This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

Plug-and-Play Algorithms for Video Snapshot Compressive Imaging

Xin Yuan, *Senior Member, IEEE,* Yang Liu, Jinli Suo, Frédo Durand and Qionghai Dai, *Senior Member, IEEE*

Abstract—We consider the reconstruction problem of video snapshot compressive imaging (SCI), which captures high-speed videos using a low-speed 2D sensor (detector). The underlying principle of SCI is to modulate sequential high-speed frames with different masks and then these encoded frames are integrated into a snapshot on the sensor and thus the sensor can be of low-speed. On one hand, video SCI enjoys the advantages of low-bandwidth, low-power and low-cost. On the other hand, applying SCI to large-scale problems (HD or UHD videos) in our daily life is still challenging and one of the bottlenecks lies in the reconstruction algorithm. Existing algorithms are either too slow (iterative optimization algorithms) or not flexible to the encoding process (deep learning based end-to-end networks). In this paper, we develop fast and flexible algorithms for SCI based on the plug-and-play (PnP) framework. In addition to the PnP-ADMM method, we further propose the PnP-GAP (generalized alternating projection) algorithm with a lower computational workload. We first employ the *image* deep denoising priors to show that PnP can recover a UHD color video with 30 frames from a snapshot measurement. Since videos have strong temporal correlation, by employing the *video* deep denoising priors, we achieve a significant improvement in the results. Furthermore, we extend the proposed PnP algorithms to the color SCI system using mosaic sensors, where each pixel only captures the red, green or blue channels. A joint reconstruction and demosaicing paradigm is developed for flexible and high quality reconstruction of color video SCI systems. Extensive results on both simulation and real datasets verify the superiority of our proposed algorithm.

Index Terms—Compressive sensing, deep learning, computational imaging, coded aperture, image processing, video processing, coded aperture compressive temporal imaging (CACTI), plug-and-play (PnP) algorithms.

1 Introduction

Conventional imaging systems are developed to *capture* more data, such as high-resolutions and large field-of-view. However, to save these captured data, image/video compression methods are immediately applied due to the limited memory and bandwidth. This "capturing images first and processing afterwards" cannot meet the unprecedented demand in recent explosive growth of artificial intelligence and robotics. To address these challenges, computational imaging [1], [2] constructively combines optics, electronics and algorithms for optimized performance [3] or to provide new abilities [4], [5] to imaging systems. Different from conventional imaging, these computational imaging systems usually capture the data in an indirect manner, mostly compressed or coded.

This paper considers one important branch of computational imaging with promising applications, the snapshot compressive imaging (SCI) [6], which utilizes a two-dimensional (2D) camera to capture the 3D video or spectral data in a snapshot. Such imaging systems adopt *compressed sampling* on a set of consecutive images–video frames (*i.e.*, CACTI [7], [8]) or spectral channels (*i.e.*, CASSI [9])–in accordance with an encoding procedure and *integrating* these sampled signals along time or spectrum to obtain the final compressed measurements. With this technique, SCI systems can capture the high-speed motion [10], [11], [12], [13] and high-resolution spectral information [14], [15] but with low memory, low bandwidth, low power and potentially low cost. In this work, we focus on video SCI.

There are two critical challenges in SCI and other computational imaging systems. The first one is the hardware imaging system to capture the compressed measurements and the second one is the reconstruction algorithm to retrieve the desired signal. From the encoder-decoder perspective, we call the imaging system "hardware encoder" and the reconstruction algorithm "software decoder".

For the first challenge in video SCI, different hardware encoders have been built and the underlying principle is to modulate the high-speed scene with a higher frequency than the sampling speed of the camera (Fig. 1). Various coding strategies have been proposed, such as using a spatial light modulator (SLM), including a digital micromirror device (DMD) [10], [11], [16], [17], [18], [19] or a dynamic mask [7], [8]. The patterns of DMD will change tens of times during one exposure time of the camera to impose the compression; a physical mask is moving within one exposure time so that different variants of the mask are imposed on the high-speed scenes to achieve the high-speed modulation.

- *X. Yuan is with School of Engineering, Westlake University, Hangzhou 310024, Zhejiang, China. He was with Bell Labs, Murray Hill, New Jersey, 07974 USA.*
  E-mail: xyuan@westlake.edu.cn. (Corresponding author).
- *Y. Liu and F. Durand are with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*
  E-mails: {yliu12, fredo}@mit.edu.
- *J. Suo and Q. Dai are with the Department of Automation, Tsinghua University, Beijing 100084, China.*
  E-mails: {jlsuo, qhdai}@tsinghua.edu.cn.
- *Our code is available at https://github.com/liuyang12/PnP-SCI_python.*

Manuscript updated July 15, 2021.

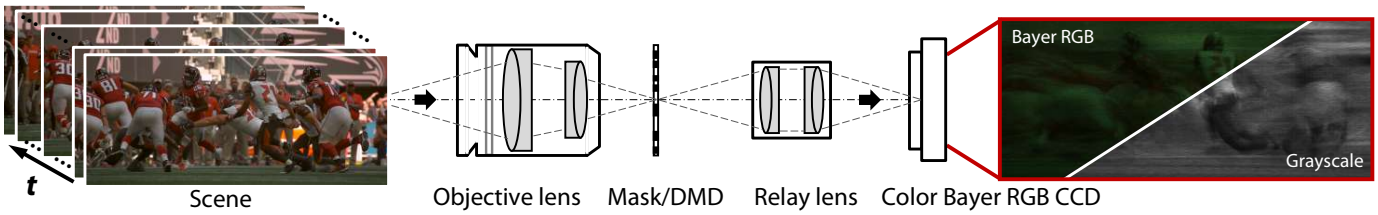


Fig. 1. Schematic of a color video SCI system and its snapshot measurement (showing in Bayer RGB mode). A "RGGB" Bayer pattern is shown.

Regarding the second challenge of software decoder, various algorithms have been employed and developed for SCI reconstruction. In addition to the widely used TwIST [20], Gaussian Mixture Model (GMM) in [21], [22] assumes the pixels within a spatial-temporal patch are drawn from a GMM. GAP-TV [23] adopts the idea of total variation (TV) minimization under the generalized alternating projection (GAP) [24] framework. Recently, DeSCI proposed in [25] has led to state-of-the-art results. However, the slow speed of DeSCI precludes its real applications, especially to the HD (1280×720), FHD (1920×1080) or UHD (3840×1644 and 3840×2160 in Fig. 7) videos, which are now commonly used in our daily life. Recall that DeSCI needs more than one hour to reconstruct a $256 \times 256 \times 8$ video from a snapshot measurement. GAP-TV, by contrast, as a fast algorithm, cannot provide decent reconstructions to be used in real applications (in general, this needs the PSNR >30dB). An alternative solution is to train an end-to-end network to reconstruct the videos [16], [26], [27], [28] for the SCI system. On one hand, this approach can finish the task within seconds (after training) and by the appropriate usage of multiple GPUs, an end-to-end sampling and reconstruction system can be built. On the other hand, this method loses the *robustness* of the network since whenever the sensing matrix (encoding process) changes, a new network has to be re-trained. Moreover, it cannot be used in adaptive video sensing [29].

Therefore, it is desirable to devise an *efficient* and *flexible* algorithm for video SCI reconstruction, especially for large-scale problems. This will pave the way of applying SCI in our daily life. Towards this end, this paper develops plug-and-play (PnP) algorithms for SCI.

1.1 Related Work

From the hardware side, in addition to capture high-speed videos, various other SCI systems have been developed to capture 3D multi/hyper-spectral images [15], [30], [31], [32], 4D spectral-temporal [5], spatial-temporal [18], depth [33], [34] and polarization [35] images, etc. These systems share the similar principle of modulating the high-dimensional signals using high-frequency patterns.

From the algorithm side, early systems usually employed the algorithms for inverse problems of other applications such as compressive sensing [36]. In general, the SCI reconstruction is an ill-posed problem and diverse priors and regularization methods have been used. Among these priors, the TV [37] and sparsity [7] are widely used. Representative algorithms include TwIST [20] and GAP-TV [23]. Recently developed algorithms specifically for SCI include GMM [21], [22] and DeSCI [25], where GMM methods use mixture of Gaussian distributions to model video patches and DeSCI applies weighted nuclear norm minimization [38] on video patches into the alternating direction method of multipliers (ADMM) [39] framework.

As mentioned before, one main bottleneck of these optimization algorithms is the slow running speed. Inspired by recent advances of deep learning on image restoration, researchers have started using deep learning in computational imaging [40], [41]. Some networks have been proposed for SCI reconstruction [14], [16], [26], [27], [42]. After training, these algorithms can provide results instantaneously and thus they can lead to end-to-end systems [28], [32], [43], [44] for SCI. However, these end-to-end deep learning methods rely heavily on the training data and further are not flexible. Specifically, when one network is trained for a specific SCI system, it cannot be used in other SCI systems provided different modulation patterns or different compression rates.

In summary, optimization methods are slow but deep learning algorithms are not flexible. To cope with these issues, most recently, researchers start to integrate the advantages of both by applying the deep denoisers into the PnP framework [45], [46], [47], [48]. Though PnP can date back to 2013 [45], it is getting powerful in real inverse problems because the usage of advanced deep denoising networks [49]. Recently, great successes have been achieved by PnP in other applications.

Bearing these concerns in mind, in this work, we integrate various denoisers into PnP framework for SCI reconstruction. Our PnP algorithms can not only provide excellent results but also are robust to different coding process and thus can be used in adaptive sensing and large-scale problems [50].

1.2 Contributions of This Work

Generally speaking, reconstruction of SCI aims to solve the trilemma, *i.e.*, speed, accuracy and flexibility. To address this challenge, our preliminary work [50] applied *frame-wise image denoiser* into the PnP framework to achieve excellent results in video SCI. Specially, we made the following contributions in [50].

1) Inspired by the plug-and-play ADMM [47] framework, we extend it to SCI and show that PnP-ADMM converges to a fixed point by considering the hardware constraints and the special structure of the sensing matrix in SCI [51].
2) We propose an efficient PnP-GAP algorithm by using various denoisers (Fig. 2) into the generalized alternating projection [23], [24] framework, which has a lower computational workload than PnP-ADMM. We prove that, under proper assumptions, the solution of PnP-GAP also converges.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

3) By employing the deep image denoiser FFDNet [52] into PnP-GAP, we show that a FHD color video (1920×1080×3×30 with 3 denoting the RGB channels and 30 the frame number) can be recovered from a snapshot measurement (Fig. 7) efficiently with PSNR close to 30dB. Compared with an end-to-end network [16], [43], dramatic resources have been saved since no re-training is required. This further makes the UHD compression using SCI to be feasible (a 3840×1644×3×40 video is reconstructed with PSNR above 30dB in Fig. 7).
4) We apply our developed PnP algorithms to extensive simulation and real datasets (captured by real SCI cameras) to verify the efficiency and robustness of our proposed algorithms.

Since videos are image sequences and they are highly correlated, it is expected that a *video denoiser* will boost up the results of video SCI reconstruction. Moreover, the color-video SCI has not been fully exploited by color image/video denoising algorithms. In particular, we make additional contributions in this paper.

5) In addition to the PnP-FFDNet [50], which integrates the image denoiser, FFDNet [52], into PnP-GAP, we further integrate the most recent video denoiser, FastDVDnet [53], into PnP-GAP to achieve better results than those reported in [50] (Fig. 2).
6) We propose joint reconstruction and demosaicing for color SCI video reconstruction. In color SCI, since each pixel only capture one of the red (R), green (G) or blue (B) channels, previous methods reconstruct each channel (as grayscale images/videos) separately and then use off-the-shelf demosaicing methods to get the final color video. To overcome the limitations in these steps, we jointly reconstruct and demosaic the color video in one shot and better results are achieved. We also build an RGB mid-scale size dataset as benchmark data for the color video SCI problem.
7) We verify the proposed PnP-FastDVDnet in the measurements captured by the newly built SCI camera in [16] at different compressive sampling rates from 10 to 50. This clearly demonstrates the feasibility and flexibility of the proposed algorithm on the large-scale data. Furthermore, this verifies that a video SCI system can capture high-speed videos at 2500 frames per second (fps) by using a camera working at 50 fps.

1.3 Organization of This Paper

The rest of this paper is organized as follows. Sec. 2 introduces the mathematical model of both grayscale and color SCI. Sec. 3 develops the PnP-ADMM under the SCI hardware constraints and develops the joint demosaicing and reconstruction for color SCI. Sec. 4 proposes the PnP-GAP algorithm and proves its convergence[1]. Sec. 5 integrates various denoisers into to the PnP framework for SCI reconstruction. Extensive results of both simulation (grayscale benchmark, mid-scale color and large-scale color) and real data are presented in Sec. 6 and Sec. 7, respectively. Sec. 8 concludes the paper.

1. We observed some error in the proof of the global convergence of PnP-GAP in [50]. Specifically, the lower bound of the second term in Eq. (25) in [50] should be 0. Therefore, the global convergence of PnP-GAP does not hold anymore. Instead, we provide another convergence proof of PnP-GAP in this paper.

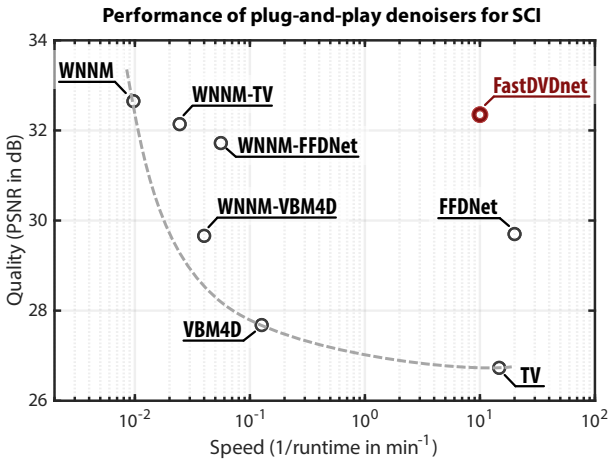


Fig. 2. Trade-off of quality and speed of various plug-and-play denoising algorithms for SCI reconstruction. Average PSNR of the six grays-scale datasets [50] are shown.

2 Mathematical Model of SCI

In the following, we first review the forward model of gray-scale video SCI. Then we extend the model to color video SCI and finally, we provide a unified model.

2.1 Gray-scale Video SCI

As depicted in Fig. 1, in the video SCI system *e.g.*, CACTI [7], consider that a B-frame (grayscale) video $\boldsymbol{X} \in \mathbb{R}^{n_x \times n_y \times B}$ is modulated and compressed by B sensing matrices (masks) $\boldsymbol{C} \in \mathbb{R}^{n_x \times n_y \times B}$, and the measurement frame $\boldsymbol{Y} \in \mathbb{R}^{n_x \times n_y}$ can be expressed as [7], [8]

$$\boldsymbol{Y} = \sum_{b=1}^{B} \boldsymbol{C}_b \odot \boldsymbol{X}_b + \boldsymbol{Z}, \tag{1}$$

where $\boldsymbol{Z} \in \mathbb{R}^{n_x \times n_y}$ denotes the noise; $\boldsymbol{C}_b = \boldsymbol{C}(:,:,b)$ and $\boldsymbol{X}_b = \boldsymbol{X}(:,:,b) \in \mathbb{R}^{n_x \times n_y}$ represent the b-th sensing matrix (mask) and the corresponding video frame respectively, and $\odot$ denotes the Hadamard (element-wise) product. Mathematically, the measurement in (1) can be expressed by

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{z}, \tag{2}$$

where $\boldsymbol{y} = \text{Vec}(\boldsymbol{Y}) \in \mathbb{R}^{n_x n_y}$ and $\boldsymbol{z} = \text{Vec}(\boldsymbol{Z}) \in \mathbb{R}^{n_x n_y}$ with $\text{Vec}(\cdot)$ vectorizing the ensued matrix by stacking columns. Correspondingly, the video signal $\boldsymbol{x} \in \mathbb{R}^{n_x n_y B}$ is

$$\boldsymbol{x} = \text{Vec}(\boldsymbol{X}) = [\text{Vec}(\boldsymbol{X}_1)^\mathsf{T}, ..., \text{Vec}(\boldsymbol{X}_B)^\mathsf{T}]^\mathsf{T}. \tag{3}$$

Unlike the global transformation based compressive sensing [54], the sensing matrix $\boldsymbol{H} \in \mathbb{R}^{n_x n_y \times n_x n_y B}$ in video SCI is sparse and is constituted by a concatenation of diagonal matrices

$$\boldsymbol{H} = [\boldsymbol{D}_1, ..., \boldsymbol{D}_B]. \tag{4}$$

where $\boldsymbol{D}_b = \text{diag}(\text{Vec}(\boldsymbol{C}_b)) \in \mathbb{R}^{n \times n}$ with $n = n_x n_y$, for $b = 1, \dots B$. Consequently, the *sampling rate* here is equal to $1/B$. It has been proved recently in [51], [55] that the reconstruction error of SCI is bounded even when $B > 1$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

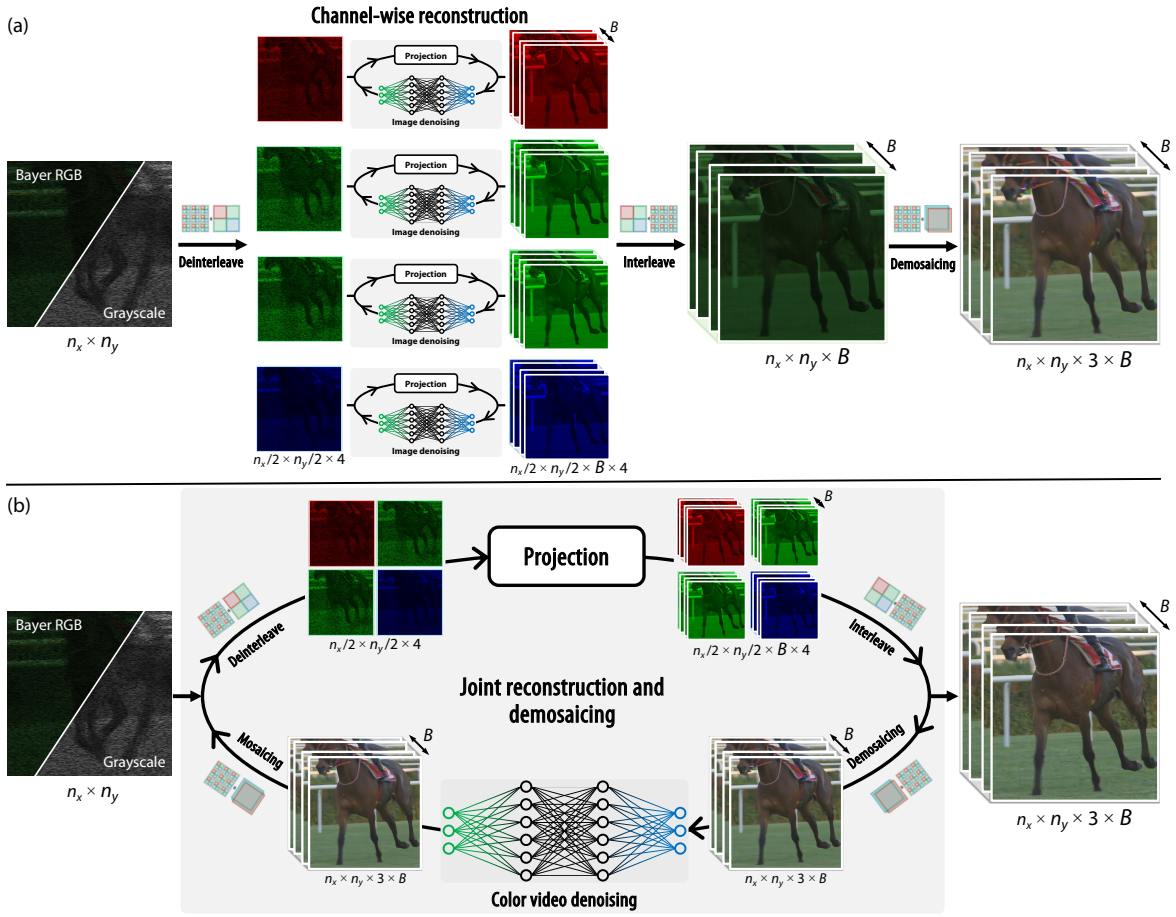


Fig. 3. Reconstruction of color SCI using mosaic sensor measurements. (a) Color SCI reconstruction by independently reconstruct RGGB channels using grayscale *image* denoising and then perform demosicing (we proposed this in [50]). The raw measurement (and the mask) is divided into four color channels, R (red), G1 (green), G2 (green) and B (blue) and these channels are reconstructed separately using the PnP-GAP with FFDNet. Then these channels are interleaved and demosaiced to obtain the final color video. (b) Proposed (in this paper) joint reconstruction and demosaicing for color SCI. The raw measurement (and the mask) is sent to the proposed PnP framework using GAP/ADMM with *color denoising* by FFDNet or FastDVDnet to output the desired color video directly. Note the demosaicing and *color video denoising* are embedded in each iteration.

2.2 Color Video SCI

In the color video case, as shown in Figs. 3, 6, 7 and 13, the raw data captured by the generally used Bayer pattern sensors have "RGGB" channels. Since the mask is imposed on each pixel, the generated measurement can be treated as a grayscale image as in Fig. 10 and when it is shown in color, the demosaicing procedure cannot generate the right color due to mask modulation (Fig. 3). In previous papers, during reconstruction, we first recover each of these four channel independently and then perform demosaicing in the reconstructed videos (upper part in Fig. 3). The final demosaiced RGB video is the desired signal [8], [50]. In this case, the raw measurement is decoupled into four components $\{\boldsymbol{Y}^{(r)}, \boldsymbol{Y}^{(g_1)}, \boldsymbol{Y}^{(g_2)}, \boldsymbol{Y}^{(b)}\} \in \mathbb{R}^{\frac{n_x}{2} \times \frac{n_y}{2}}$. Similarly, the corresponding masks and videos are denoted by $\{\boldsymbol{C}^{(r)}, \boldsymbol{C}^{(g_1)}, \boldsymbol{C}^{(g_2)}, \boldsymbol{C}^{(b)}\} \in \mathbb{R}^{\frac{n_x}{2} \times \frac{n_y}{2} \times B}$, $\{\boldsymbol{X}^{(r)}, \boldsymbol{X}^{(g_1)}, \boldsymbol{X}^{(g_2)}, \boldsymbol{X}^{(b)}\} \in \mathbb{R}^{\frac{n_x}{2} \times \frac{n_y}{2} \times B}$, respectively. The forward model for each channel is now

$$\begin{aligned}
\boldsymbol{Y}^{(r)} &= \textstyle\sum_{b=1}^{B} \boldsymbol{C}_b^{(r)} \odot \boldsymbol{X}_b^{(r)} + \boldsymbol{Z}^{(r)}, \\
\boldsymbol{Y}^{(g_1)} &= \textstyle\sum_{b=1}^{B} \boldsymbol{C}_b^{(g_1)} \odot \boldsymbol{X}_b^{(g_1)} + \boldsymbol{Z}^{(g_1)}, \\
\boldsymbol{Y}^{(g_2)} &= \textstyle\sum_{b=1}^{B} \boldsymbol{C}_b^{(g_2)} \odot \boldsymbol{X}_b^{(g_2)} + \boldsymbol{Z}^{(g_2)}, \\
\boldsymbol{Y}^{(b)} &= \textstyle\sum_{b=1}^{B} \boldsymbol{C}_b^{(b)} \odot \boldsymbol{X}_b^{(b)} + \boldsymbol{Z}^{(b)}.
\end{aligned} \tag{5}$$

In this color case, the desired signal is $\boldsymbol{X}^{(rgb)} \in \mathbb{R}^{n_x \times n_y \times 3 \times B}$, where 3 denotes the R, G and B channels in the color video. The demosaicing is basically an interpolation process from $\boldsymbol{X}^{(r)}$ to $\tilde{\boldsymbol{X}}^{(r)} \in \mathbb{R}^{n_x \times n_y \times B}$, from $\{\boldsymbol{X}^{(g_1)}, \boldsymbol{X}^{(g_2)}\}$ to $\tilde{\boldsymbol{X}}^{(g)} \in \mathbb{R}^{n_x \times n_y \times B}$ and from $\boldsymbol{X}^{(b)}$ to $\tilde{\boldsymbol{X}}^{(b)} \in \mathbb{R}^{n_x \times n_y \times B}$. Note that the interpolation rate for red and blue channel is from 1 pixel to 4 pixels, whereas for the green channel it is from 2 pixels to 4 pixels.

Utilizing the vectorized formulation, let $\{\tilde{\boldsymbol{x}}^{(r)}, \tilde{\boldsymbol{x}}^{(g)}, \tilde{\boldsymbol{x}}^{(b)}\} \in \mathbb{R}^{n_x n_y B}$ denote the vectorized representations of $\{\tilde{\boldsymbol{X}}^{(r)}, \tilde{\boldsymbol{X}}^{(g)}, \tilde{\boldsymbol{X}}^{(b)}\}$, $\{\boldsymbol{y}^{(r)}, \boldsymbol{y}^{(b)}\} \in \mathbb{R}^{\frac{n_x n_y}{4}}$ denote the vectorized representations of $\boldsymbol{Y}^{(r)}, \boldsymbol{Y}^{(b)}\}$ and $\boldsymbol{y}^{(g)} = \begin{bmatrix} \boldsymbol{y}^{(g_1)} \\ \boldsymbol{y}^{(g_2)} \end{bmatrix} \in \mathbb{R}^{\frac{n_x n_y}{2}}$ denote the concatenated vector-representation of $\{\boldsymbol{Y}^{(g_1)}, \boldsymbol{Y}^{(g_2)}\}$. Similar notations are also used for the noise term. We arrive at

$$\begin{aligned}
\boldsymbol{y}^{(r)} &= \boldsymbol{H}^{(r)} \tilde{\boldsymbol{x}}^{(r)} + \boldsymbol{z}^{(r)}, \\
\boldsymbol{y}^{(g)} &= \boldsymbol{H}^{(g)} \tilde{\boldsymbol{x}}^{(g)} + \boldsymbol{z}^{(g)}, \\
\boldsymbol{y}^{(b)} &= \boldsymbol{H}^{(b)} \tilde{\boldsymbol{x}}^{(b)} + \boldsymbol{z}^{(b)},
\end{aligned} \tag{6}$$

where $\{\boldsymbol{H}^{(r)}, \boldsymbol{H}^{(b)}\} \in \mathbb{R}^{\frac{n_x n_y}{4} \times n_x n_y B}$ and $\boldsymbol{H}^{(g)} \in \mathbb{R}^{\frac{n_x n_y}{2} \times n_x n_y B}$. The structures of $\{\boldsymbol{H}^{(r)}, \boldsymbol{H}^{(g)}, \boldsymbol{H}^{(b)}\}$ are similar to (4) for grayscale video but include the down-

sampling (mosaic) process, which decimates pixels in an interleaving way following the mosaic pattern of the sensor.

Following this, let the captured mosaic compressed measurement and the desired color video be $\boldsymbol{y} \in \mathbb{R}^{n_x n_y}$ and $\boldsymbol{x} \in \mathbb{R}^{3n_x n_y B}$, respectively. We have

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}^{(r)} \\ \boldsymbol{y}^{(g)} \\ \boldsymbol{y}^{(b)} \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} \tilde{\boldsymbol{x}}^{(r)} \\ \tilde{\boldsymbol{x}}^{(g)} \\ \tilde{\boldsymbol{x}}^{(b)} \end{bmatrix}, \tag{7}$$

and the full froward model of color-video SCI can be modeled by

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{H}^{(r)} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H}^{(g)} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{H}^{(b)} \end{bmatrix} \boldsymbol{x} + \boldsymbol{z}. \tag{8}$$

This formulation along with the grayscale one is the unique forward model of video SCI and apparently, the color one is more challenging.

2.3 A Unified Forward Model of Video SCI

From the gray-scale model in Eq. (2) and the color model in Eq. (8), we can see that $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{z}$ is a model can represent both cases. However, in order to unify the model, as shown in Fig. 3, we introduce $\boldsymbol{T}_{\mathcal{M}}$ to denote the mosaicing and deinterleaving process. We thus rewrite the forward model of color-SCI as

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{T}_{\mathcal{M}}\boldsymbol{x} + \boldsymbol{z}, \tag{9}$$

where the gray-scale and color cases can be represented respectively by

- For gray-scale videos, $\boldsymbol{T}_{\mathcal{M}} = \boldsymbol{I}$ is an identity matrix and $\boldsymbol{x} \in \mathbb{R}^{n_x n_y B}$ is the vectorized concatenation of B frames as in Eq. (3).
- For color videos, $\boldsymbol{x} \in \mathbb{R}^{3n_x n_y B}$ is the vectorized concatenation of B frames of RGB channels while $\boldsymbol{T}_{\mathcal{M}} \in \mathbb{R}^{n_x n_y B \times 3n_x n_y B}$ is the mosaicing and deinterleaving process. Note that $\boldsymbol{T}_{\mathcal{M}}$ can also be represented by a concatenation of diagonal matrices with the diagonal matrix being an identity matrix when the color channel is picked and being a zero matrix when the color channel is not picked.

As depicted in the top-part of Fig. 3, for color video SCI, previous studies usually first reconstruct the four Bayer channels of the video independently and then employ the off-the-shelf demosaicing algorithm to get the desired color videos [25], [50]. However, this final performance of the reconstructed video will be limited by both steps (channel-wise reconstruction and demosaicing). In this paper, we derive a joint reconstruction and demosaicing framework for color video SCI (lower part in Fig. 3) directly based on the model derived in (8). More importantly, it is a unified PnP framework, where different demosaicing and color denoising algorithms can be used.

3 Plug-and-Play ADMM for SCI

Apparently, $\boldsymbol{H}$ is a fat matrix, *i.e.*, more columns than rows and thus the inverse problem of SCI is an ill-posed problem. To solve this problem, a regularization term, a.k.a. a prior is generally employed. The inversion problem of SCI can thus be modeled as

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, f(\boldsymbol{x}) + \lambda g(\boldsymbol{x}), \tag{10}$$

where $f(\boldsymbol{x})$ can be seen as the forward imaging model, *i.e.*, $\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x}\|_2^2$ and $g(\boldsymbol{x})$ is a prior being used. While diverse priors have been used in SCI such as TV, sparsity and low-rank, in this work, we focus on the deep denoising prior, which has shown superiority recently on various image restoration tasks. Note that, since SCI systems aim to reconstruct high-speed video, a video deep denoising prior is desired [53]. On the other hand, since videos are essentially consequent images, recently advanced deep denoising priors for images can also be used [49], [52]. It has been shown in our preliminary paper that an efficient image denoising prior can lead to good results for SCI [50]. However, this frame-wise image denoising prior [52] limits the performance of video denoising since it ignored the strong temporal correlation in neighbouring frames. In this work, we employ the most recent video denoiser, FastDVDnet [53], as the denoising prior in our PnP framework and it leads to better results than those reported in [50].

3.1 Review the Plug-and-Play ADMM

Using ADMM [39], introducing an auxiliary parameter $\boldsymbol{v}$, the unconstrained optimization in (10) can be converted to

$$(\hat{\boldsymbol{x}}, \hat{\boldsymbol{v}}) = \operatorname{argmin}_{(\boldsymbol{x},\boldsymbol{v})}\, f(\boldsymbol{x}) + \lambda g(\boldsymbol{v}), \text{ subject to } \boldsymbol{x} = \boldsymbol{v}. \tag{11}$$

This minimization can be solved by the following sequence of sub-problems [47]

$$\boldsymbol{x}^{(k+1)} = \operatorname{argmin}_{\boldsymbol{x}}\, f(\boldsymbol{x}) + \tfrac{\rho}{2}\|\boldsymbol{x} - (\boldsymbol{v}^{(k)} - \tfrac{1}{\rho}\boldsymbol{u}^{(k)})\|_2^2, \tag{12}$$

$$\boldsymbol{v}^{(k+1)} = \operatorname{argmin}_{\boldsymbol{v}}\, \lambda g(\boldsymbol{v}) + \tfrac{\rho}{2}\|\boldsymbol{v} - (\boldsymbol{x}^{(k+1)} + \tfrac{1}{\rho}\boldsymbol{u}^{(k)})\|_2^2, \tag{13}$$

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + \rho(\boldsymbol{x}^{(k+1)} - \boldsymbol{v}^{(k+1)}), \tag{14}$$

where the superscript $^{(k)}$ denotes the iteration number.

In SCI and other inversion problems, $f(\boldsymbol{x})$ is usually a quadratic form and there are various solutions to Eq. (12). In PnP-ADMM, the solution of Eq. (13) is replaced by an *off-the-shelf* denoising algorithm, to yield

$$\boldsymbol{v}^{(k+1)} = \mathcal{D}_{\sigma_k}(\boldsymbol{x}^{(k+1)} + \tfrac{1}{\rho}\boldsymbol{u}^{(k)}). \tag{15}$$

where $\mathcal{D}_{\sigma_k}$ denotes the denoiser being used with σ_k being the standard deviation of the assumed additive white Gaussian noise in the k-th iteration. In [47], the authors proposed to update the ρ in each iteration by $\rho_{k+1} = \gamma_k \rho_k$ with $\gamma_k \geq 1$ and setting $\sigma_k = \sqrt{\lambda/\rho_k}$ for the denoiser. This essentially imposed the *non-increasing noise level* of the denoiser in Assumption 1 defined in Sec. 4.2. Chan *et al.* [47] defined the *bounded denoiser* and proved the *fixed point* convergence of the PnP-ADMM.

Definition 1. *(Bounded Denoiser [47]): A bounded denoiser with a parameter σ is a function $\mathcal{D}_\sigma : \mathbb{R}^n \to \mathbb{R}^n$ such that for any input $\boldsymbol{x} \in \mathbb{R}^n$,*

$$\tfrac{1}{n}\|\mathcal{D}_\sigma(\boldsymbol{x}) - \boldsymbol{x}\|_2^2 \leq \sigma^2 C, \tag{16}$$

for some universal constant C independent of n and σ.

With this definition (constraint on the denoiser) and the assumption of f: $[0,1]^n \to \mathbb{R}$ having bounded gradient, which is for any $\boldsymbol{x} \in [0,1]^n$, there exists $L < \infty$ such that $\|\nabla f(\boldsymbol{x})\|_2/\sqrt{n} \leq L$, the authors of [47] have proved that: the iterates of the PnP-ADMM demonstrates a fixed-point convergence. That is, there exists $(\boldsymbol{x}^*, \boldsymbol{v}^*, \boldsymbol{u}^*)$ such that $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_2 \to 0$, $\|\boldsymbol{v}^{(k)} - \boldsymbol{v}^*\|_2 \to 0$, and $\|\boldsymbol{u}^{(k)} - \boldsymbol{u}^*\|_2 \to 0$ as $k \to \infty$.

3.2 PnP-ADMM for SCI

We now consider the SCI model stated in Eq. (9). Specifically, we consider the loss function $f(\boldsymbol{x})$ as

$$f(\boldsymbol{x}) = \tfrac{1}{2}\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}\|_2^2. \tag{17}$$

Consider all the pixel values are normalized into $[0,1]$.

Lemma 1. *In SCI, the function $f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}\|_2^2$ has bounded gradients,* i.e., $\|\nabla f(\boldsymbol{x})\|_2 \leq B\|\boldsymbol{x}\|_2$.

Proof. The gradient of $f(\boldsymbol{x})$ in SCI is

$$\nabla f(\boldsymbol{x}) = \boldsymbol{T}_{\mathcal{M}}^{\mathsf{T}}\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H}\boldsymbol{T}_{\mathcal{M}}\boldsymbol{x} - \boldsymbol{T}_{\mathcal{M}}^{\mathsf{T}}\boldsymbol{H}^{\mathsf{T}}\boldsymbol{y}, \tag{18}$$

where $\boldsymbol{H}$ is a concatenation of diagonal matrices of size $n \times nB$ as shown in Eq. (4).

- The $\boldsymbol{T}_{\mathcal{M}}^{\mathsf{T}}\boldsymbol{H}^{\mathsf{T}}\boldsymbol{y}$ is a non-negative constant since both the measurement $\boldsymbol{y}$ and the mask (and $\boldsymbol{T}_{\mathcal{M}}$) are non-negative in nature.
- Now let's focus on $\boldsymbol{T}_{\mathcal{M}}^{\mathsf{T}}\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H}\boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}$. Firstly, consider $\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H}\boldsymbol{x}$; since

$$\begin{aligned}\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H} &= \begin{bmatrix}\boldsymbol{D}_1\\ \vdots \\ \boldsymbol{D}_B\end{bmatrix}\begin{bmatrix}\boldsymbol{D}_1 \cdots \boldsymbol{D}_B\end{bmatrix} \\ &= \begin{bmatrix}\boldsymbol{D}_1^2 & \boldsymbol{D}_1\boldsymbol{D}_2 & \cdots & \boldsymbol{D}_1\boldsymbol{D}_B \\ \boldsymbol{D}_1\boldsymbol{D}_2 & \boldsymbol{D}_2^2 & \cdots & \boldsymbol{D}_2\boldsymbol{D}_B \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{D}_1\boldsymbol{D}_B & \boldsymbol{D}_2\boldsymbol{D}_B & \cdots & \boldsymbol{D}_B^2\end{bmatrix},\end{aligned} \tag{19}$$

due to this special structure, $\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H}\boldsymbol{x}$ is the weighted sum of the $\boldsymbol{x}$ and $\|\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H}\boldsymbol{x}\|_2 \leq BC_{\max}\|\boldsymbol{x}\|_2$, where $C_{\max}$ is the maximum value in the sensing matrix. Usually, the sensing matrix is normalized to $[0,1]$ (where '1' means the light is fully transmitted while '0' means the light is blocked) and this leads to $C_{\max} = 1$ and therefore $\|\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H}\boldsymbol{x}\|_2 \leq B\|\boldsymbol{x}\|_2$. In addition, since $\boldsymbol{T}_{\mathcal{M}}^{\mathsf{T}}\boldsymbol{T}_{\mathcal{M}}$ is a diagonal matrix with the diagonal matrix being either 0 or 1, we have $\|\boldsymbol{T}_{\mathcal{M}}^{\mathsf{T}}\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H}\boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}\|_2 \leq B\|\boldsymbol{x}\|_2$, and therefore, the bound is the same for both gray-scale and color video SCI.

Thus, $\nabla f(\boldsymbol{x})$ is bounded. This bound can be further shrunk using the concentration of measure by imposing specific distributions on the sensing matrix [51]. □

Lemma 1 along with the bounded denoiser in Definition 1 gives us the following Corollary.

Corollary 1. *Consider the sensing model of SCI in* (9)*. Given $\{\boldsymbol{H}, \boldsymbol{T}_{\mathcal{M}}, \boldsymbol{y}\}$, $\boldsymbol{x}$ is solved iteratively via PnP-ADMM with bounded denoisers, then $\boldsymbol{x}^{(k)}$ and $\boldsymbol{\theta}^{(k)}$ will converge to a fixed point.*

Proof. The proof follows [47] and thus omitted here. □

We hereby give some comments on the convergence.

- The convergence proof of PnP-ADMM relies on the *bounded denoiser* assumption in Definition 1. Though it is not guaranteed that the employed FFDnet and Fast-DVDnet used in this work satisfy this assumption, we experimentally found that by the similar setting of the proposed σ_k (thus λ_k) in [47], the PnP-ADMM performs well in our SCI problem. We further provide a hyper-parameter study in Sec. 6.3. This introduces two future research directions: 1) modifying the deep denoising network to make it bounded, 2) deriving a new proof regime.
- Another convergence proof framework of PnP is using **Consensus Equilibrium** (CE) [56], which is based on the solution of a set of equilibrium equations that balance data fit and regularity. Some comments on the convergence of the proposed PnP-GAP using this CE regime are shown in Sec. 4.2.

3.3 Joint Demosaicing and SCI Reconstruction

For the color SCI described in Eq. (8), one straightforward way to solve it is to define $\tilde{\boldsymbol{H}} = \boldsymbol{H}\boldsymbol{T}_{\mathcal{M}}$, whic plays the role of joint mosaicing and video compressive sensing. In this way, we can employ the PnP algorithms developed for gray-scale video SCI (where $\boldsymbol{T}_{\mathcal{M}} = \boldsymbol{I}$) to solve it but using color denoising algorithms. Unfortunately, empirically we did not obtain good results in this manner; thus we have the following splitting derivation and make necessary changes to develop an efficient and stable solution for color SCI.

Specifically, introducing an auxiliary variable $\boldsymbol{w} = \boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}$, the color-SCI inversion problem can be formulated as

$$\begin{aligned}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{v}}, \hat{\boldsymbol{w}}) = \operatorname{argmin}_{(\boldsymbol{x},\boldsymbol{v},\boldsymbol{w})}\ &\tfrac{1}{2}\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{w}\|_2^2 + \lambda g(\boldsymbol{v}) \\ &\text{subject to } \boldsymbol{w} = \boldsymbol{T}_{\mathcal{M}}\boldsymbol{x} \text{ and } \boldsymbol{x} = \boldsymbol{v}\,.\end{aligned} \tag{20}$$

The Lagrangian of the minimization is

$$\begin{aligned}\mathcal{L}_{\rho,\ell,\lambda}(\boldsymbol{x},\boldsymbol{v},\boldsymbol{w};\boldsymbol{u},\boldsymbol{\theta}) = &\tfrac{1}{2}\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{w}\|_2^2 + \lambda g(\boldsymbol{v}) + \tfrac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2 \\ &+ \boldsymbol{u}^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{v}) + \tfrac{\ell}{2}\|\boldsymbol{w} - \boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}\|_2^2 + \boldsymbol{\theta}^{\mathsf{T}}(\boldsymbol{w} - \boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}).\end{aligned} \tag{21}$$

Invoking ADMM [39], this minimization can be solved by the following sequence of sub-problems:

$$\boldsymbol{w}^{(k+1)} = \operatorname*{argmin}_{\boldsymbol{w}} \tfrac{1}{2}\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{w}\|_2^2 + \tfrac{\ell}{2}\|\boldsymbol{w} - \boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}^{(k)}\|_2^2, \tag{22}$$

$$\boldsymbol{x}^{(k+1)} = \operatorname*{argmin}_{\boldsymbol{x}} \tfrac{\ell}{2}\|\boldsymbol{w}^{(k+1)} - \boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}\|_2^2 + \tfrac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{v}^{(k)}\|_2^2, \tag{23}$$

$$\boldsymbol{v}^{(k+1)} = \operatorname{argmin}_{\boldsymbol{v}} \tfrac{\rho}{2}\|\boldsymbol{x}^{(k+1)} - \boldsymbol{v}\|_2^2 + \lambda g(\boldsymbol{v}), \tag{24}$$

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + \rho(\boldsymbol{x}^{(k+1)} - \boldsymbol{v}^{(k+1)}), \tag{25}$$

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \ell(\boldsymbol{w}^{(k+1)} - \boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}^{(k+1)}), \tag{26}$$

where $^{(k)}$ again denotes the iteration number. Eq. (22) has a closed-form solution that can be computed efficiently due to the special structure of the SCI sensing matrix [25]. Eq. (24) is a denoising problem where various denoisers can be used, thus PnP. Eq. (23) is a quadratic form with a closed-form solution: $\boldsymbol{x}^{(k+1)} = (\rho\boldsymbol{I} + \ell\boldsymbol{T}_{\mathcal{M}}^{\mathsf{T}}\boldsymbol{T}_{\mathcal{M}})^{-1}(\ell\boldsymbol{T}_{\mathcal{M}}^{\mathsf{T}}\boldsymbol{w}^{(k+1)} + \rho\boldsymbol{v}^{(k)})$. However, this will lead to a similar solution as using $\tilde{\boldsymbol{H}} = \boldsymbol{H}\boldsymbol{T}_{\mathcal{M}}$, which as mentioned before, cannot provide good results unless with a good initialization, which however, will add the running time.

Hereby, we tackle Eq. (23) from another perspective, in fact from a PnP angle. Note that the key of using *implicit*

priors in (24) is to solve $\boldsymbol{v}$ by PnP using some pre-defined (or pre-trained networks) space and push the solution to (or getting close to) this space. Based on this principle, since (23) is basically a demosaicing process, which is ill-posed and the second term $\frac{\rho}{2}\|\boldsymbol{x}-\boldsymbol{\theta}^{(k)}\|_2^2$ plays the role of priors (or pre-defined space), instead of using this term directly, we *relax* it with a prior $\psi(\boldsymbol{x})$. Eq. (23) can now be formulated as

$$\boldsymbol{x}^{(k+1)} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \frac{\ell}{2}\|\boldsymbol{w}^{(k+1)} - \boldsymbol{T}_{\mathcal{M}}\boldsymbol{x}\|_2^2 + \rho\,\psi(\boldsymbol{x}). \quad (27)$$

Using the idea of PnP, this can be solved by off-the-shelf demosaicing algorithms [57], [58] and we denote it by

$$\boldsymbol{x}^{(k+1)} = \mathcal{D}_{\mathcal{M}}(\boldsymbol{w}^{(k+1)}). \quad (28)$$

This is yet another PnP step in our algorithm and thus we have two PnP steps for color SCI reconstruction. In the experiments, we have further found that due to the introduction of $\psi(\boldsymbol{x})$, the update of $\boldsymbol{\theta}$ is not necessary. In other words, the PnP-ADMM for color SCI decomposes the denoising step in (15) into the following 2 steps:

$$\tilde{\boldsymbol{v}}^{k+1} = \mathcal{D}_{\mathcal{M}}(\boldsymbol{x}^{(k+1)} + \tfrac{1}{\rho}\boldsymbol{u}^{(k)}),\ \ \boldsymbol{v}^{k+1} = \mathcal{D}_{\sigma}(\tilde{\boldsymbol{v}}^{k+1}), \quad (29)$$

where $\mathcal{D}_{\mathcal{M}}$ is the (PnP) demosaicing algorithm being used[2] and $\mathcal{D}_{\sigma}$ now denotes the *color denoising* algorithms.

In the experiments, we have found that both 'Malvar04' [59] and 'Menon07' [60] can lead to stable results and they are performing better than the fast bilinear interpolation. However, the time consumption of 'Menon07' is about 4× longer than 'Malvar04' with limited gain. Therefore, we used 'Malvar04' in our color-SCI reconstruction. We believe a deep learning based algorithm can lead to better results for demosaicing. However, we noticed that existing deep learning based demosaicing is not stable in our color-SCI task, though they do perform well for the sole demosaicing task. We leave this robust demosaicing deep learning network for the future work. Nonetheless, our proposed PnP algorithm can adopt any new demosaicing and denoising algorithms to improve the results.

4 PLUG-AND-PLAY GAP FOR SCI

In this section, following the generalized alternating projection (GAP) algorithm [24] and the above conditions on PnP-ADMM, we propose the PnP-GAP for SCI, which has a lower computational workload (thus faster) than PnP-ADMM. For the sake of simplicity, we omit $\boldsymbol{T}_{\mathcal{M}}$ (since it only adds one more step in the solution as derived above) in the following derivation.

4.1 Algorithm

Different from the ADMM in Eq. (11), GAP solves SCI by the following problem

$$(\hat{\boldsymbol{x}},\hat{\boldsymbol{v}}) = \operatorname{argmin}_{\boldsymbol{x},\boldsymbol{v}} \tfrac{1}{2}\|\boldsymbol{x}-\boldsymbol{v}\|_2^2 + \lambda g(\boldsymbol{v}),\ \text{s.t.}\ \ \boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}. \quad (30)$$

Similarly to ADMM, the minimizer in Eq. (30) is solved by a sequence of subproblems and we again let k denotes the iteration number.

2. Demosaicing is conducted after interleaving. Similarly, mosaicing and then deinterleaving are conducted before projection in Eq. (31), as shown in Fig. 3(b).

- Solving $\boldsymbol{x}$: given $\boldsymbol{v}$, $\boldsymbol{x}^{(k+1)}$ is updated via an Euclidean projection of $\boldsymbol{v}^{(k)}$ on the linear manifold: $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}$,

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{v}^{(k)} + \boldsymbol{H}^{\mathsf{T}}(\boldsymbol{H}\boldsymbol{H}^{\mathsf{T}})^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{v}^{(k)}). \quad (31)$$

  Recalling (4), $\{\boldsymbol{D}_i\}_{i=1}^B$ is a diagonal matrix $\boldsymbol{D}_i = \operatorname{diag}(D_{i,1},\ldots,D_{i,n})$. Thereby, $\boldsymbol{H}\boldsymbol{H}^{\mathsf{T}}$ is diagonal matrix, *i.e.*,

$$\boldsymbol{R} = \boldsymbol{H}\boldsymbol{H}^{\mathsf{T}} = \operatorname{diag}(R_1,\ldots,R_n), \quad (32)$$

  where $R_j = \sum_{b=1}^{B} D_{i,j}^2, \forall j = 1,\ldots,n$. Eq. (31) can thus be computed efficiently.
- Solving $\boldsymbol{v}$: given $\boldsymbol{x}$, updating $\boldsymbol{v}$ can be seen as a denoising problem and

$$\boldsymbol{v}^{(k+1)} = \mathcal{D}_{\sigma}(\boldsymbol{x}^{(k+1)}). \quad (33)$$

  Here, various denoiser can be used with $\sigma = \sqrt{\lambda}$.
  Similar to PnP-ADMM, for color video SCI, (33) will be decomposed into 2 steps: $\tilde{\boldsymbol{v}}^{k+1} = \mathcal{D}_{\mathcal{M}}(\boldsymbol{x}^{(k+1)}), \boldsymbol{v}^{k+1} = \mathcal{D}_{\sigma}(\tilde{\boldsymbol{v}}^{k+1})$, where $\mathcal{D}_{\mathcal{M}}$ and $\mathcal{D}_{\sigma}$ are the demosaicing and color denoising algorithms defined in (28) and (29), respectively.

Algorithm 1 Plug-and-Play GAP for SCI

Require: $\boldsymbol{H}$, $\boldsymbol{y}$.
Initial $\boldsymbol{v}^{(0)}$, λ_0, $\xi < 1$.
while Not Converge **do**
  Update $\boldsymbol{x}$ by Eq. (31).
  if Gray-scale videos **then**
    Update $\boldsymbol{v}$ by denoiser $\boldsymbol{v}^{(k+1)} = \mathcal{D}_{\sigma_k}(\boldsymbol{x}^{(k+1)})$.
  else if Color videos **then**
    $\tilde{\boldsymbol{v}}^{(k+1)} = \mathcal{D}_{\mathcal{M}}(\boldsymbol{x}^{(k+1)}), \boldsymbol{v}^{k+1} = \mathcal{D}_{\sigma_k}(\tilde{\boldsymbol{v}}^{(k+1)})$.
  end if
  if $\Delta_{k+1} \geq \eta\Delta_k$ **then**
    $\lambda_{k+1} = \xi\lambda_k$.
  else
    $\lambda_{k+1} = \lambda_k$.
  end if
  $\sigma_{k+1} = \sqrt{\lambda_{k+1}}$.
end while

We can see that in each iteration, the only parameter to be tuned is λ and we thus set $\lambda_{k+1} = \xi_k\lambda_k$ with $\xi_k \leq 1$. Inspired by the PnP-ADMM, we update λ by the following two rules:

a) Monotone update by setting $\lambda_{k+1} = \xi\lambda_k$, with $\xi < 1$.
b) Adaptive update by considering the relative residue:

$$\Delta_{k+1} = \tfrac{1}{\sqrt{nB}}\left(\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|_2 + \|\boldsymbol{v}^{(k+1)} - \boldsymbol{v}^{(k)}\|_2\right).$$

For any $\eta \in [0,1)$ and let $\xi < 1$ be a constant, λ_k is conditionally updated according to the following settings:
i) If $\Delta_{k+1} \geq \eta\Delta_k$, then $\lambda_{k+1} = \xi\lambda_k$.
ii) If $\Delta_{k+1} < \eta\Delta_k$, then $\lambda_{k+1} = \lambda_k$.

With this adaptive updating of λ_k, the full PnP-GAP algorithm for SCI is exhibited in Algorithm 1.

4.2 Convergence

It can be seen from Algorithm 1 that in each iteration, we only need to tune the noise level of the denoiser. Intuitively, similar to other optimization based algorithms in the inversion algorithms [61], we expect the noise level to be big in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

the first few iterations and it is getting smaller when the iteration increases. Based on this intuition, we make the following assumption on the *noise level of the denoiser* in PnP, which will help to prove the convergence of PnP-GAP.

Assumption 1. *(Non-increasing noise level) The denoiser in each iteration of PnP-GAP $\mathcal{D}_{\sigma_k}: \mathbb{R}^{nB} \to \mathbb{R}^{nB}$ performs denoising in a non-increasing order,* i.e., *$\sigma_{k+1} \leq \sigma_k$. Further, when $k \to +\infty$, $\sigma_k \to 0$.*

As mentioned above, as the algorithm proceeds we expect the algorithm's estimate of the underlying signal to become more accurate, which means that the denoiser needs to deal with a less noisy signal. This is also guaranteed by the λ setting in Algorithm 1 and imposed by ρ setting in the PnP-ADMM [47]. With this assumption, we have the following convergence result of PnP-GAP.

Theorem 1. *Consider the sensing model of SCI. Given $\{\boldsymbol{H}, \boldsymbol{y}\}$, $\boldsymbol{x}$ is solved by PnP-GAP with a bounded denoiser with noise level in a non-increasing order, then $\boldsymbol{x}^{(k)}$ converges.*

Proof. From (31), $\boldsymbol{x}^{(k+1)} = \boldsymbol{v}^{(k)} + \boldsymbol{H}^\top(\boldsymbol{H}\boldsymbol{H}^\top)^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{v}^{(k)})$, we have

$$\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} = \boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)} + \boldsymbol{H}^\top \boldsymbol{R}^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{v}^{(k)}). \quad (34)$$

Following this,

$$\begin{aligned}
&\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|_2^2 \\
=&\|\boldsymbol{v}^{(k)} + \boldsymbol{H}^\top \boldsymbol{R}^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{v}^{(k)}) - \boldsymbol{x}^{(k)}\|_2^2 && (35)\\
=&\|\boldsymbol{v}^{(k)} + \boldsymbol{H}^\top \boldsymbol{R}^{-1}(\boldsymbol{H}\boldsymbol{x}^{(k)} - \boldsymbol{H}\boldsymbol{v}^{(k)}) - \boldsymbol{x}^{(k)}\|_2^2 \\
=&\|(\boldsymbol{I} - \boldsymbol{H}^\top \boldsymbol{R}^{-1}\boldsymbol{H})(\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)})\|_2^2 && (36)\\
=&\|\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)}\|_2^2 - \|\boldsymbol{R}^{-\frac{1}{2}}\boldsymbol{H}(\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)})\|_2^2 && (37)\\
\leq&\|\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)}\|_2^2 && (38)\\
=&\|\mathcal{D}_{\sigma_k}(\boldsymbol{x}^{(k)}) - \boldsymbol{x}^{(k)}\|_2^2 && (39)\\
\leq&\sigma_k^2 nBC, && (40)
\end{aligned}$$

where $\boldsymbol{R} = \boldsymbol{H}\boldsymbol{H}^\top$ as defined in (32) and the following shows the derivation from (36) to (37)

$$\begin{aligned}
&\|(\boldsymbol{I} - \boldsymbol{H}^\top \boldsymbol{R}^{-1}\boldsymbol{H})(\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)})\|_2^2 \\
&= (\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)})^\top [\boldsymbol{I} - \boldsymbol{H}^\top(\boldsymbol{H}\boldsymbol{H}^\top)^{-1}\boldsymbol{H}]^\top \\
&\quad \cdot [\boldsymbol{I} - \boldsymbol{H}^\top(\boldsymbol{H}\boldsymbol{H}^\top)^{-1}\boldsymbol{H}](\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)}) \\
&= (\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)})^\top [\boldsymbol{I} - 2\boldsymbol{H}^\top \boldsymbol{R}^{-1}\boldsymbol{H} + \boldsymbol{H}^\top \boldsymbol{R}^{-1}\boldsymbol{R}\boldsymbol{R}^{-1}\boldsymbol{H}] \\
&\quad \cdot (\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)}) \\
&= (\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)})^\top (\boldsymbol{I} - \boldsymbol{H}^\top \boldsymbol{R}^{-1}\boldsymbol{H})(\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)}) \\
&= \|\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)}\|_2^2 - (\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)})^\top \boldsymbol{H}^\top \boldsymbol{R}^{-1}\boldsymbol{H}(\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)}) \\
&= \|\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)}\|_2^2 - \|\boldsymbol{R}^{-\frac{1}{2}}\boldsymbol{H}(\boldsymbol{v}^{(k)} - \boldsymbol{x}^{(k)})\|_2^2.
\end{aligned}$$

In (40) we have used the bounded denoiser. Using Assumption 1 (non-increasing noise level), we have $\sigma_k \to 0$, $\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|_2^2 \to 0$ and thus $\boldsymbol{x}^{(k)}$ converges. □

As mentioned in Sec. 3, the proof here is based on the assumption of bounded denoiser. It is possible to relax this condition using the framework of consensus equilibrium [56]. Under CE, the SCI reconstruction problem can be reformulated as the solution of a set of equilibrium equations that balance data fit and regularity. In each iteration, we can employ *multiple numbers of deep denoiser with different settings* of the noise level σ_k. As used in [56] for the image denoising applications, by imposing different weights to the solutions of these deep denoisers, we can make sure the denoising results converges to the fixed point. Combing this with (39), we can conduct that $\mathcal{D}_{\sigma_k}(\boldsymbol{x}^{(k)}) \to \boldsymbol{x}^{(k)}$, which will lead to $\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|_2^2 \to 0$ and thus PnP-GAP converges.

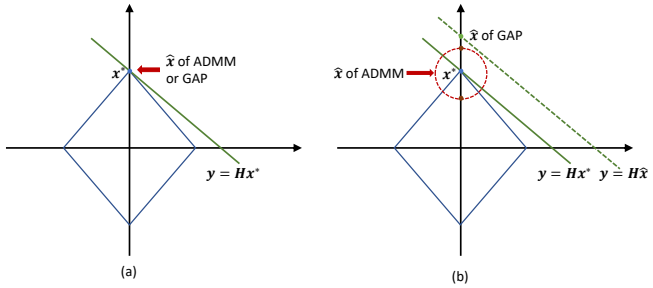


Fig. 4. Demonstration of the solution of ADMM and GAP for a two-dimensional sparse signal, where $\boldsymbol{x}^*$ denotes the truth. (a) In the noise-free case, both ADMM and GAP have a large chance to converge to the true signal. (b) In the noisy case, GAP will converge to the green dot (the cross-point of dash-green line and vertical axis), whereas the solution of ADMM will be one of the two red dots that the red circle crosses the vertical axis.

4.3 PnP-ADMM vs. PnP-GAP

Comparing PnP-GAP in Eqs (31) and (33) and PnP-ADMM in Eqs (12)-(14), we can see that PnP-GAP only has two subproblems (rather than three as in PnP-ADMM) and thus the computation is faster. It was pointed out in [25] that in the noise-free case, ADMM and GAP perform the same with appropriate parameter settings, which has been mathematically proved. However, in the noisy case, ADMM usually performs better since it considers noise in the model and below we give a geometrical explanation.

In Fig. 4, we use a two-dimensional sparse signal (with the ℓ_1 assumption shown as the diamond shape in blue lines in Fig. 4) as an example to compare ADMM and GAP. Note that the key difference is that, in both noise-free and noisy cases, GAP always imposes the solution $\hat{\boldsymbol{x}}$ on the line of $\boldsymbol{y} = \boldsymbol{H}\hat{\boldsymbol{x}}$ by Eq. (31). In the noise-free case in Fig. 4(a), we can see that since GAP imposes $\hat{\boldsymbol{x}}$ on the green line, it will converge to the true signal $\boldsymbol{x}^*$. ADMM does not have this constraint but minimizes $\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x}\|_2^2$, the solution might be a little bit off the true signal $\boldsymbol{x}^*$. However, with appropriate parameter settings and a good initialization, it also has a large chance to converge to the true signal. In the noisy case, GAP sill imposes $\hat{\boldsymbol{x}}$ on the line of $\boldsymbol{y} = \boldsymbol{H}\hat{\boldsymbol{x}}$, shown by the dash-green line in Fig. 4(b). In this case, due to noise, this line might deviate from the solid green line where the true signal lies on. GAP will thus converge to the green-point where the dash-green line crosses the vertical axis. On the other hand, by minimizing $\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x}\|_2^2$, the solution of ADMM can be in the dash-red circle depending on the initialization. Considering the sparse constraint, the final solution of ADMM would be one of the two red dots that the red circle crosses the vertical axis. Therefore, in the noisy case, the Euclidean distance between the GAP solution and the true signal ($\|\hat{\boldsymbol{x}} - \boldsymbol{x}^*\|_2$) might be larger than that of ADMM. However, the final solution of ADMM depends on the initialization and it is not guaranteed to be more accurate than GAP.

The PnP framework can be recognized as a deep denoising network plus an inverse problem solver. Other solvers

such as TwIST [20] and FISTA [62] can also be used [63] and may also lead to convergence results under proper conditions. According to our experience, TwIST usually converges slowly and FISTA sometimes sticks to limited performance. More analysis can be found in [64]. Hence, in the experiments, we use PnP-GAP for simulation data and PnP-ADMM for real data.

5 Integrate Various Denoisers into PnP for SCI Reconstruction

In the above derivation, we assume the denoiser existing and in this section, we briefly introduce different denoisers. These denoisers have different speed and quality.

5.1 Non-deep Denoiser

In conventional denoising algorithms, a prior is usually employed to impose the piece-wise constant (by TV), sparsity (by bases or learn-able dictionaries) or low-rank (similar patches groups). These algorithms usually have a clear objective function. In the following, we briefly categorize these algorithms into following classes. For a detailed review, please refer to [65].

- Global constraint based algorithms such as TV [37] minimize the total variations of the entire image and have been extended to videos [66].
- Global sparsity based algorithms impose the coefficients of the image under specific basis such as wavelet or Shearlet [67].
- Patch based algorithms usually learn a dictionary using methods such as K-SVD [68] for image patches and then impose sparsity on the coefficients.
- Patch-group based algorithms exploit the nonlocal similarity of image patches and impose the sparsity [69] or low-rank [38] on these similar patch groups.

Among these denoisers, usually, a faster denoiser *e.g.*, TV, is very efficient, but cannot provide high-quality results. The middle class algorithms *e.g.*, K-SVD and BM3D [70] can provide decent results with a longer running time. More advanced denoising algorithm such as WNNM [38] can provide better results, but even slower. On the other hand, while extensive denoising algorithms for images have been developed, VBM4D [71] is still one of the state-of-the-art algorithms for video denoising. In our previous work, we have extended WNNM into SCI for gray-scale videos leading to the state-of-the-art algorithm (DeSCI [25]) for SCI, which performs better than PnP-VBM4D as shown in Fig. 2 but paying the price of a longer running time.

5.2 Deep Denoiser

Another line of emerging denoising approaches is based on deep learning [49], which can provide decent results within a short time after training, but they are usually not robust to noise levels and in high noisy cases, the results are not good. Different from conventional denoising problems, in SCI reconstruction, the noise level in each iteration is usually from large to small and the dynamic range can from 150 to 1, considering the pixel values within $\{0, 1, \ldots, 255\}$. Therefore, a flexible denoiser that is robust to the input noise level is desired. Fortunately, FFDNet [52] has provided us a fast and flexible solution under various noise levels. However, since FFDNet is developed for images, we perform the denoising step in PnP for SCI frame-wise; for the color SCI problem, we used the gray-scale denoising for each channel in [50]. As discussed before and shown in Fig. 3, using joint demosaicing and reconstruction by employing color denoiser of FFDNet instead of grayscale ones can improve the results significantly. Please refer to Table 2 and Fig. 6 .

Most recently, we notice that the FastDVDnet [53] also satisfies these desired (fast and flexible) properties. Importantly, FastDVDnet is developed for video denoising which takes account of the strong temporal correlation within consequent video frames. By using FastDVDnet into PnP, we have achieved even better results on both grayscale and color videos than those of FFDNet. Please refer to Table 1 for grayscale video SCI results and Table 2 for the color SCI.

5.3 Hybrid Denoiser

By integrating these denoising algorithms into PnP-GAP/ADMM, we can have different algorithms (Table 1 and Fig. 2) with different results. It is worth noting that DeSCI can be seen as PnP-WNNM, and its best results are achieved by exploiting the correlation across video frames. On the other hand, most existing deep denoising priors are still based on images. Therefore, it is not unexpected that the results of PnP-GAP/ADMM-FFDNet are not as good as DeSCI. As mentioned above, by using video denoising priors such as FastDVDnet, the results can be improved.

In addition, these different denoisers can be used in parallel, *i.e.*, one after each other in one GAP/ADMM iteration or used sequentially, *i.e.*, the first K_1 iterations using FFDNet and the next K_2 iterations using WNNM to achieve better results. This is a good way to balance the performance and running time. Please refer to the performance and running time in Fig. 2 and Table 1. These different denoising priors can also be served as the complementary priors in image/video denoising [16].

6 Simulation Results

We apply the proposed PnP algorithms to both simulation [25], [26] and real datasets captured by the SCI cameras [7], [8], [16]. In addition to the widely used grayscale benchmark datasets [50], we also build a mid-scale color dataset consisting of 6 color videos (details in Sec. 6.2) and we hope they will serve as the benchmark data of color SCI problems. This benchmark dataset is used to verify the performance of our proposed PnP-GAP for joint reconstruction and demosaicing compared with other algorithms. Finally, we apply the proposed joint method to the large-scale datasets introduced in [50] and show better results using FastDVDnet for color video denoising.

Conventional denoising algorithms include TV [23], VBM4D [71] and WNNM [38] are used for comparison. For the deep learning based denoiser, we have tried various algorithms and found that FFDNet [52] provides the best results among image denoising methods while FastDVDnet [53] provides the best results among video denoising approaches. Both PSNR and SSIM [72] are employed

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 1
Grayscale benchmark dataset: The average results of PSNR in dB (left entry in each cell) and SSIM (right entry in each cell) and run time per measurement/shot in minutes by different algorithms on 6 benchmark datasets.

Algorithm	Kobe	Traffic	Runner	Drop	Crash	Aerial	Average	Run time (min)
GAP-TV [23]	26.46, 0.8448	20.89, 0.7148	28.52, 0.9092	34.63, 0.9704	24.82, 0.8383	25.05, 0.8281	26.73, 0.8509	0.07
DeSCI [25]	**33.25**, 0.9518	**28.71**, 0.9250	**38.48**, **0.9693**	43.10, 0.9925	27.04, 0.9094	25.33, 0.8603	**32.65**, 0.9347	103.0
PnP-VBM4D	30.60, 0.9260	26.60, 0.8958	30.10, 0.9271	26.58, 0.8777	25.30, 0.8502	26.89, 0.8521	27.68, 0.8882	7.9
PnP-FFDNet [50]	30.50, 0.9256	24.18, 0.8279	32.15, 0.9332	40.70, 0.9892	25.42, 0.8493	25.27, 0.8291	29.70, 0.8924	0.05 (GPU)
PnP-WNNM-TV	33.00, 0.9520	26.76, 0.9035	38.00, 0.9690	43.27, 0.9927	26.25, 0.8972	25.53, 0.8595	32.14, 0.9290	40.8
PnP-WNNM-VBM4D	33.08, **0.9537**	28.05, 0.9191	33.73, 0.9632	28.82, 0.9289	26.56, 0.8874	27.74, 0.8852	29.66, 0.9229	25.0
PnP-WNNM-FFDNet	32.54, 0.9511	26.00, 0.8861	36.31, 0.9664	**43.45**, **0.9930**	26.21, 0.8930	25.83, 0.8618	31.72, 0.9252	17.9
GAP-TV*	26.92, 0.8378	20.66, 0.6905	29.81, 0.8949	34.95, 0.9664	24.48, 0.7988	24.81, 0.8105	26.94, 0.8332	**0.03**
PnP-FFDNet*	30.33, 0.9252	24.01, 0.8353	32.44, 0.9313	39.68, 0.9864	24.67, 0.8330	24.29, 0.8198	29.21, 0.8876	**0.03** (GPU)
PnP-FastDVDnet*	32.73, 0.9466	27.95, **0.9321**	36.29, 0.9619	41.82, 0.9892	**27.32**, **0.9253**	**27.98**, **0.8966**	32.35, **0.9420**	0.10 (GPU)

* Implemented with Python (PyTorch for FFDNet and FastDVDnet), where the rest are implemented with MATLAB (MatConvNet for FFDNet).

as metrics to compare different algorithms. Note that in our preliminary paper [50], all the codes are conducted in MATLAB, while in this work, we re-write the code of GAP-TV, PnP-FFDNet using Python, to be consistent with PnP-FastDVDnet. However, we do notice a difference of FFDNet; the performance of Python version[3] is a little bit worse (0.49dB lower in PSNR for the grayscale benchmark data) than the counterpart conducted in MATLAB[4]. We also notice that GAP-TV in Python is more than $2\times$ faster than MATLAB with a slightly better result. We show these in Table 1.

6.1 Benchmark Data: Grayscale Videos

We follow the simulation setup in [25] of six datasets, *i.e.*, `Kobe`, `Traffic`, `Runner`, `Drop`, `Crash`, and `Aerial` [26][5], where $B = 8$ video frames are compressed into a single measurement. Table 1 summarizes the PSNR and SSIM results of these 6 benchmark data[6] using various denoising algorithms, where DeSCI can be categorized as GAP-WNNM, and PnP-WNNM-FFDNet used 50 iterations FFDNet and then 60 iterations WNNM, similar for PnP-WNNM-VBM4D. PnP-FastDVDnet used 60 iterations and we used 5 neighbouring frames for video denoising. It can be observed that:

i) By using GPU, PnP-FFDNet is now the fastest algorithm[7]; it is very close to GAP-TV, meanwhile providing more than 2dB higher PSNR than GAP-TV. Therefore, PnP-FFDNet can be used as *an efficient baseline* in SCI reconstruction. Since the average PSNR is close to 30dB, it is applicable in real cases. This will be further verified in the following subsections on mid-scale and large-scale color datasets.

ii) DeSCI still provides the best results on average PSNR; however, by combing other algorithms with WNNM, comparable results (*e.g.*PnP-WNNM-FFDNet) can be achieved by only using $1/6$ computational time.

iii) PnP-FastDVDnet provides the best results on average SSIM and regarding PSNR, it is only 0.3dB lower than DeSCI but $1000\times$ faster. PnP-FastDVDnet is the only algorithm that can provide a higher SSIM than DeSCI.

iv) Comparing PnP-FFDnet with PnP-FastDVDnet, we observe that utilizing the temporal correlation has improved the results significantly, *i.e.*, more than 3dB in PSNR and 0.05 in SSIM.

Fig. 5 plots selected frames of the six datasets using different algorithms. It can be seen that though DeSCI still leads to the

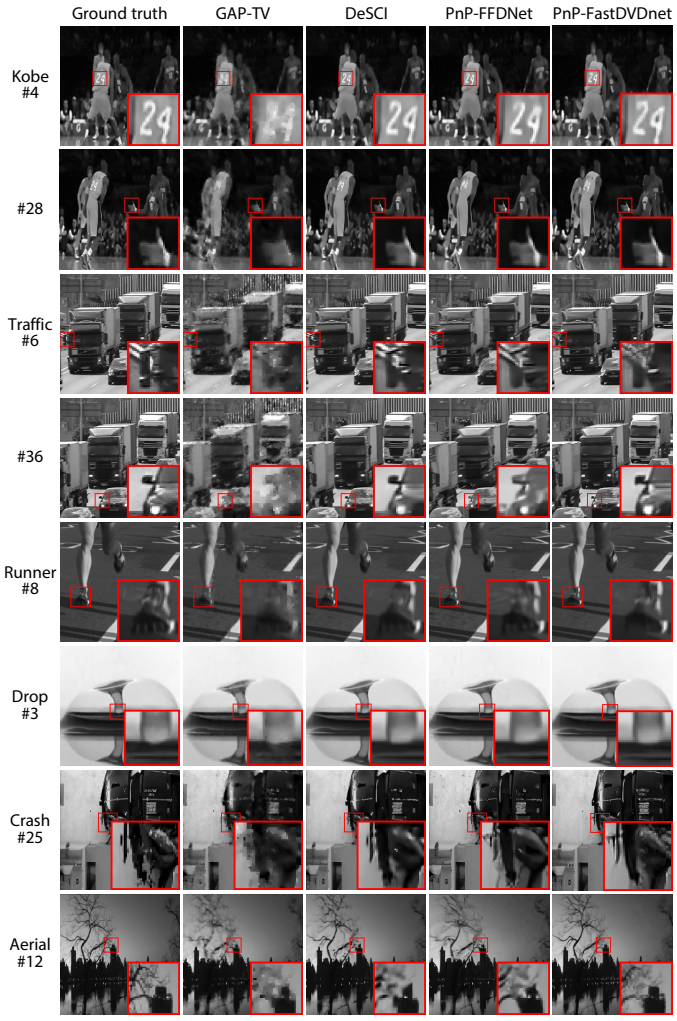


Fig. 5. Comparison of reconstructed frames of different PnP-GAP algorithms (GAP-TV [23], DeSCI [25], PnP-FFDNet [50], and PnP-FastDVDnet) on six simulated grayscale video SCI datasets of spatial size 256×256 and $B = 8$.

3. Code from https://github.com/cszn/KAIR.

4. Code from https://github.com/cszn/FFDNet.

5. The results of DeSCI (GAP-WNNM) is different from those reported in [26] because of parameter settings of DeSCI, specifically the input estimated noise levels for each iteration stage. We use exactly the same parameters as the DeSCI paper [25], which is publicly available at https://github.com/liuyang12/DeSCI.

6. Note that though we shared more measurements on the Github, for the results in Table 1,we used 4 measurements in `Kobe`, 6 measurements in `Traffic`, 1 measurement in `runner` and `Drop`, 4 measurements in `Crash` and `Aerial`.

7. Only a regular GPU is needed to run FFDNet and since FFDNet is performed in a frame-wise manner, we do not need a large amount of CPU or GPU RAM (no more than 2GB here) compared to other video denoisers using parallelization (even with parallelization, other algorithms listed here are unlikely to outperform PnP-FFDNet in terms of speed).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 2
Mid-scale Bayer benchmark dataset: the average results of PSNR in dB (left entry in each cell) and SSIM (right entry) and running time per measurement/shot in minutes by different algorithms on 6 benchmark color Bayer datasets. All codes are implemented in Python (Pytorch for deep denoising) except DeSCI, which is using MATLAB.

Algorithm	Beauty	Bosphorus	Jockey	Runner	ShakeNDry	Traffic	Average	Run time (min)
GAP-TV	33.08, 0.9639	29.70, 0.9144	29.48, 0.8874	29.10, 0.8780	29.59, 0.8928	19.84, 0.6448	28.47, 0.8636	0.18
DeSCI (GAP-WNNM)	34.66, **0.9711**	32.88, 0.9518	34.14, 0.9382	36.16, 0.9489	30.94, 0.9049	24.62, 0.8387	32.23, 0.9256	1544
PnP-FFDNet-gray	32.62, 0.9595	28.24, 0.8976	31.89, 0.9064	30.57, 0.8657	27.82, 0.8593	21.02, 0.7086	28.69, 0.8662	0.43 (GPU)
PnP-FFDNet-color	33.88, 0.9665	33.13, 0.9540	34.25, 0.9350	34.67, 0.9271	32.24, 0.9379	24.02, 0.8299	32.03, 0.9251	0.95 (GPU)
PnP-FastDVDnet-gray	32.60, 0.9597	30.74, 0.9292	32.93, 0.9160	32.43, 0.8764	29.80, 0.8964	22.64, 0.7658	30.19, 0.8906	0.87 (GPU)
PnP-FastDVDnet-color	**35.12**, 0.9709	**35.80**, **0.9701**	**35.14**, **0.9442**	**38.00**, **0.9618**	**33.35**, **0.9452**	**27.22**, **0.9085**	**34.11**, **0.9501**	0.95 (GPU)

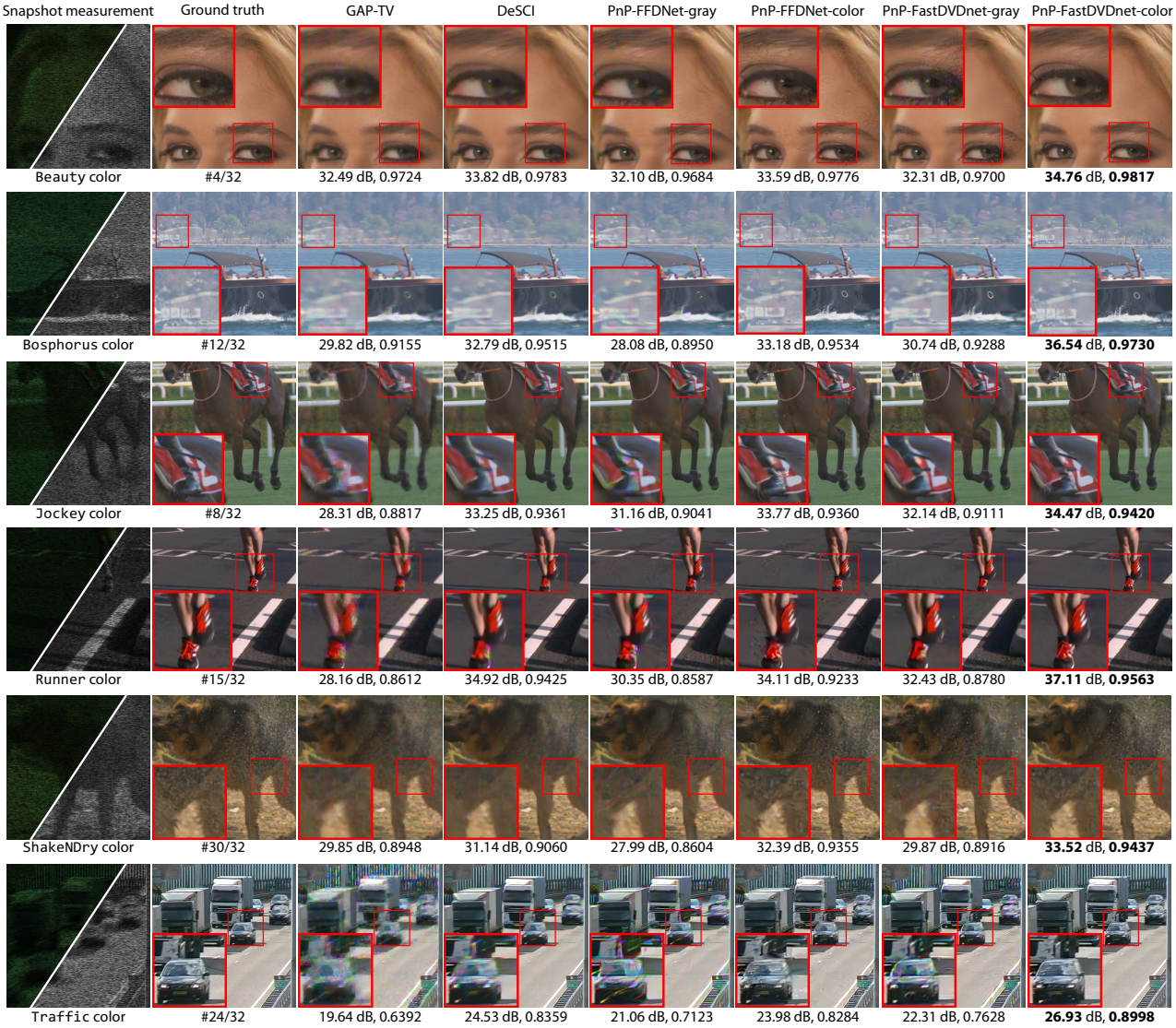


Fig. 6. Comparison of reconstructed frames of PnP-GAP algorithms (GAP-TV [23], DeSCI [25], PnP-FFDNet [50], and PnP-FastDVDnet) on six simulated benchmark color video SCI datasets of size $512 \times 512 \times 3$ and $B = 8$. Please refer to the full videos in the supplementary material.

highest PSNR, the difference between PnP-FastDVDNet and DeSCI is very small. By investigating the temporal correlation, PnP-FastDVDNet can provide finer details than PnP-FFDNet and sometimes DeSCI; please refer to the zoomed parts of `Aerial` in Fig. 5. We used the same parameter settings (described in Sec. 6.3) for all the datasets. Fine tuning the parameters for each dataset can lead to better results.

6.2 Benchmark Data: Color RGB-Bayer Videos

As mentioned before, in this paper, we propose the joint reconstruction and demosaicing using the PnP framework for color SCI shown in the lower part of Fig. 3. To verify the performance, we hereby generate a color RGB video dataset with 6 scenes of spatial size $512 \times 512 \times 3$, and here 3 denotes the RGB channels. Similar to the grayscale case, we used compression rate $B = 8$. The schematic of a color video SCI system is shown in Fig. 1. Every 8 consequent video frames are first interleaved to the mosaic frames of size $512 \times 512 \times 8$; then these mosaic frames are modulated by shifting binary masks of size $512 \times 512 \times 8$ and finally summed to get the compressed mosaic measurement of size 512×512. For each dataset, we have 4 compressed measurements and thus in total 32 RGB video frames. As shown in

Fig. 6, these datasets include `Beauty`, `Bosphorus`, `Jockey`, `ShakeNDry`[8], `Runner`[9] and `Traffic`[10]. In order to keep the video quality, we crop (instead of resize) the video frames with a spatial size of 512×512. We dub these datasets the 'mid-scale color data' due to the size in-between the small size grayscale benchmark data and the large-scale data discussed in the next subsection.

For other algorithms, we perform the reconstruction and demosaicing separately. The R, G1, G2, and B channels are reconstructed separately and then we employ the '`demosaic`' function in MATLAB to get the final RGB video. To verify the performance of this joint procedure, we compare the PnP-FFDnet/FastDVDnet using joint processing (color denoising) and separately reconstruction (grayscale denoising) shown in Fig. 3.

Table 2 summarizes the PSNR and SSIM results of these datasets. We have the following observations.

- i) When using color denoising for joint reconstruction and demosaicing, both PnP-FFDNet-color and PnP-FastDVDnet-color outperform DeSCI.
- ii) Color denoising significantly improves the results over grayscale denoising, *i.e.*, for FFDNet, the improvement is 3.34dB and for FastDVDnet, it is even 3.92dB in PSNR.
- iii) Regarding the running time, both PnP-FastDVDnet-color and PnP-FFDNet-color need about 1 minute per measurement, while they only need about 0.4 minutes for PnP-FFDNet-gray. Therefore, most time was consumed by demosaicing. As mentioned in Sec. 3.3, we hope deep learning based demosaicing will provide a fast and better result in the future.

Figure 6 plots selected reconstruction frames of different algorithms for these 6 RGB Bayer datasets with the snapshot measurement shown on the far left. Note that, due to the Bayer pattern of the sensor, the captured measurement is actually grayscale shown on the lower-right, whereas due to the coding in the imaging system, the demosaiced measurement depicts the wrong color shown in the upper-left part of the measurement. It can be seen from Fig. 6 that PnP-FastDVDnet-color and PnP-FFDNet-color are providing smooth motions and fine spatial details. Some color mismatch exists in the GAP-TV, DeSCI and PnP-FFDNet-gray and PnP-FastDVDnet-gray. For instance, in the `Traffic` data, the color of the cars is incorrectly reconstructed for these methods; similar case exists in the water drops in the `ShakeNDry`. Overall, GAP-TV provides blurry results and DeSCI sometimes over-smooths the background such as the lawn in the `Jockey` data. PnP-FastDVDnet-color provides the finest details in the complicated background such as the trees in `Bosphorus`. Similar to grayscale video SCI, hereby we used the same parameter settings (discussed in the following subsection) for all the datasets. Fine tuning the parameters for each dataset can lead to better results.

6.3 Hyper-parameter Setting

Recalling the proposed PnP-GAP algorithm in Algorithm 1, in practical applications, the only parameter that needs to be tuned is the noise level σ_k in each iteration. In Assumption 1, we impose that the σ_k is a non-increasing sequence. We thus tested 3 different settings of σ_k using the proposed algorithm. i) Set $\sigma_k = \sigma^{(1)}$ for K_1 iterations and then $\sigma_k = \sigma^{(2)}$ for K_2 iterations and then $\sigma_k = \sigma^{(3)}$ for K_3 iterations and so on and so forth, where $\sigma^{(1)} > \sigma^{(2)} > \sigma^{(3)} > \cdots > 0$; ii) set $\sigma_k = \sigma_0$ at the beginning and then multiply it by $\xi \in (0, 1)$ in the next iteration, *i.e.*, $\sigma_k = \xi\sigma_{k-1}$; iii) set $\sigma_k = \sigma^{(s)}$ (a small value) for K iterations, which usually needs a warm starting point. In [50], we used setting i), which leads to faster convergence results for the grayscale dataset than setting ii) with similar final results.

Regarding color video SCI, Fig. 8 plots the results of these 3 settings of σ_k using PnP-FastDVDnet-color in the mid-scale 'Beauty' dataset with the first measurement. It can be seen that: 1) when PnP-FastDVDnet-color is initialized by GAP-TV, all of them lead to similar results and converge fast (only need about 30 iterations) as shown in the right part of Fig. 8; 2) by setting σ to a small value, *i.e.*, setting iii) with $\sigma^{(s)}$ = 12, we can get almost monotonically increasing PSNR values and only 10 iterations (initialized by GAP-TV) can lead to excellent results; 3) by setting different σ values for a number of iterations such as the setting in i), we may get a higher PSNR but the PSNR curve drops when σ switches from one value to another; 4) when PnP-FastDVDnet-color is initialized by 0, only the setting of i) can lead to good results as shown in the left of Fig. 8. In summary, if the setting satisfies our non-increasing noise level assumption, PnP-FastDVDnet will lead to stable results (similar for PnP-FFDNet and others). Interestingly, the setting of iii) with a constant small σ will lead to a fast convergence when it is initialized by GAP-TV (even with a couple iterations).

6.4 Large-scale Data

Similar to the benchmark data, we simulate the color video SCI measurements for large-scale data with four YouTube slow-motion videos, *i.e.*, `Messi`[11], `Hummingbird`[12], `Swinger`[13], and `Football`[14]. A sequence of color scene is coded by the corresponding shifted random binary masks at each time step and finally summed up to form a snapshot measurement on the color Bayer RGB sensor (with a "RGGB" Bayer color filter array)[15]. To verify the flexibility of the proposed PnP algorithm, we consider different spatial size and different compression rate B.

8. `Beauty`, `Bosphorus`, `Jockey`, `ShakeNDry` are downloaded from http://ultravideo.cs.tut.fi/#testsequences.

9. Downloaded from https://www.videvo.net/video/elite-runner-slow-motion/4541.

10. Downloaded from http://dyntex.univ-lr.fr/database.html.

11. https://www.youtube.com/watch?v=sbPrevs6Pd4

12. https://www.youtube.com/watch?v=RtUQ_pz5wlo

13. https://www.youtube.com/watch?v=cfnbyX9G5Rk

14. https://www.youtube.com/watch?v=EGAuWZYe2No

15. Note these results are different from the ones reported in [50]. The reason is that the measurements are generally in different ways. In [50], we up-sampled the raw video by putting each color channel as the mosaic R, G1, G2, and B channels. This leads to two identical G channels and the reconstructed and the size of demosaiced image is doubled (both in width and height). For example, for UHD color video `Football` with original image size of 3840×1644, the reconstructed video frames have the size of 7680×3288 (demosaiced). This is different from the Bayer pattern model described in Sec. 3.3. After some researching on the camera design, in this paper, we follow the Bayer RGGB pattern color video SCI model developed in Sec. 2 to generate the new measurements being used in the experiments. We are convinced that this is more appropriate and closer to real color cameras.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

Fig. 7. Reconstructed frames of PnP-GAP algorithms (GAP-TV [23], PnP-FFDNet [50], and PnP-FastDVDnet) on four simulated large-scale video SCI datasets. Please refer to the full videos in the supplementary material.

TABLE 3
Running time (minutes) of large-scale data using different algorithms.

Large-scale dataset	Pixel resolution	GAP-TV	PnP-FFDNet-gray	PnP-FFDNet-color	PnP-FastDVDnet-gray	PnP-FastDVDnet-color
`Messi` color	$1920 \times 1080 \times 3 \times 20$	15.1	5.2	42.2	7.6	43.1
`Hummingbird` color	$1920 \times 1080 \times 3 \times 30$	20.3	6.6	61.2	10.6	54.0
`Swinger` color	$3840 \times 2160 \times 3 \times 15$	39.2	13.2	138.8	21.3	138.4
`Football` color	$3840 \times 1644 \times 3 \times 40$	83.0	30.6	308.8	50.7	298.1

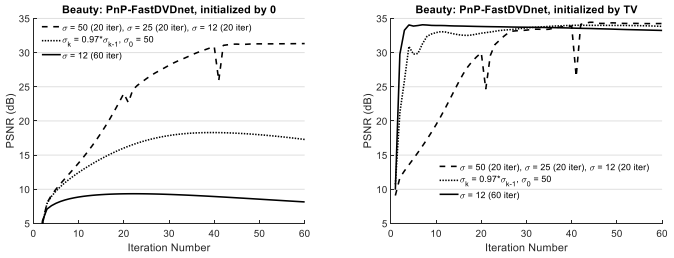


Fig. 8. Reconstruction quality by different settings of the noise level σ_k for the proposed PnP-FastDVDnet on the first measurement of the 'Beauty' data. Left: initialized by 0, right: initialized by GAP-TV. We assume the maximum pixel value being 255 and 'iter' in the ( ) of the legend denotes iterations.

- `Messi20` color: A $1920 \times 1080 \times 3 \times 20$ video reconstructed from a snapshot.
- `Hummingbird30` color: A $1920 \times 1080 \times 3 \times 30$ video reconstructed from a snapshot.
- `swinger15` color: A $3840 \times 2160 \times 3 \times 15$ video reconstructed from a snapshot.
- `football40` color: A $3840 \times 1644 \times 3 \times 40$ video reconstructed from a snapshot.

Due to the extremely long running time of other algorithms, we hereby only show the results of GAP-TV, PnP-FFDNet and PnP-FastDVDnet; both FFDNet and FastDVDnet used color denoising as in the mid-scale benchmark data; only grayscale FFDNet denoising was used in [50].

Figure 7 plots selected reconstruction frames of these three algorithms, where we can see that i) due to many fine details, GAP-TV cannot provide high quality results, ii) both PnP-FFDNet and PnP-FastDVDnet lead to significant improvements over GAP-TV (at least 3.69 dB in PSNR), and iii) PnP-FastDVDnet leads to best results on the first three datasets and for the last one, `football40`, it is 0.39dB lower than PnP-FFDNet. This might due to the crowded players in the scene, which is in favor of FFDNet denoising. Importantly, for all these large-scale dataset, with a compression rate varying from 15 to 40, we can all get the reconstruction up to (or at least close to) 30dB. This proves that the video SCI can be used in our daily life videos.

Regarding the running time, as shown in Table 3, for all these large-scale datasets, PnP with grayscale denoising (PnP-FFDNet-gray and PnP-FastDVDnet-gray) can finish the reconstruction within one hour. However, when the color denoising algorithms are used, the running time is $10\times$ longer. Again, as mentioned in the simulation, most of the time is consumed by the demosaicing algorithms and we expect a robust deep demosaicing network can speed up the reconstruction. Due to this, the running time of PnP-FastDVDnet-color is very similar to PnP-FFDNet-color. Even this, for the HD ($1920 \times 1080 \times 3$) video data with B up to 30, the reconstruction can be finished within 1 hour, but the other algorithms such as DeSCI are not feasible as it will take days. For the UHD ($3840 \times 1644 \times 3$) videos, even at $B = 40$, the reconstruction can be finished in hours. Note that since spatial pixels in video SCI are decoupled, we can also use multiple CPUs or GPUs performing on blocks rather than the entire frame to speed up the reconstruction.

In addition to these large-scale data with different spatial sizes and compression rates, another way to construct large-scale data for a specific SCI system is to fix the spatial size,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

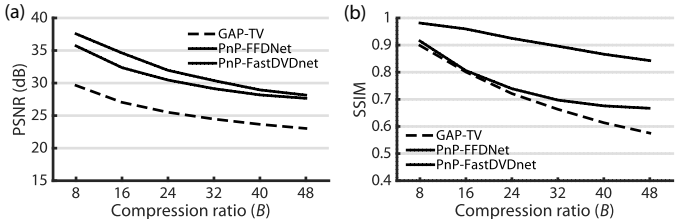
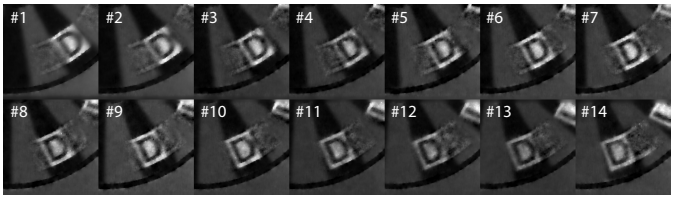
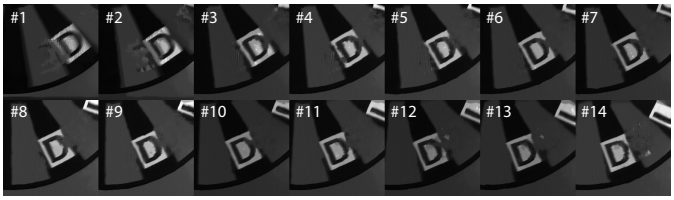


Fig. 9. Reconstruction PSNR in dB (a) and SSIM (b), varying compression rates B from 8 to 48 of the proposed PnP methods (PnP-FFDNet and PnP-FastDVDnet) and GAP-TV [23] .

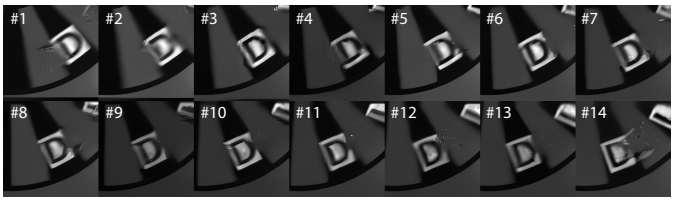
but with various compression rates. In this case, the data scales with B, which is also challenging to other algorithms including deep learning ones[16]. Hereby, we conduct simulation of the `Hummingbird` data with different compression rates $B = 8, 16, 24, 32, 40, 48$ with results shown in Fig. 9. It can be seen that even at B=48, both PnP-FFDNet and PnP-FastDVDnet can reconstruct the video at PSNR close to 27dB; regarding SSIM, PnP-FastDVDnet achieves 0.85 at B=48, which is >0.15 higher than PnP-FFDNet and >0.25 higher than GAP-TV. Therefore, our proposed PnP algorithms are robust and flexible to different compression rates. This will be further verified by the real data in Sec. 7.2.



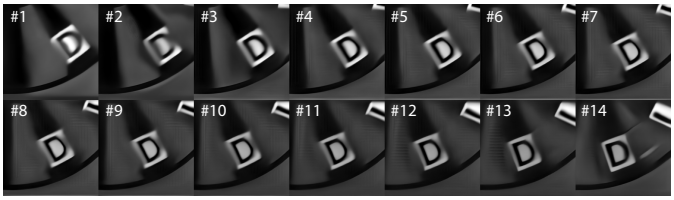
(a) GAP-TV



(c) DeSCI



(c) PnP-FFDNet



(d) PnP-FastDVDnet

Fig. 10. Real data: `chopper wheel` ($256 \times 256 \times 14$).

7 Real Data

We now apply the proposed PnP framework to real data captured by SCI cameras to verify the robustness of the proposed algorithms. Different data captured by different video SCI cameras are used and these data are of different spatial size, different compression rate and using different modulation patterns. We first verify PnP by using grayscale data [7], [17] with a fixed B; then we conduct the experiments of grayscale data with different compression rates captured by the same system [18]. Lastly, we show the results of color data captured by the SCI system in [8]. Note that, in Sec. 7.2, for the first time, we show that a $512 \times 512 \times 50$ `Hand` video reconstructed from a snapshot in high quality with each frame having a motion.

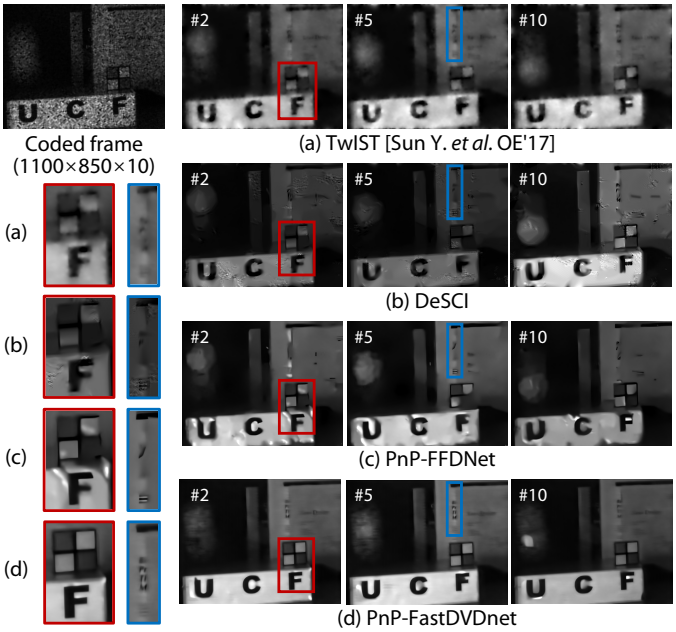


Fig. 11. Real data: `UCF` high-speed video SCI ($1100 \times 850 \times 10$).

TABLE 4
Running time (seconds) of real data using different algorithms.

Real dataset	Pixel resolution	GAP-TV	DeSCI	PnP-FFDNet	PnP-FastDVDnet
`chopperwheel`	$256 \times 256 \times 14$	11.6	3185.8	**2.7**	18.3
`hammer color`	$512 \times 512 \times 22$	94.5	4791.0	**12.6**	136.6
`UCF`	$1100 \times 850 \times 10$	300.8	2938.8	**12.5**	132.6
`hand10`	$512 \times 512 \times 10$	37.8	2880.0	**19.3**	29.5
`hand20`	$512 \times 512 \times 20$	88.7	4320.0	**42.4**	63.9
`hand30`	$512 \times 512 \times 30$	163.0	6120.0	**74.7**	107.7
`hand50`	$512 \times 512 \times 50$	303.4	12600.0	**144.5**	203.9

7.1 Grayscale Videos with Fixed Compression Rate

In this subsection, we verify the proposed PnP algorithm by the following data:

- `Chopper wheel` data captured by the original CACTI paper [7] is of spatial size 256×256 and $B = 14$. The results of GAP-TV, DeSCI, PnP-FFDNet and PnP-FastDVDnet are shown in Fig. 10, where we can see that though DeSCI, PnP-FFDNet and PnP-FastDVDnet can all provide good results. Due to the temporal correlation of video investigated in FastDVDnet, the results of PnP-FastDVDnet is having a consistent brightness and of smooth motion.
- `UCF data` captured by the video SCI system built in [17] is of large size 1100×850 with $B = 10$. The results are shown in Fig. 11, which has a complicated background and a dropping ball on the left. It can be seen clearly that PnP-FastDVDnet provides a clean background with fine details.

16. In [16], it is failed to train a deep neural networks for $B > 30$ even with a spatial size 512×512 due to the limited GPU memory.

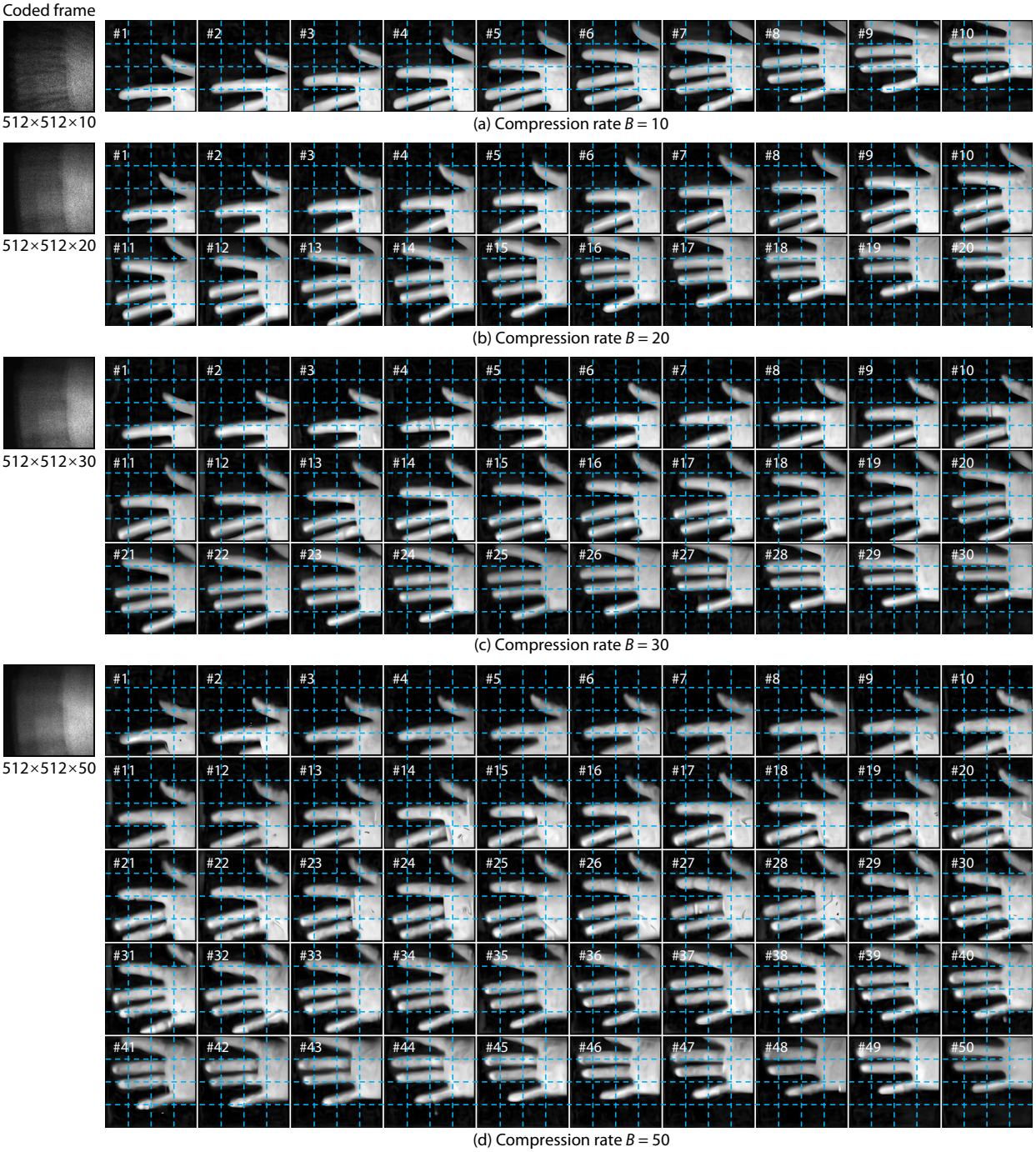


Fig. 12. Real data: `Hand` high-speed video SCI ($512 \times 512 \times B$) with compression rates, B, vary from 10 to 50. Dashed grids are added to aid the visualization of motion details. PnP-FastDVDnet is used for the reconstruction.

Again, since these data are of different sizes and compression rates, it is challenging to use the recently developed end-to-end deep neural networks [28] to perform all the tasks. For instance, the training time for each task will be of weeks and it consumes a significant amount of power and memory to train the network for large-scale data such as `UCF`. By contrast, in our proposed PnP framework, the same pre-trained FFDNet or FastDVDnet is used for all these tasks and the results are obtained in seconds.

The running time of different algorithms for these real data are shown in Table 4. We can see that PnP-FFDNet, which only takes a few seconds for the reconstruction of these grayscale datasets, can provide comparable results as DeSCI, which needs hours even when performed in a frame-wise manner. PnP-FFDNet is significantly better than the speed runner-up GAP-TV (for the top two datasets) in terms of motion-blur reduction and detail preservation, as shown in Figs. 10 and 11. PnP-FFDNet is at least more than $4\times$ faster than GAP-TV and when the data size is getting larger, the running time increases slower than GAP-TV. In this way, PnP algorithms for SCI achieves a good balance of efficiency and flexibility and PnP-FFDNet could serve as a baseline for SCI recovery.

PnP-FastDVDnet costs about $10\times$ longer than PnP-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3099035, IEEE Transactions on Pattern Analysis and Machine Intelligence

FFDNet (upper part in Table 4), which is the price for a higher reconstruction quality. We also notice that the running time of PnP-FastDVDnet for large-scale data `UCF` is shorter than GAP-TV. This shows another gain of PnP based algorithm, *i.e.*, ready to scale up. This will be further verified by the following `Hand` data. Therefore, we recommend the PnP-FFDNet as a new baseline, and if a higher quality result is desired, PnP-FastDVDnet is a good choice with a longer running time (but still $20\times$ shorter than DeSCI).

7.2 Grayscale Videos with Various Compression Rates

Next, we test the PnP algorithms by the data captured by a recently built video SCI system in [16], where similar scenes were captured by different compression rates, *i.e.*, $B = \{10, 20, 30, 40, 50\}$[17]. Unlike the data reported in the previous subsection, here the data are of the same spatial size 512×512, but a compression rate of 50 will need a significant amount of GPU memory to train a deep neural network for the inversion. As shown in Fig. 9, this is another way to construct the large-scale data. Another challenge in video SCI is that though high compression rate results have been reported before, whether the reconstructed video can resolve such a high-speed motion is still a question.

To address these concerns, hereby, we show the reconstruction of `Hand` data with $B = 10, 20, 30, 50$ in Fig. 12, where we can see that at $B = 50$, each frame is different from the previous one, and this leads to a high-speed motion to at least a few pixels motion per reconstructed frame. Regarding the running time, it can be seen from Table 4 that both PnP-FFDNet and PnP-FastDVDnet are faster than GAP-TV. At $B = 50$, PnP-FFDNet can finish the reconstruction of one measurement in 2.4 minutes and PnP-FastDVDnet needs 3.4 minutes but providing better results. By contrast, DeSCI needs 210 minutes (3.5 hours) to reconstruct 50 frames from a snapshot.

We have also tried to reconstruct this video by training the networks proposed in [16] and [28]; however, the quality of the results from the trained networks for this `Hand` dataset are poor when $B > 20$, mainly due to the high-speed motions. By contrast, in Fig. 12, we can see clear details are reconstructed by the proposed PnP-FastDVDnet. Therefore, it is confident to state that the video SCI system along with our proposed PnP algorithm can achieve a compression rate of 50. When the camera is working at 50 fps [18], the built system can be used to capture high-speed videos at 2500 fps with high quality reconstruction. Due to the space limit, we do not show results of other data here and more results can be found in the supplementary videos.

7.3 Color Videos

Lastly, we verify the proposed algorithm on the color video SCI data captured by [8], which has the same model as described in Section 3.3. Following the procedure in the mid-scale color data, an RGB video of $B = 22$ frames with size of $512 \times 512 \times 3$ is reconstructed from a single Bayer mosaic measurement shown in Fig. 13 of the data `hammer`. Along with the running time in Table 4, we can see that

17. Data at: https://github.com/mq0829/DL-CACTI.

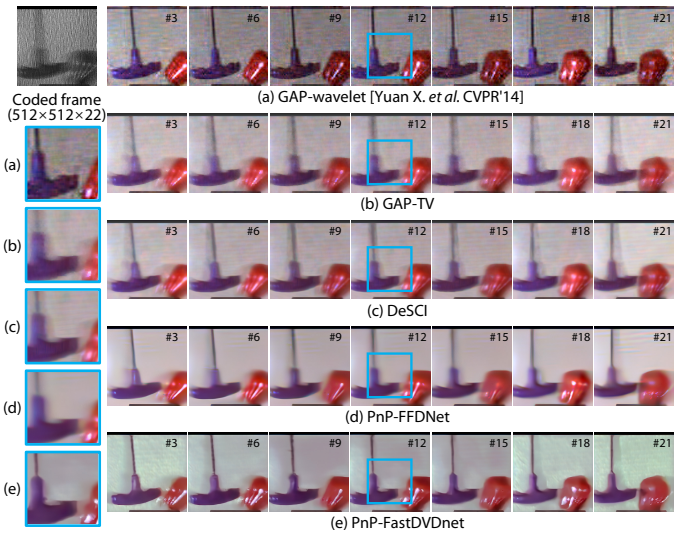


Fig. 13. Real data: `Hammer` color video SCI ($512 \times 512 \times 3 \times 22$).

PnP-FFDnet, which only takes about 12 seconds for reconstruction, can provide comparable results as DeSCI, which needs hours. GAP-wavelet [8] cannot remove noise in the background and GAP-TV shows blurry results. PnP-FFDnet shows sharper edges than DeSCI with a clean background. PnP-FastDVDnet reconstructs sharper boundaries and finer details of the hammer than DeSCI and PnP-FFDNet, but needs 136 seconds (2.27 minutes) for the reconstruction. We do notice the greenish background of PnP-FastDVDnet, which may come from the smoothing artifacts of brightness across different frames. We will test more color video SCI data using the proposed PnP algorithms in the future.

8 Conclusions

We proposed plug-and-play algorithms for the reconstruction of snapshot compressive video imaging systems. By integrating deep denoisers into the PnP framework, we not only get excellent results on both simulation and real datasets, but also provide reconstruction in a short time with sufficient flexibility. Convergence results of PnP-GAP are proved and we first time show that SCI can be used in large-scale (HD, FHD and UHD) daily life videos. This paves the way of practical applications of SCI.

Regarding the future work, one direction is to incorporating an efficient demosaicing network to speed up the reconstruction and also improve the video quality. The other direction is to build a real large-scale video SCI system to be used in advanced cameras [58].

Acknowledgments.

The work of Jinli Suo and Qionghai Dai is partially supported by NSFC 61722110, 61931012, 61631009 and Beijing Municipal Science & Technology Commission (BMSTC) (No. Z181100003118014). X. Yuan and Y. Liu contribute equally to this paper.

References

[1] Y. Altmann, S. McLaughlin, M. J. Padgett, V. K. Goyal, A. O. Hero, and D. Faccio, "Quantum-inspired computational imaging," *Science*, vol. 361, no. 6403, 2018.

[2] J. N. Mait, G. W. Euliss, and R. A. Athale, "Computational imaging," *Adv. Opt. Photon.*, vol. 10, no. 2, pp. 409–483, Jun 2018.

[3] D. J. Brady, W. Pang, H. Li, Z. Ma, Y. Tao, and X. Cao, "Parallel cameras," *Optica*, vol. 5, no. 2, 2018.

[4] D. J. Brady, A. Mrozack, K. MacCabe, and P. Llull, "Compressive tomography," *Advances in Optics and Photonics*, vol. 7, no. 4, p. 756, 2015.

[5] T.-H. Tsai, P. Llull, X. Yuan, L. Carin, and D. J. Brady, "Spectral-temporal compressive imaging," *Optics Letters*, vol. 40, no. 17, pp. 4054–4057, Sep 2015.

[6] X. Yuan, D. J. Brady, and A. K. Katsaggelos, "Snapshot compressive imaging: Theory, algorithms, and applications," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 65–88, 2021.

[7] P. Llull, X. Liao, X. Yuan, J. Yang, D. Kittle, L. Carin, G. Sapiro, and D. J. Brady, "Coded aperture compressive temporal imaging," *Optics Express*, vol. 21, no. 9, pp. 10 526–10 545, 2013.

[8] X. Yuan, 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)*, 2014, Journal Article, pp. 3318–3325.

[9] A. A. Wagadarikar, N. P. Pitsianis, X. Sun, and D. J. Brady, "Video rate spectral imaging using a coded aperture snapshot spectral imager," *Optics Express*, vol. 17, no. 8, pp. 6368–6388, 2009.

[10] 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.

[11] D. Reddy, A. Veeraraghavan, and R. Chellappa, "P2C2: Programmable pixel compressive camera for high speed imaging," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Conference Proceedings, pp. 329–336.

[12] X. Yuan and S. Pang, "Structured illumination temporal compressive microscopy," *Biomedical Optics Express*, vol. 7, pp. 746–758, 2016.

[13] C. Deng, Y. Zhang, Y. Mao, J. Fan, J. Suo, Z. Zhang, and Q. Dai, "Sinusoidal sampling enhanced compressive camera for high speed imaging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2019.

[14] X. Miao, X. Yuan, Y. Pu, and V. Athitsos, "λ-net: Reconstruct hyperspectral images from a snapshot measurement," in *IEEE/CVF Conference on Computer Vision (ICCV)*, 2019.

[15] X. Yuan, T.-H. Tsai, R. Zhu, P. Llull, D. Brady, and L. Carin, "Compressive hyperspectral imaging with side information," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 6, pp. 964–976, September 2015.

[16] M. Qiao, Z. Meng, J. Ma, and X. Yuan, *APL Photonics*, vol. 5, no. 3, p. 030801, 2020.

[17] Y. Sun, X. Yuan, and S. Pang, "Compressive high-speed stereo imaging," *Opt Express*, vol. 25, no. 15, pp. 18 182–18 190, 2017.

[18] M. Qiao, X. Liu, and X. Yuan, "Snapshot spatial-temporal compressive imaging," *Opt. Lett.*, 2020.

[19] ——, "Snapshot temporal compressive microscopy using an iterative algorithm with untrained neural networks," *Opt. Lett.*, 2021.

[20] J. M. Bioucas-Dias and M. A. Figueiredo, "A new twist: Two-step iterative shrinkage/thresholding algorithms for image restoration," *IEEE Transactions on Image processing*, vol. 16, no. 12, pp. 2992–3004, 2007.

[21] 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 Transaction on Image Processing*, vol. 24, no. 1, pp. 106–119, January 2015.

[22] J. Yang, X. Yuan, X. Liao, P. Llull, G. Sapiro, D. J. Brady, and L. Carin, "Video compressive sensing using Gaussian mixture models," *IEEE Transaction on Image Processing*, vol. 23, no. 11, pp. 4863–4878, November 2014.

[23] X. Yuan, "Generalized alternating projection based total variation minimization for compressive sensing," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 2539–2543.

[24] X. Liao, H. Li, and L. Carin, "Generalized alternating projection for weighted-$\ell_{2,1}$ minimization with applications to model-based compressive sensing," *SIAM Journal on Imaging Sciences*, vol. 7, no. 2, pp. 797–823, 2014.

[25] Y. Liu, X. Yuan, J. Suo, D. Brady, and Q. Dai, "Rank minimization for snapshot compressive imaging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 12, pp. 2990–3006, Dec 2019.

[26] J. Ma, X. Liu, Z. Shou, and X. Yuan, "Deep tensor admm-net for snapshot compressive imaging," in *IEEE/CVF Conference on Computer Vision (ICCV)*, 2019.

[27] Y. Li, M. Qi, R. Gulve, M. Wei, R. Genov, K. N. Kutulakos, and W. Heidrich, "End-to-end video compressive sensing using anderson-accelerated unrolled networks," in *2020 IEEE International Conference on Computational Photography (ICCP)*, 2020, pp. 1–12.

[28] Z. Cheng, R. Lu, Z. Wang, H. Zhang, B. Chen, Z. Meng, and X. Yuan, "Birnat: Bidirectional recurrent neural networks with adversarial training for video snapshot compressive imaging," in *European Conference on Computer Vision (ECCV)*, August 2020.

[29] X. Yuan, J. Yang, P. Llull, X. Liao, G. Sapiro, D. J. Brady, and L. Carin, "Adaptive temporal compressive sensing for video," *IEEE International Conference on Image Processing*, pp. 1–4, 2013.

[30] L. Wang, Z. Xiong, H. Huang, G. Shi, F. Wu, and W. Zeng, "High-speed hyperspectral video acquisition by combining nyquist and compressive sampling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 4, pp. 857–870, April 2019.

[31] Z. Meng, M. Qiao, J. Ma, Z. Yu, K. Xu, and X. Yuan, "Snapshot multispectral endomicroscopy," *Opt. Lett.*, vol. 45, no. 14, pp. 3897–3900, Jul 2020.

[32] Z. Meng, J. Ma, and X. Yuan, "End-to-end low cost compressive spectral imaging with spatial-spectral self-attention," in *European Conference on Computer Vision (ECCV)*, August 2020.

[33] P. Llull, X. Yuan, L. Carin, and D. J. Brady, "Image translation for single-shot focal tomography," *Optica*, vol. 2, no. 9, pp. 822–825, 2015.

[34] X. Yuan, X. Liao, P. Llull, D. Brady, and L. Carin, "Efficient patch-based approach for compressive depth imaging," *Applied Optics*, vol. 55, no. 27, pp. 7556–7564, Sep 2016.

[35] T.-H. Tsai, X. Yuan, and D. J. Brady, "Spatial light modulator based color polarization imaging," *Optics Express*, vol. 23, no. 9, pp. 11 912–11 926, May 2015.

[36] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, April 2006.

[37] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[38] 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)*, 2014, pp. 2862–2869.

[39] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, January 2011.

[40] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4509–4522, Sept 2017.

[41] X. Yuan and Y. Pu, "Parallel lensless compressive imaging via deep convolutional neural networks," *Optics Express*, vol. 26, no. 2, pp. 1962–1977, Jan 2018.

[42] M. Yoshida, A. Torii, M. Okutomi, K. Endo, Y. Sugiyama, R.-i. Taniguchi, and H. Nagahara, "Joint optimization for compressive video sensing and reconstruction under hardware constraints," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[43] Z. Cheng, B. Chen, G. Liu, H. Zhang, R. Lu, Z. Wang, and X. Yuan, "Memory-efficient network for large-scale video compressive sensing," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[44] T. Huang, W. Dong, X. Yuan, J. Wu, , and G. Shi, "Deep gaussian scale mixture prior for spectral compressive imaging," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[45] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *2013 IEEE Global Conference on Signal and Information Processing*, 2013, pp. 945–948.

[46] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, G. T. Buzzard, L. F. Drummy, J. P. Simmons, and C. A. Bouman, "Plug-and-play priors for bright field electron tomography and sparse interpolation," *IEEE Transactions on Computational Imaging*, vol. 2, no. 4, pp. 408–423, 2016.

[47] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-play ADMM for image restoration: Fixed-point convergence and applications," *IEEE Transactions on Computational Imaging*, vol. 3, pp. 84–98, 2017.

[48] E. K. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin, "Plug-and-play methods provably converge with properly trained denoisers," in *ICML*, 2019.

[49] L. Zhang and W. Zuo, "Image restoration: From sparse and low-rank priors to deep priors [lecture notes]," *IEEE Signal Processing Magazine*, vol. 34, no. 5, pp. 172–179, 2017.

[50] X. Yuan, Y. Liu, J. Suo, and Q. Dai, "Plug-and-play algorithms for large-scale snapshot compressive imaging," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[51] S. Jalali and X. Yuan, "Snapshot compressed sensing: Performance bounds and algorithms," *IEEE Transactions on Information Theory*, vol. 65, no. 12, pp. 8005–8024, Dec 2019.

[52] K. Zhang, W. Zuo, and L. Zhang, "FFDNet: Toward a fast and flexible solution for CNN-based image denoising." *IEEE Trans. Image Processing*, vol. 27, no. 9, pp. 4608–4622, 2018. [Online]. Available: http://dblp.uni-trier.de/db/journals/tip/tip27.html#ZhangZZ18

[53] M. Tassano, J. Delon, and T. Veit, "Fastdvdnet: Towards real-time deep video denoising without flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[54] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE signal processing magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[55] S. Jalali and X. Yuan, "Compressive imaging via one-shot measurements," in *IEEE International Symposium on Information Theory (ISIT)*, 2018.

[56] G. T. Buzzard, S. H. Chan, S. Sreehari, and C. A. Bouman, "Plug-and-play unplugged: Optimization-free reconstruction using consensus equilibrium," *SIAM Journal on Imaging Sciences*, vol. 11, no. 3, pp. 2001–2020, 2018.

[57] X. Li, B. Gunturk, and L. Zhang, "Image demosaicing: a systematic survey," in *Visual Communications and Image Processing 2008*, vol. 6822. SPIE, 2008, pp. 489 – 503.

[58] D. J. Brady, L. Fang, and Z. Ma, "Deep learning for camera data acquisition, control, and image estimation," *Adv. Opt. Photon.*, vol. 12, no. 4, pp. 787–846, Dec 2020.

[59] R. Malvar, L.-w. He, and R. Cutler, "High-quality linear interpolation for demosaicing of bayer-patterned color images," in *International Conference of Acoustic, Speech and Signal Processing*, May 2004.

[60] D. Menon, S. Andriani, and G. Calvagno, "Demosaicing with directional filtering and a posteriori decision," *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 132–141, 2007.

[61] Z. Zha, X. Yuan, B. Wen, J. Zhou, J. Zhang, and C. Zhu, "A benchmark for sparse coding: When group sparsity meets rank minimization," *IEEE Transactions on Image Processing*, vol. 29, pp. 5094–5109, 2020.

[62] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Img. Sci.*, vol. 2, no. 1, pp. 183–202, Mar. 2009.

[63] S. Zheng, Y. Liu, Z. Meng, M. Qiao, Z. Tong, X. Yang, S. Han, and X. Yuan, "Deep plug-and-play priors for spectral snapshot compressive imaging," *Photonics Research*, vol. 9, Jan 2021.

[64] X. Yuan, "Various total variation for snapshot video compressive imaging," *arXiv: 2005.08028*, May 2020.

[65] Z. Zha, X. Yuan, B. Wen, J. Zhang, J. Zhou, and C. Zhu, "Image restoration using joint patch-group-based sparse representation," *IEEE Transactions on Image Processing*, vol. 29, pp. 7735–7750, 2020.

[66] S. Yang, J. Wang, W. Fan, X. Zhang, P. Wonka, and J. Ye, "An efficient admm algorithm for multidimensional anisotropic total variation regularization problems," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, pp. 641–649.

[67] P. Yang, L. Kong, X. Liu, X. Yuan, and G. Chen, "Shearlet enhanced snapshot compressive imaging," *IEEE Transactions on Image Processing*, vol. 29, pp. 6466–6481, 2020.

[68] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[69] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 2272–2279.

[70] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, August 2007.

[71] M. Maggioni, G. Boracchi, A. Foi, and K. O. Egiazarian, "Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms," *IEEE Transactions on Image Processing*, vol. 21, pp. 3952–3966, 2012.

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

Xin Yuan (SM'16) obtained his B.Eng and M.Eng from Xidian University in 2007 and 2009, respectively, and his Ph.D. from the Hong Kong Polytechnic University in 2012. He had been a Post-Doctoral Associate with the Department of Electrical and Computer Engineering, Duke University from 2012 to 2015 and a video analysis and coding lead researcher at Bell Labs, Murray Hill, NJ, USA from 2015 to 2021. He joins Westlake University in 2021. His research interests include computational imaging, signal processing and machine learning.

Yang Liu received the B.S. degree in automation and M.S. degree in control theory and engineering in the Department of Automation from Tsinghua University, Beijing, China in 2016 and 2019, respectively. He is currently pursuing the PhD in electrical engineering and computer science at the Massachusetts Institute of Technology. His research interests include computational imaging and photography.

Jinli Suo received the B.S. degree in computer science from Shandong University, Shandong, China, in 2004 and the Ph.D. degree from the Graduate University of Chinese Academy of Sciences, Beijing, China, in 2010. She is currently an associate professor with the Department of Automation, Tsinghua University, Beijing, China. Her research interests include computer vision, computational photography, and statistical learning.

Frédo Durand received the PhD degree from Grenoble University, France, in 1999, supervised by Claude Puech and George Drettakis. He is the Amar Bose Professor of Computing in electrical engineering and computer science department with the Massachusetts Institute of Technology, and a member of the Computer Science and Artificial Intelligence Laboratory (CSAIL). From 1999 till 2002, he was a post-doc in the MIT Computer Graphics Group with Julie Dorsey.

Qionghai Dai (SM'05) received the M.E. and Ph.D. degrees in computer science and automation from Northeastern University, Shenyang, China, in 1994 and 1996, respectively. He has been the Faculty Member of Tsinghua University since 1997. He is currently a professor with the Department of Automation, Tsinghua University, Beijing, China, and the director of the Broadband and Digital Media Laboratory. His research areas include computational photography and microscopy, computer vision and graphics, and video communication. He is Associate Editor of the JVCI, the IEEE TNNLS, and the IEEE TIP. He is a senior member of the IEEE. Prof. Dai is a member of the Chinese Academy of Engineering.