device (DMD) as the high-speed spatial modulator,\(^{29}\) which provides up to 22 000 refreshing rates; different compression rates of 10, 20, 30, 40, and 50 are implemented, providing (reconstructed) video frame rates of 500, 1000, 1500, 2000, and 2500, respectively, for a fixed capturing (camera) frame rate of 50 fps (frames per second).
We train an end-to-end CNN by applying residual learning\(^7\) to the encoder–decoder structure\(^8\) for fast SCI reconstruction.

We develop a PnP-ADMM algorithm by integrating the FFDNet\(^9\) and the total variation (TV) denoising algorithms into the ADMM framework, dubbed PnP-TV-FFDNet.

We compare the results of E2E-CNN, PnP-TV-FFDNet with the state-of-the-art iterative optimization method DeSCI.\(^1\)\(^\text{10}\)

It is recommended that for a fixed setup, a well-trained E2E-CNN can provide high accuracy results within a short time. Without training data, PnP-TV-FFDNet can provide decent results in a limited running time. DeSCI can still provide visually excellent results but suffers from long running time. Therefore, a good trade-off of speed, accuracy, and flexibility is PnP-TV-FFDNet, which can be used as a new baseline in video SCI reconstruction.

The rest of this paper is organized as follows: Sec. II describes the optical setup of our video SCI system. Section III derives the mathematical model of video SCI and reviews the recent theoretical results. Section IV develops the E2E-CNN for SCI reconstruction, and quantitative simulation results are shown in Sec. VII. Section VIII concludes the entire paper.

II. HARDWARE SETUP OF VIDEO SCI
The optical setup of our system is depicted in Fig. 2. A commercial photographic lens (L1) images the object onto an intermediate plane, where a DMD (Vialux, DLP7000, 768 × 1024 elements) is employed to apply spatial modulation to the high-speed image sequences. A 4f system consisting of a tube lens L2 (f = 150 mm) and an objective lens (4×, f = 45 mm, NA = 0.1) relays the modulated image to a camera (Basler, acA1300-1024 frame rate of 50, whereas the DMD operates at various frame rates between 10 to 50 (for clarification, Cr = 10 means 10× compression and so on and so forth). The camera and the DMD are triggered by a data acquisition board (NI, USB6341). The modulation patterns used in the reconstruction are pre-calibrated by illuminating DMD with a uniform light and capturing the images of the DMD patterns with the camera.

In practice, the two controlled states of a DMD micromirror are oriented +12 and −12\(^\circ\) relative to the array panel. In order to steer the reflected light from the DMD to the camera, the BS is horizontally tilted by 12\(^\circ\) (not illustrated in Fig. 2). Besides, since the hinges of the micromirror are along the diagonal of the mirror, the mirrors rotate around an axis that is oriented 45\(^\circ\) to the array dimensions.

To facilitate the setup alignment, we make the hinge axis vertical by rotating the DMD by 45\(^\circ\) (within the panel plane). Accordingly, the camera is also rotated 45\(^\circ\) to align its sensor-array to the DMD array.

III. MATHEMATICAL MODEL OF VIDEO CS
Let (x, y, t) denote the transverse spatial and temporal dimensions of a dynamic scene f(x, y, t). Video SCI compresses the scene onto a two-dimensional (2D) monochrome camera with a limited frame rate, e.g., 50 fps used in our experiments. Let (x′, y′, t′) denote the measurement-space coordinates. The measurement formed on the detector plane can be represented as

\[
g(x', y', t') = \int_{-\Delta x}^{\Delta x} \int_{-\Delta y}^{\Delta y} f(x,y,t) T(x,y,t) \times p\left(\frac{x-x'}{\Delta x}, \frac{y-y'}{\Delta y}\right) dx dy dt, \tag{1}\]

where T(x, y, t) denotes the temporal modulation introduced by the DMD, variables \(\Delta x\) and \(\Delta y\) denote the (square) pixel pitch and the integration time of the camera, and \(N_x\), \(N_y\), and \(N_t\) denote the length in the spatial and temporal dimensions. The spatial and temporal pixel sampling functions are given by \(p\) and \(p_t\), respectively.

The pixel sampling (first on DMD and then on the camera) process discretizes the continuous projected signal. Such discretization can be represented by applying rectangular functions in all dimensions in (1). Consider that B high-speed frames \(\{X_k\}_{k=1}^B \in \mathbb{R}^{n_x \times n_y}\) are modulated by the coding patterns \(\{C_k\}_{k=1}^B \in \mathbb{R}^{n_x \times n_y}\), correspondingly (Fig. 1). The measurement \(Y\) is given by

\[
Y = \sum_{k=1}^B X_k \odot C_k + G, \tag{2}\]

where \(\odot\) denotes the Hadamard (element-wise) product and \(G\) represents the noise. For all B pixels (in the B frames) at position (i, j), \(i = 1, \ldots, n_x; j = 1, \ldots, n_y\), they are collapsed to form one pixel in the snapshot measurement as

\[
y_{ij} = \sum_{k=1}^B e_{i,k} x_{i,k} + g_{ij}. \tag{3}\]

Define

\[
x = \begin{bmatrix} x_1^T \ldots \ldots x_B^T \end{bmatrix}^T, \tag{4}\]

where \(x_k = \text{vec}(X_k)\), and let \(D_k = \text{diag}(\text{vec}(C_k))\) for \(k = 1, \ldots, B\), where vec() vectorizes the matrix inside () by stacking the columns.
and diag() transfers the ensured vector into a matrix by putting each element of the vector to the diagonal positions. We, thus, have the vector formulation of the sensing process,
\[ y = \Phi x + g, \]
(5)
where \( \Phi \in \mathbb{R}^{n \times m} \) is the sensing matrix with \( n = n_T n_B \), \( x \in \mathbb{R}^m \) is the desired signal, and \( g \in \mathbb{R}^n \) again denotes the noise.

Unlike traditional CS, the sensing matrix considered here is not a dense matrix. In SCI, the matrix \( \Phi \) in (5) has a very special structure and can be written as
\[ \Phi = [D_1, \ldots, D_B], \]
(6)
where \( \{D_i\}_{i=1}^B \) are diagonal matrices. Therefore, the compressive sampling rate in SCI is equal to \( 1/B \).

It is worth noting that due to the special structure of \( \Phi \) in (6), we have \( \Phi \Phi^T \) being a diagonal matrix. This fact will be useful to derive the efficient algorithm in Sec. V for handling the massive data in SCI.

One natural question is whether it is theoretically possible to recover \( x \) from the measurement \( y \) defined in Eq. (5) for \( B > 1 \).

Most recently, by using the compression-based compressive sensing regime, this has been addressed in Refs. 47 and 48 via the following theorem, where \( \{f, g\} \) denotes the encoder/decoder, respectively.

**Theorem 1** (Ref. 48). Assume that \( \forall x \in \mathcal{D}, \|x\|_\infty \leq \xi \). Further assume that the rate-1 code achieves distortion \( \delta \) on \( \mathcal{D} \). Moreover, for \( i = 1, \ldots, B \), \( D_i = \text{diag}(D_{i1}, \ldots, D_{in}) \) and \( \{D_{ik}\}_{k=1}^n \overset{i.i.d.}{\sim} \mathcal{N}(0,1) \). For \( x \in \mathcal{D} \) and \( y = \sum_{i=1}^B D_i x \), let \( \hat{x} \) denote the solution of compressible signal pursuit optimization. Assume that \( \varepsilon > 0 \) is a free parameter, such that \( \varepsilon \leq \frac{\delta}{2} \). Then,
\[ \frac{1}{nB} \|x - \hat{x}\|_2^2 \leq \delta + \rho^2 \varepsilon, \]
(7)
with a probability larger than \( 1 - 2^{nB+1} e^{-n(\frac{\rho^2}{2})^2} \).

Details of the optimization and proof can be found in Ref. 48. Most importantly, Theorem 1 characterizes the performance of SCI recovery by connecting the parameters of the (compression/decompression) code, its rate \( r \), and its distortion \( \delta \) to the number of frames \( B \) and the reconstruction quality. This theoretical finding strongly encourages our algorithmic design using both deep learning and optimizations for SCI systems.

Recall the forward model of SCI in (5), where we can see that it is an ill-posed problem. To solve this problem, as shown in Fig. 3(a), one common way is to employ a prior, e.g., TV or sparsity. More complicated priors, such as low rankness of similar patch groups, have also been utilized. While these iterative algorithms are usually time consuming, motivated by recent advances in deep learning, the E2E-CNNs have been used for video SCI, shown in Fig. 3(c). However, as mentioned before, these E2E-CNNs usually need a significant amount of training data and time. When the sensing matrix, \( \Phi \), changes, another network has to be re-trained. This again needs additional time and training data. To mitigate this, PnP frameworks have been proposed to use pre-trained deep denoising priors into the optimization framework [Fig. 3(b)] as a marriage of both to seek a trade-off of time and flexibility.

In the following, we first build an E2E-CNN for SCI reconstruction and then derive the PnP framework by considering the hardware constrains of SCI. Performances of these methods will be compared thoroughly with extensive real data captured by our video SCI system. For a quantitative comparison, we also conduct simulations on synthetic data to compare these algorithms.

**IV. END-TO-END CNNS FOR VIDEO CS RECONSTRUCTION**

In this section, we build an end-to-end CNN to perform the SCI reconstruction.
A. Network structure

As illustrated in Fig. 4, E2E-CNN is based on a convolutional encoder–decoder architecture with residual connection.44 We impose five residual blocks in both encoder and decoder parts, which are connected by two convolution layers. The input first passes one convolution layer for multi-dimensional feature extraction. Each convolution operation is followed by ReLU activation and batch normalization. Besides, the output of one encoder residual block is added to the input of the mapped decoder residual block. Note that we use summation \( \oplus \) instead of concatenation according to experimental results. Furthermore, we use long-span residual connection to synthesize the network input into the final reconstruction. We employ tanh as the activation function of the network to ensure a desired scale of the final output. Note that we use neither pooling nor upsampling in our network to avoid losing image details.59

To reduce the burden in learning the forward operator of the imaging system, we use \( \Phi \) as an approximate inverse operator to initialize the network input as \( \Phi (\Phi y)^{-1} y \), which has the same dimension and scale with \( x \). Such setting is considered to be reliable for CI problems43 and has been widely used in recent works.2,53

B. Datasets and implementation

For model training, we capture 30 high-speed motions of daily life objects, e.g., boxes, letters, toys, etc., using a conventional high-speed camera. By scaling, cropping, and rotating, we generate 2400 life objects, e.g., boxes, letters, toys, etc., using a conventional high-speed camera. By scaling, cropping, and rotating, we generate 2400 frames image of 512 × 512 pixels. The corresponding measurements \( y \) of the truth are generated by simulating the system’s forward model using the calibrated coding patterns \( \Phi \).

For each training pair \( \{ (\Phi y)^{-1} y, x \} \), we train the model by feeding \( \Phi (\Phi y)^{-1} y \) into the network and using the Adam optimizer to minimize the objective function—root mean square error (RMSE) and multiscale structural similarity index (MS-SSIM) between the network output and the truth \( x \). Specifically, the loss function of our model is

\[
\mathcal{L}_{CNN} = \alpha \| x - \hat{x} \|_2^2 + \beta [1 - \text{MS-SSIM}(x, \hat{x})],
\]

where \( \hat{x} \) denotes the output of the network. A detailed description on this loss function can be found in Ref. 55. The parameters \( \alpha \) and \( \beta \) are set to 1 and 0.1 in our model training.

To reduce the gap between the simulated and real data, shot noise in the real system is simulated by Poisson sampling the synthesized measurements. Specifically, for each epoch, we use measurements of the same scene sets but with different shot noise realization to improve the robustness of the network. The training is implemented in TensorFlow on a NVIDIA GeForce GTX 1080Ti graphics processing unit (GPU). The learning rate is initially set to 0.01, scaled by 0.8 after each 30 epochs.

V. Plug-and-Play Algorithms for Video SCI Reconstruction

As mentioned before, the E2E-CNNs lack the flexibility for different tasks, e.g., different compression rates in this work. In the following, we develop the PnP algorithms for video SCI.

We aim to recover \( x \) based on the captured measurement \( y \) and coding patterns \( \Phi \). Similar with previous research, priors (here including both the conventional and deep priors) are usually employed to solve this problem. Let \( R(x) \) denote the prior to be used, the reconstruction target is formulated as

\[
\hat{x} = \arg \min_x \frac{1}{2} \| y - \Phi x \|_2^2 + \tau R(x),
\]

where \( \tau \) is a parameter to balance the fidelity term (the \( \ell_2 \)-norm) and the prior.

Hereby, as in Ref. 21, we employ the ADMM framework to solve the problem, and by introducing the variable \( z \), Eq. (9) is now formulated as

\[
\hat{x} = \arg \min_x \frac{1}{2} \| y - \Phi x \|_2^2 + \tau R(z), \quad \text{s.t.} \quad x = z.
\]

We use the superscript \( t \) to index the iteration number and introduce an auxiliary variable \( u \). Equation (10) is solved by iteratively solving the following three sub-problems:
By utilizing the special structure of $\Phi$ and denoising algorithms, we propose to employ various denoising algorithms. Conventional denoising algorithms, such as PnP-ADMM, can be updated in one shot, efficiently via element-wise operation rather than calculating the gradient. For instance, the gradient of $\Phi$ can be calculated as
\[
\nabla \Phi(x) = \Phi^T \Phi x - \Phi^T y.
\]
where $\Phi$ is a block diagonal matrix of size $n \times nB$ as in Eq. (6).

A. Utilizing deep denoising priors

In order to embrace deep learning, deep denoising priors are employed to solve Eq. (12) without re-training the model. This set-asides time and training data and enables the algorithm's flexibility. In our work, after trying several deep denoisers, we found that FFDNet was used as the denoiser.

In our experiments, we observed that solely using FFDNet leads to undesired artifacts (especially at large compression rates), while a TV prior alone usually leads to noisy reconstructions. One reason is that FFDNet was trained by Gaussian noise, while in our video SCI, in each iteration, the noise distribution is different. Motivated by this, we propose to use a joint denoising strategy as follows:
\[
\begin{align*}
    z_t^{\text{J}} &= \text{FFDNet}(x_t^{\text{J}} - u_t, \sigma_t), \\
    z_h^{\text{J}} &= \text{TV}(x_t^{\text{J}} - u_t, \sigma_t),
\end{align*}
\]
where $\sigma_t$ is the estimated Gaussian noise level in each iteration. The final $z^{\text{J}}$ in each iteration is achieved by
\[
z^{\text{J}} = \alpha z_h^{\text{J}} + (1 - \alpha) z_t^{\text{J}},
\]
where $0 \leq \alpha \leq 1$ is a weighted parameter. This can be recognized as a complement prior strategy. We further observed that $\alpha$ should be decreasing with the iteration, i.e., at the beginning, mostly the TV denoiser has a larger weight, while the FFDNet denoiser starts getting more relevance as the reconstruction going on. We term this algorithm PnP-TV-FFDNet.

B. Convergence analysis

PnP-ADMM has been proved to converge to a fixed point under the conditions of a bounded denoiser and a bounded gradient of the loss function. In our case, the loss function is
\[
\mathcal{L}(x) = \frac{1}{2} \| y - \Phi x \|_2^2.
\]
In this paper, we assume that the denoisers are all bounded. Taking the hardware constraints of video SCI into consideration, we prove that the gradient of $\mathcal{L}(x)$ is bounded. The gradient of $\mathcal{L}(x)$ in SCI is
\[
\nabla \mathcal{L}(x) = \Phi^T \Phi x - \Phi^T y,
\]
where $\Phi$ is a block diagonal matrix of size $n \times nB$ as in Eq. (6).

VI. EXPERIMENTAL RESULTS

In this section, we present extensive experimental results to verify our hardware setup and different algorithms.

A. Hardware and algorithms verification

We validate our system through two dynamic scenes: moving letters (“labs”) and falling dominoes. Various compression ratios from 10 to 50 are adopted to achieve different image frame rates from 500 to 2500 with a fixed measurement (camera) frame rate of 50.
FIG. 5. Experimental results of moving letters—“labs.” The far-left column shows the measurements under different compression ratios. The upper part shows the reconstruction from the E2E-CNN models, trained for Cr = 10, 20, and 30 separately. The lower part shows the reconstruction from the PnP-TV-FFDNet algorithm for Cr = 40 and 50. Grids are added to help visualize the motion.
FIG. 6. Experimental results of falling dominoes. The far-left column shows the measurements under different compression ratios. The upper part shows the reconstruction from the E2E-CNN models, trained for Cr = 10, 20, and 30 separately. The lower part shows the reconstruction from the PnP-TV-FFDNet algorithm for Cr = 40 and 50. Grids are added to help visualize the motion.
FIG. 7. Reconstruction of falling dominoes with different algorithms. The first row shows the compressed measurements with different compression ratios from 10 to 50. Five methods are used to reconstruct the video from the measurements. Two representative excerpts are shown for each measurement (selected frame numbers are indicated in the second row). Due to limited GPU memory, for the E2E-CNN method, we only show the results for Cr = 10, 20, and 30 obtained from three separately trained networks. Image size is 512 × 512 pixels. Multimedia view: https://doi.org/10.1063/1.5140721.1

FIG. 8. Reconstruction of water balloon falling and bouncing with different algorithms. The first row shows the compressed measurements with different compression ratios from 10 to 50. Five methods are used to reconstruct the video from the measurement. Two representative excerpts are shown for each measurement (selected frame numbers are indicated in the second row). Similar to Fig. 7 (multimedia view), we only show the results for Cr = 10, 20, and 30 obtained from E2E-CNN. Image size is 512 × 512 pixels. Multimedia view: https://doi.org/10.1063/1.5140721.2
The measurements are shown on the left columns in Figs. 5 and 6. Due to limited GPU memory, we only trained the E2E-CNN models for $Cr \leq 30$ (each with a different network). The training and reconstruction take 27 h and 70 ms for $Cr = 10$, 36 h and 90 ms for $Cr = 20$, and 43 h and 120 ms for $Cr = 30$, respectively. Although the training sample numbers are the same, a larger $Cr$ needs a larger memory since the data size is getting larger and, thus, a longer training time. The results of E2E-CNN are shown in the upper parts in Figs. 5 and 6, from which we observe clean and clear letters or domino boxes, in contrast to the noisy and blurred measurements, with different locations and/or shapes at different times (grids are added to help visualize the motion). We also observe that higher $Cr$ does provide higher temporal resolutions paying little price at the spatial resolution (or image quality).

For $Cr \geq 40$, we use PnP-TV-FFDNet to perform the reconstruction, which takes 70 s and 88 s, respectively. From the results, we observe that higher $Cr$ provides higher temporal resolution which helps visualize finer movements, but the image quality would degrade, become less uniform in intensity and less sharp in edges, as $Cr$ increases.

### B. Results with different algorithms

Through three dynamic scenes and under five different compression ratios, we now compare the performance of E2E-CNN and PnP-TV-FFDNet with another three iterative algorithms: GAP-TV, GAP-TV + deep denoising (GAP-TV + DD), and DeSCI (PnP-WNNM). The first scene is the falling dominoes which is a relatively slow scene [Fig. 7 (Multimedia view)]; the second one is the fast motion of water balloon falling and bouncing [Fig. 8 (Multimedia view)]; the third one is pendulum balls [Fig. 9 (Multimedia view)], which is fast yet has more detail features. Table I lists the computation time of different methods based on a desktop of 16 CPU cores @ 3.2 GHz and an NVIDIA GTX 1080Ti GPU. For the iterative methods, the time is based on 50 iterations.

In general, Figs. 7–9 (Multimedia view) and Table I reveal a trade-off between reconstruction speed, image quality, and flexibility. The E2E-CNN method provides the fastest reconstruction speed (tens of milliseconds) but is less flexible than the iterative algorithms; the pre-trained network works only with specific mask

| $Cr$ | 10 | 20 | 30 | 40 | 50 |
|------|----|----|----|----|----|
| GAP-TV (s) | 50 | 80 | 120 | 150 | 180 |
| GAP-TV + DD (s) | +0.2 | +0.5 | +1.0 | +1.2 |
| DeSCI (h) | 0.8 | 1.2 | 1.7 | 2.5 | 3.5 |
| PnP-TV-FFDNet (s) | 18 | 36 | 52 | 70 | 88 |
| E2E-CNN | 0.1 s | 0.16 s | 0.2 s | Out of memory | Out of memory |

(20 h) (25 h) (28 h) (training) (20 h) (25 h) (28 h)
TABLE II. Performance comparison of GAP-TV, DeSCI, PnP-TV-FFDNet, and E2E-CNN on five simulation scenes using PSNR (in dB, left in each cell) and SSIM (normalized to 1, right in each cell) as the metrics. The boldface marks the maximum values (best results) in each column.

| Method       | Scene       | Kobe       | Traffic     | Runner      | Aerial      | Crash       | Average     |
|--------------|-------------|------------|-------------|-------------|-------------|-------------|-------------|
|              |             | 26.46, 0.8848 | 20.89, 0.7148 | 28.52, 0.9092 | 25.05, 0.8281 | 24.82, 0.8383 | 25.15, 0.8350 |
| GAP-TV       |             | 33.25, 0.9518 | 28.71, 0.9205 | 38.48, 0.9693 | 25.33, 0.8603 | 27.04, 0.9094 | 30.56, 0.9223 |
| DeSCI        |             | 30.50, 0.9256 | 24.18, 0.8279 | 32.15, 0.9332 | 25.27, 0.8291 | 25.42, 0.8493 | 27.50, 0.8730 |
| PnP-TV-FFDNet|             | 29.02, 0.8612 | 23.45, 0.8375 | 34.43, 0.9580 | 27.52, 0.8822 | 26.40, 0.8858 | 28.16, 0.8850 |
| E2E-CNN      |             |            |             |             |             |             |             |

FIG. 10. Simulation results of GAP-TV, DeSCI, PnP-TV-FFDNet, and E2E-CNN on five scenes.
patterns, image size, and compression ratio. Although a generalized (also larger and deeper) network can be trained to adapt for various system parameters, this would require a huge GPU memory and is, thus, not practical for general users. The DeSCI provides state-of-the-art image qualities [e.g., comparing the water balloon and the letters “LAB” in Fig. 8 (Multimedia view) in the dashed yellow rectangles], but hours of computation is required. GAP-TV consumes much less computation time (1–3 min), but the results would appear noisy. However, we find that by simply applying a fast (≤1.2 s) deep denoising (FFDNet) process after the GAP-TV iterations, the noise can be significantly reduced. Among these iterative methods, the PnP-TV-FFDNet is the fastest one (0.2–1.5 min) because the denoising process using a deep denoising network is fast. The image quality is similar to DeSCI under Cr ≤ 20. For Cr ≥ 30, artificial distortions [see the regions indicated by dashed red rectangles in Figs. 7–9 (Multimedia view)] start to degrade the image. To reduce the artifacts, the noise estimation parameter in the algorithm can be set to high values, which functions by ignoring more detailed features.

Note that the results of E2E-CNN in Figs. 7–9 (Multimedia view) are inferred by the same trained networks (3 networks for three compression ratios) with the same training dataset. In most cases, E2E-CNN provides the best result on static scenes, regardless of the compression ratio. However, for fast motion scenes with a high compression ratio, E2E-CNN is not good enough, e.g., the top part of the balloon in Fig. 8 (Multimedia view). One reason is that the motion dynamics of the balloon is faster than other two scenes. More training datasets may help the E2E-CNN, or a deeper network is required. This again pops out the concern of using E2E-CNNs for video SCI, i.e., lacking flexibility. By contrast, other algorithms are more robust to different scenes and motions.

VII. SIMULATION RESULTS

Aiming at a quantitative comparison, in this section, we conduct simulations on five motion scenes—Kobe, Runner, Traffic, Aerial, and Crash—used in Refs. 18 and 36. In the simulation, every eight (Cr = 8) video frames with a size of 256 × 256 pixels were modulated by eight different binary masks and then collapsed to a single measurement. For the E2E-CNN model, we generated 25 000 training pairs using the DAVIS5 dataset. Considering the small range of motion in simulation data, we make a minor adjustment on the network and training, i.e., removing the long-span residual connection between the input and output and increasing the weight of MS-SSIM loss to 0.7.

We compared the proposed E2E-CNN and PnP-TV-FFDNet method with GAP-TV and DeSCI by evaluating their peak-signal-to-noise-ratio (PSNR) and the structured similarity index (SSIM)7 in Table II. We can observe that DeSCI performs best on almost all scenes except the Aerial. E2E-CNN has a 0.66 dB higher PSNR in average than PnP-TV-FFDNet. Similar to the real data, DeSCI still provides best results on average, gaining >2.5 dB in PSNR over deep learning methods, again by the price of long running time.

Figure 10 visualizes some representative frames, from which we can see that E2E-CNN performs better on static details (e.g., buildings in Aerial), while PnP-TV-FFDNet recovers clearer motions (e.g., feet in Runner) than E2E-CNN.

In summary, we obtain similar conclusions to the experimental data from these simulations. This further validates our end-to-end study.

VIII. CONCLUSIONS

We have built a video snapshot compressive imaging system using a digital micromirror device as the dynamic modulator and developed an end-to-end convolutional neural network and a plug-and-play algorithm that jointly used TV and FFDNet denoising priors for reconstruction. Various plug-and-play algorithms using different priors have been compared with the developed methods under different dynamic scenes and compression ratios.

Given a pre-trained network for a specific system, the E2E-CNN method provides the fastest reconstruction speed and is able to achieve real-time reconstruction by employing a faster GPU available on the market. The PnP-TV-FFDNet algorithm is the fastest one among the iteration-based algorithms due to the use of the fast deep denoising priors and is able to provide similar image quality to DeSCI (state-of-the-art in terms of image quality) at low compression ratios (≤20). Considering speed, quality, flexibility, cost, and ease-of-use, we believe that the PnP-TV-FFDNet method is a preferable baseline for video SCI reconstruction although it starts to suffer from artifacts when Cr ≥ 30. The artifacts might be reduced in future by employing more relevant deep denoising priors. For a fixed system, the E2E-CNN method, however, would be more preferable since it provides better image quality and much faster speed.

Due to the high capturing speed and the near real-time reconstruction (by the E2E-CNN method), we expect our end-to-end SCI system to find applications in traffic surveillance, sports photography, and autonomous vehicles. Regarding future work, we expect to exploit the correlation between video frames to train a video-wise deep denoising prior, rather than the frame-wise prior presented in this paper, for the proposed PnP framework to improve the reconstruction quality. For the E2E-CNN method, a flexible network compatible with varying image size and compression ratio is expected.

AUTHOR’S CONTRIBUTIONS

M.Q. and Z.M. contributed equally to this work.

REFERENCES

1. D. L. Donoho, “Compressed sensing,” IEEE Trans. Inf. Theory 52, 1289–1306 (2006).
2. C. Emmanuel, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” IEEE Trans. Inf. Theory 52, 489–509 (2006).
3. Y. Hitomi, J. Gu, M. Gupta, T. Mitsunaga, and S. K. Nayar, “Video from a single coded exposure photograph using a learned over-complete dictionary,” in 2011 International Conference on Computer Vision (IEEE, 2011), pp. 287–294.
4. D. Reddy, A. Veeraraghavan, and R. Chellappa, “P2c2: Programmable pixel compressive camera for high speed imaging,” in CVPR 2011 (IEEE, 2011), pp. 329–336.
5. P. Llull, X. Liao, X. Yuan, J. Yang, D. Kittle, L. Carin, G. Sapito, and D. J. Brady, “Coded aperture compressive temporal imaging,” Opt. Express 21, 10526–10545 (2013).
Y. Xu, P. Llull, X. Liao, J. Yang, D. J. Brady, G. Sapiro, and L. Carin, “Low-cost compressive sensing for color video and depth,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (IEEE), 2014, pp. 3318–3325.

Y. Sun, X. Yuan, and S. Pang, “High-speed compressive range imaging based on active illumination,” Opt. Express 24, 22836–22846 (2016).

Y. Sun, X. Yuan, and S. Pang, “Compressive high-speed stereo imaging,” Opt. Express 25, 18182–18190 (2017).

X. Yuan and S. Pang, “Structured illumination temporal compressive microscopy,” Biomed. Opt. Express 7, 746–758 (2016).

X. Yuan, Y. Sun, and S. Pang, “Compressive video sensing with side information,” Appl. Opt. 56, 2697–2704 (2017).

X. Yuan and S. Pang, “Compressive video microscopy via structured illumination,” in 2016 IEEE International Conference on Image Processing (ICIP) (IEEE), 2016, pp. 1589–1593.

X. Cao, T. Yue, X. Lin, S. Lin, X. Yuan, Q. Dai, L. Carin, and D. J. Brady, “Single-shot compressive spectral imaging with a multi-disperser architecture,” Opt. Express 15, 14013–14027 (2007).

A. Wagadarikar, R. John, R. Willett, and D. Brady, “Single disperser design for coded aperture snapshot spectral imaging,” Appl. Opt. 47, B44–B51 (2008).

A. A. Wagadarikar, N. P. Pitsianis, X. Sun, and D. J. Brady, “Video rate spectral imaging using a coded aperture snapshot spectral imager,” Opt. Express 17, 6368–6388 (2009).

X. Yuan, T.-H. Tsai, R. Zhu, P. Llull, D. Brady, and L. Carin, “Compressive hyperspectral imaging with side information,” IEEE J. Sel. Top. Signal Process. 9, 964–976 (2015).

X. Cao, T. Yue, X. Lin, X. Yuan, Q. Dai, L. Carin, and D. J. Brady, “Computational snapshot multispectral cameras: Toward dynamic capture of the spectral world,” IEEE Signal Process. Mag. 33, 95–108 (2016).

T.-H. Tsai, P. Llull, X. Yuan, L. Carin, and D. J. Brady, “Spectral-temporal compressive imaging,” Opt. Lett. 40, 4054–4057 (2015).

Y. Liu, X. Yuan, J. Suo, D. Brady, and Q. Dai, “Rank minimization for snapshot compressive imaging,” IEEE Trans. Pattern Anal. Mach. Intell. 41, 2990–3006 (2019).

X. Yuan and Y. Pu, “Parallel lensless compressive imaging via deep convolutional neural networks,” Opt. Express 26, 1962–1977 (2018).

X. Miao, X. Yuan, Y. Pu, and V. Athitsos, “J-net: Reconstruct hyperspectral images from a snapshot measurement,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (IEEE), 2019.

S. H. Chan, X. Wang, and O. A. Elgendy, “Plug-and-play ADMM for image restoration: Fixed-point convergence and applications,” IEEE Trans. Comput. Imaging 3, 84–98 (2017).

X. Yuan, “Generalized alternating projection based total variation minimization for compressive sensing,” in 2016 IEEE International Conference on Image Processing (ICIP) (IEEE), 2016, pp. 2539–2543.

V. Altmann, S. McLaughlin, M. J. Padgett, V. K. Goyal, A. O. Hero, and D. Faccio, “Quantum-inspired computational imaging,” Science 361, eaat2298 (2018).

J. Bioucas-Dias and M. Figueiredo, “A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image restoration,” IEEE Trans. Image Process. 16, 2992–3004 (2007).

J. Yang, X. Yuan, X. Liao, P. Llull, G. Sapiro, D. J. Brady, and L. Carin, “Video compressive sensing using Gaussian mixture models,” IEEE Trans. Image Process. 23, 4863–4878 (2014).

J. Yang, X. Liao, X. Yuan, P. Llull, D. J. Brady, G. Sapiro, and L. Carin, “Compressive sensing by learning a Gaussian mixture model from measurements,” IEEE Trans. Image Process. 24, 106–119 (2015).

S. Gu, L. Zhang, W. Zuo, and X. Feng, “Weighted nuclear norm minimization with application to image denoising,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (IEEE), 2014, pp. 2862–2869.

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. 3, 1–122 (2011).

X. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep neural networks,” in Advances in Neural Information Processing Systems 25, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc., 2012), pp. 341–349.

K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising,” IEEE Trans. Image Process. 26, 3142–3155 (2017).

J.-H. R. Chang, C.-I. Li, B. Poczos, B. V. Kumar, and A. C. Sankaranarayanan, “One network to solve them all—Solving linear inverse problems using deep projection models,” in 2017 IEEE International Conference on Computer Vision (ICCV) (IEEE), 2017, pp. 5889–5898.

M. Illidis, L. Spoinoulas, and A. K. Katsaggelos, “Deep fully-connected networks for video compressive sensing,” Digital Signal Process. 72, 9–18 (2018).

K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, “Deep convolutional neural network for inverse problems in imaging,” IEEE Trans. Image Process. 26, 4509–4522 (2017).

K. Kulkarni, S. Lohit, P. Turaga, R. Kerviche, and A. Ashok, “Reconnect: Non-iterative reconstruction of images from compressively sensed random measurements,” in CVPR, 2016.

A. Mousavi and R. G. Baraniuk, “Learning to invert: Signal recovery via deep convolutional networks,” in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (IEEE), 2017, pp. 2272–2276.

X. Yuan, J. Suo, D. Brady, and Q. Dai, “Deep tensor admm-net for snapshot compressive imaging,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (IEEE), 2019.

A. Sinha, J. Lee, S. Li, and G. Barbastathis, “Lensless computational imaging through deep learning,” Optica 4, 1117–1125 (2017).

K. Xu and F. Ren, “CSVideoNet: A real-time end-to-end learning framework for high-frame-rate video compressive sensing,” in 2018 IEEE Winter Conference on Applications of Computer Vision (WACV) (IEEE), 2018, pp. 1680–1688.

M. Yoshida, A. Torii, M. Okutomi, K. Endo, Y. Sugiyama, R. i-Taniuchi, and H. Nagahara, “Joint optimization for compressive video sensing and reconstruction under hardware constraints,” in The European Conference on Computer Vision (ECCV), 2018.

R. Hershey, J. L. Roux, and F. Weninger, “Deep unfolding: Model-based inspiration of novel deep architectures,” preprint arXiv:1409.2574 (2014).

J. Yang, J. Sun, H. Li, and Z. Xu, “Deep admm-net for compressive sensing mri,” in Advances in Neural Information Processing Systems 29, edited by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Curran Associates, Inc., 2016), pp. 10–18.

X. Yuan, J. Yang, P. Llull, X. Liao, G. Sapiro, D. J. Brady, and L. Carin, “Adaptive temporal compressive sensing for video,” in 2013 IEEE International Conference on Image Processing (IEEE), 2013, pp. 14–18.

K. Zhang, W. Zuo, and L. Zhang, “FFDNet: Toward a fast and flexible solution for CNN-based image denoising,” IEEE Trans. Image Process. 27, 4608–4622 (2018).

K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (IEEE), 2016, pp. 770–778.

O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in Medical Imaging Computing and Computer-Assisted Intervention (MICCAI), LNCS Vol. 9351 (Springer, 2015), pp. 234–241; arXiv:1505.04597 (cs.CV).

S. Jalali and A. Maleki, “From compression to compressed sensing,” Appl. Comput. Harmonic Anal. 40, 352–385 (2016).

S. Jalali and X. Yuan, “Compressive imaging via one-shot measurements,” in IEEE International Symposium on Information Theory (ISIT), 2018.

S. Jalali and X. Yuan, “Snapshot compressed sensing: Performance bounds and algorithms,” IEEE Trans. Inf. Theory 65, 8005–8024 (2019).

X. Yuan, V. Rao, S. Han, and L. Carin, “Hierarchical infinite divisibility for multiscale shrinkage,” IEEE Trans. Signal Process. 62, 4363–4374 (2014).

X. Mao, C. Shen, and Y.-B. Yang, “Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections,” in Advances in Neural Information Processing Systems (Neural Information Processing Systems Foundation, Inc., 2016), pp. 2802–2810.

G. Barbastathis, A. Ozcan, and G. Sivils, “On the use of deep learning for computational imaging,” Optica 6, 921–943 (2019).
A. Goy, K. Arthur, S. Li, and G. Barbastathis, “Low photon count phase retrieval using deep learning,” Phys. Rev. Lett. 121, 243902 (2018).

M. Lyu, W. Wang, H. Wang, H. Wang, G. Li, N. Chen, and G. Situ, “Deep-learning-based ghost imaging,” Sci. Rep. 7, 17865 (2017).

D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” preprint arXiv:1412.6980 (2014).

Z. Wang, E. P. Simoncelli, and A. C. Bovik, “Multiscale structural similarity for image quality assessment,” in The Thirty-Seventh Asilomar Conference on Signals, Systems and Computers, 2003 (IEEE, 2003), Vol. 2, pp. 1398–1402.

J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool, “The 2017 David challenge on video object segmentation,” preprint arXiv:1704.09675 (2017).

Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” IEEE Trans. Image Process. 13, 600–612 (2004).