Scalable Coding for High-Resolution, High-Compression Ratio Snapshot Compressive Video

Felipe Guzmán, Nelson Díaz, *Member, IEEE*, Bastian Romero and Esteban Vera, *Senior Member, IEEE*

***Abstract*—High-speed cameras are crucial for capturing fast events beyond human perception, although challenges in terms of storage, bandwidth, and cost hinder their widespread use. As an alternative, snapshot compressive video can overcome these challenges by exploiting the principles of compressed sensing to capture compressive projections of dynamic scenes into a single image, which is then used to recover the underlying video by solving an ill-posed inverse problem. However, scalability in terms of spatial and temporal resolution is limited for both acquisition and reconstruction. In this work, we leverage time-division multiplexing to design a versatile scalable coded aperture approach that allows unseen spatio-temporal scalability for snapshot compressive video, offering on-the-fly, high-compression ratios with minimal computational burden and low memory requirements. The proposed sampling scheme is universal and compatible with any compressive temporal imaging sampling matrices and reconstruction algorithm aimed for low spatio-temporal resolutions. Simulations validated with a series of experimental results confirm that we can compress up to 512 frames of 2K $\times$ 2K resolution into a single snapshot, equivalent to a compression ratio of 0.2%, delivering an overall reconstruction quality exceeding 30 dB in PSNR for conventional reconstruction algorithms, and often surpassing 36 dB when utilizing the latest state-of-the-art deep learning reconstruction algorithms. The results presented in this paper can be reproduced in the following GitHub repository: https://github.com/FOGuzman/All-scalable-CACTI.**

***Index Terms*—High-compressive video, High-resolution video, Compressed sensing, Snapshot compressive video, Scalable coded aperture.**

I. Introduction

HIGH-SPEED video serves as a remarkable tool for analyzing highly dynamic events, finding applications in a myriad of fields such as robotics [1], [2], autonomous driving [3], fluid dynamics [4], astronomy [5], and medicine [6]. Nevertheless, conventional high-speed cameras are costly and, more than often, sacrifice spatial resolution. Moreover, they require massive bandwidth and storage capabilities to handle the acquired high-speed videos.

In contrast, snapshot compressive video (SCV) [7] arises as an alternative to conventional video sampling, which exploits compressed sensing (CS) methodology to capture a single projection of high-dimensional spatio-temporal data by modulating video frames of a datacube using a dynamic coded aperture (CA). For instance, in the coded aperture compressive temporal imaging (CACTI) system [8], multiple frames are optically modulated using a shifting random mask or a digital micromirror device (DMD), and then integrated into a single snapshot during exposure time. The underlying video is recovered using sophisticated reconstruction algorithms that receive a single optically compressed image [9]–[12] as input. Nowadays, decompress snapshot compressive imaging (DeSCI) [13] is considered the state-of-the-art in terms of iterative recovery algorithms, at the expense of a high computational burden.

To accelerate the compressive reconstruction process, deep learning (DL) [14] has emerged as an alternative for achieving a significantly higher inference speed [7], [15]–[18]. This approach also delivers exceptional reconstruction quality; however, it lacks generalization to unseen masks during training or changes in data dimensionality such as the number of masks or their spatial resolution. Furthermore, the training process is extensive and computationally intensive. Some later works combine iterative algorithms with DL-based inference stages, by incorporating a denoiser [19] or through iterations of unrolling networks [20].

Although state-of-the-art reconstruction methods produce high-quality videos, they also experience a significant loss of quality when trying higher temporal multiplexing ($\geq$16 frames). Recently, DL methods have been used to address large-scale SCV challenges. For example, Cheng *et al.* [21] proposed a decoupled network, which does not save all gradients during backpropagation, allowing training larger models with lower video memory requirements. When using denoiser networks, Yuan *et al.* [22], [23] managed to recover a $3840\times1644\times48$ color video from a single measurement with a peak signal-to-noise ratio (PSNR) above 30 dB. Nevertheless, these neural networks are constrained by a fixed input size and require a substantial amount of VRAM. Moreover, loading only 50 frames of a 3840×1644 (4K) color video along with the DL model implies a memory usage exceeding 10 GB, which eliminates the possibility of using these methods on low-power, cheaper computers. This issue becomes more challenging as the resolution increases, for instance, in 8K video. For example, Wang *et al.* [17] proposed a DL model that reconstructs high-dimensional video into a series of spatial blocks–in the spirit of traditional image compression such as JPEG–while also requiring mask generalization within existing trained DL-based methods. In addition, temporal scalability

This work was supported by Fondo Nacional de Desarrollo Científico y Tecnológico (FONDECYT 1221883); Agencia Nacional de Investigación y Desarrollo (ANILLO ATE220022, DOCTORADO 2024-2124559).

Felipe Guzmán, Bastian Romero, Nelson Díaz and Esteban Vera are with the School of Electrical Engineering, Pontificia Universidad Católica de Valparaíso, Valparaiso 2340000, Chile (e-mail: felipe.guzman@pucv.cl; nelson.diaz@pucv.cl; bastian.romero@pucv.cl; esteban.vera@pucv.cl)

has not been addressed in most of these studies because measurements are only spatially decoupled.

A simple yet effective way to temporarily decouple the measurements is to divide the sensor pixels to expose them to the scene at different times. For instance, the approaches in [24] and [25] used a temporal mosaic technique to generate high-speed videos by decimating the exposed pixels in each frame. Another design presented in [26] used a dual-axis galvanometer to divide the sensor into sectors by moving the light to different regions within the image plane and multiplexing different instants during a single integration cycle of the sensor, but this method sacrifices the spatial resolution. To further expand the idea of time-division multiplexing, we propose a scalable coded aperture (SCA) design approach that uses a temporal mosaic of low resolution SCV sampling schemes. This involves sensor decimation not only to differentiate spatial positions at different times but also to integrate multiple frames within specific temporal intervals using existing compressive coding strategies such as in the CACTI. Thus, our SCA method uses the divide-and-conquer tactic by segmenting the reconstruction task into smaller manageable datacubes that can be efficiently recovered through conventional compressive reconstruction techniques or pretrained DL algorithms, and then they can be combined during an upscaling stage. This work makes the following contributions:

1) The introduction of a mathematical framework to design SCAs for high-resolution snapshot compressive imaging at high compression ratios, merging time-division multiplexing and conventional compressive temporal sampling matrices.
2) The description of a reconstruction strategy that exploits the designed SCAs to recover the underlying video. By using a divide and conquer approach, we recover small datacubes using off-the-shelf CS recovery algorithms, which are later interpolated into a high-resolution video using tensor completion or superresolution.
3) The presentation of a comprehensive series of simulations and experimental evaluations using the proposed reconstruction techniques to assess the efficacy of the SCA design for ultra scalable SCV.

This article is organized as follows: Section II describes the acquisition model for SCV and the design of the SCA. Section III corresponds to the reconstruction process from the measurement to a high-dimensional datacube. Section IV presents simulation results with our SCA design. Section V discusses the experimental results performed in hardware, and the conclusions are summarized in Section VI

II. Forward model

Mathematically, the snapshot compressive measurement obtained with the CACTI system is described as

$$Y_{i,j} = \sum_{t=1}^{T} \mathcal{X}_{i,j,t} \odot \mathcal{C}_{i,j,t} + \Omega_{i,j}, \tag{1}$$

where $\mathcal{X} \in \mathbb{R}^{M\times N\times T}$ is the spatiotemporal datacube with a spatial resolution of $M \times N$ and T is the number of frames. The subscript t denotes the t^{th} frame of $\mathcal{X}$ at the discrete spatial positions (i, j), $\mathcal{C} \in \{0, 1\}^{M\times N\times T}$ is a binary cube that describes the spatiotemporal CA, $\odot$ is the Hadamard product, and $\mathbf{\Omega} \in \mathbb{R}^{M\times N}$ is a Gaussian readout noise component of the measurement system.

Conventional approaches to designing a CA can be divided into two types: multiplexing and subsampling methods. A multiplexing CA often uses a 50% random Bernoulli distribution for the light transmittance (Tr), *i.e.*, the amount of light that passes through the CA, as shown in Fig. 1(a). Although this method enables the compression of B frames per measurement, it comes at the cost of a more challenging-compressive inference-inverse problem. In contrast, subsampling works by capturing selective pixels from the datacube without any multiplexing, as shown in the example depicted in Fig. 1(b) [27]. Although this time frame reconstruction can be achieved by simple interpolation, it comes at the expense of a potential loss of spatial information proportional to the aimed extension of temporal resolution. On the other hand, our proposed SCA method, illustrated in Fig. 1(c), merges multiplexing and subsampling strategies to enhance the frame count and decouple measurements temporally. Note that in traditional SCI architectures T is directly defined as the desired compression ratio. However, since in our proposed method the CA $\mathcal{C}$ is parametrized, now T also depends on the number of groups K and the multiplexing factor B.

A. Scalable coded aperture design method

To construct the proposed SCA that allows video multiplexing but also decouples frame groups, we first divide the sensor into sectors defined by a temporal mosaic (TM), where the size of TM is $\sqrt{K} \times \sqrt{K}$, such that K is the number of groups that divide the sampling space, with $\sqrt{K} \in \mathbb{N}$. TM defines the position of each pixel according to the position matrix $\mathbf{P} \in \{1, ..., K\}^{\sqrt{K}\times\sqrt{K}}$. The positions of $\mathbf{P}$ plays a crucial role in the smoothness when transitioning from one group to the following.

Four examples of TM with $K = 16$ are shown in Fig. 2. The first example is shown in Fig. 2(a), which is the naïve approach, yet a significant jump in subpixel position occurs when transitioning from group 4 to group 5. The second example is depicted in Fig. 2(b) which shows a random permutation. However, this design is not ideal due to uncertainty regarding the distance between each transition of the groups. The third example is illustrated in Fig. 2(c), which shows an optimization algorithm [27], [28] that ensures equal distances for all positions; however, for larger values of $\sqrt{K}$, this distance becomes too large and may be noticeable in all transitions between groups. The fourth example is our SCA design, which is represented in Fig. 2(d) by applying horizontal flipping to the even rows of Fig. 2(a). This arrangement reduces the distance at each transition except for the final one from group 13 to group 1, which is noticeable as we change from one measurement to another within a full cycle. With the matrix $\mathbf{P}$ leading the position of each kernel, we can construct a tensor $\mathcal{A} \in \{0, 1\}^{\sqrt{K}\times\sqrt{K}\times K}$ of stacked kernels as follows

$$\mathcal{A}_{i,j,k} = \begin{cases} 1 & \text{if } k = P_{i,j} \\ 0 & \text{if } k \neq P_{i,j} \end{cases}. \tag{2}$$

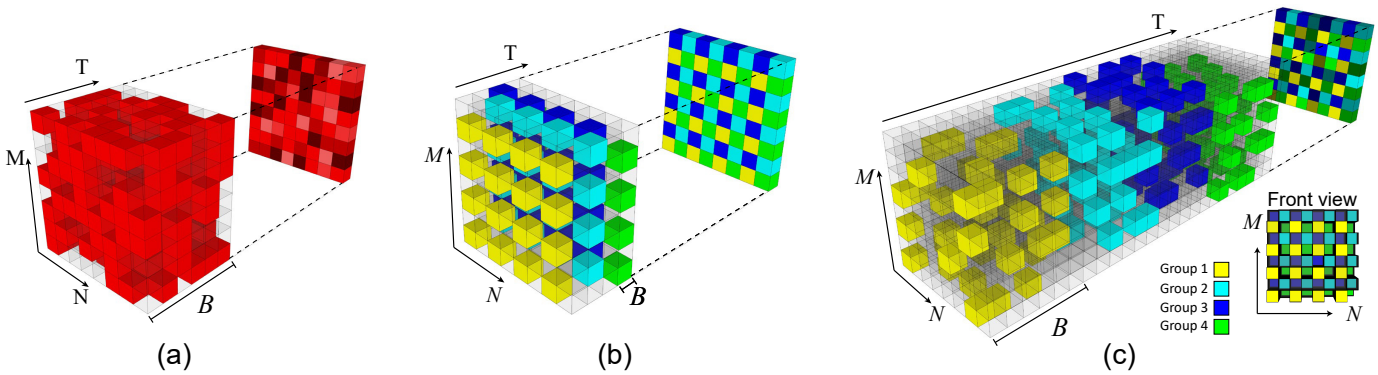


Fig. 1. Comparison of sampling strategies for CACTI with a multiplexing factor B. (a) random binary sampling, where red voxels denotes 50% activation. (b), subsampling method similar to multispectral filter array but applied as a temporal offset [27]. (c) our proposed SCA approach that merges compressive and subsampling techniques, for this example the number of groups is $K = 4$.

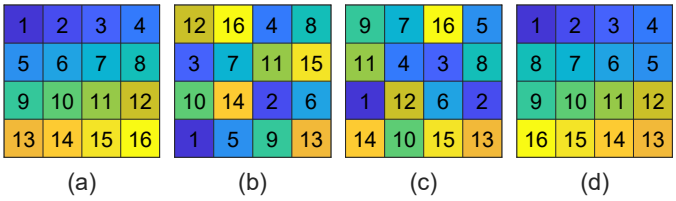


Fig. 2. Four designs of the TM, where the number of groups is $K = 16$, (a) sequential pattern; (b) random TM; (c) designed pattern from equal distances; (d) zigzag pattern.

In the temporal analysis using the subsampling method from Fig. 1(b), each kernel in $\mathcal{A}$ extracts a single frame from the datacube. In contrast, our SCA approach multiplexes a set of frames per kernel, with B as the multiplexing factor. Considering this, we have to group all the frames in the datacube, forming K groups of B frames as follows

$$G_{k,:} = \mathbf{\Theta}^{\alpha} G_{k-1,:}, \tag{3}$$

where $\mathbf{G} \in \{0,1\}^{K \times T}$ is the frame grouping matrix that shows in which frames each group is active. To displace the multiplexing frames of each group in time, we define the permutation matrix $\mathbf{\Theta} \in \{0,1\}^{T \times T}$ as

$$\mathbf{\Theta} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}, \tag{4}$$

and the temporal dispersion α, that is an integer constrained between 0 and B that controls the temporal overlap. Fig. 3 illustrates how the grouping matrix $\mathbf{G}$ assigns different groups to the frames using four values of temporal dispersion. In the case of $\alpha = 0$, as shown in Fig. 3(a), there is no temporal dispersion, which means that each group is sampling the same 8 frames at each subpixel position, resulting in an upsampled version of the traditional CACTI system. On the other hand, the case for $\alpha = B$ is shown in Fig. 3(d), where there is no temporal overlap between frames.

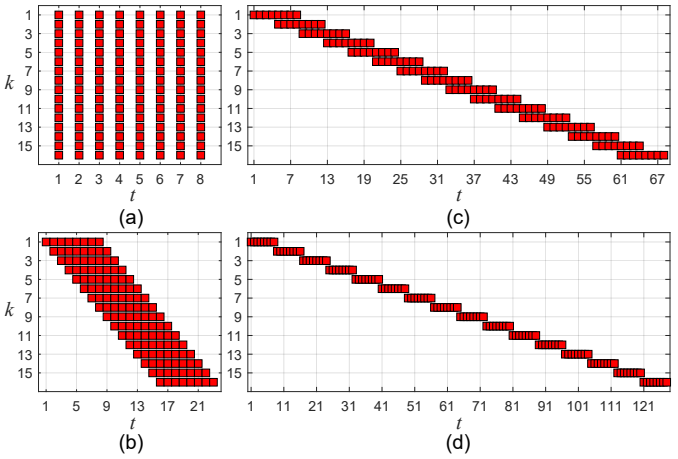


Fig. 3. Four examples of the frame grouping matrix $\mathbf{G}$, whose number of groups is set constant at $K = 16$ (see Eq. (3)). Each example varies the temporal dispersion α and the number of frames T according to Eq. (5), *i.e.*, (a) $\alpha = 0$ denotes no dispersion; (b) $\alpha = 1$ indicates one pixel dispersion; (c) $\alpha = 4$ four pixel dispersion; and (d) $\alpha = B$ maximum temporal dispersion when α is equal to the multiplexing factor. In each plot k represents the group index with each group compressing B frames and t denotes the axis of the frame index sampled by each group.

Nevertheless, as shown in Fig. 4(a), there is a reduction in sampling density with increasing values of α, where $K = 16$, $B = 8$ and $\alpha = 1$, frame 1 activates only group 1; in frame 2, groups 1 and 2 are active, and so forth. The highest sampling density is observed between frames 8 and 16, during which all eight groups are active. Therefore, to ensure a consistent sampling rate across all frames, we start by activating groups from frames 10 to 16. This approach can be facilitated by introducing a Boolean parameter $\beta \in \{0,1\}$ to activate asynchronous sampling (AS), thereby setting a cap on the total number of frames T sampled (refer to Fig. 4(b)), the resulting expression for the number of frames is

$$T = \begin{cases} \alpha(K - 1 + \beta) + B(1 - \beta), \alpha \neq 0 \vee \beta \neq 1 \\ B, \quad s.t. \;\; \alpha = 0 \wedge \beta = 1. \end{cases} \tag{5}$$

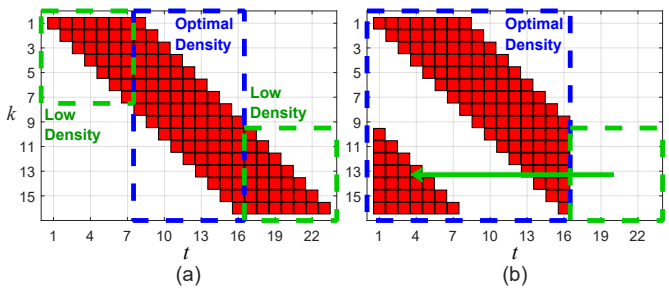


Fig. 4. Example of a frame grouping matrix $\mathbf{G}$ (see Eq. (3)) with AS β, number of groups $K = 16$ and temporal dispersion $\alpha = 1$ and multiplexing factor $B = 8$. Without AS adjustment (a), the system has a total of 23 sampled frames; (b), when making the AS adjustment, the number of sampled frames decreases to 16.

B. Scalable coded aperture design algorithm

We define $\mathcal{E} \in \{0,1\}^{m\times n\times B}$ as an arbitrary compressive sampling tensor for SCV, which can be a classical random CA with 50% of transmittance or any designed one [7]. Then, using the grouping matrix $\mathbf{G}$ and the computed stacked kernels $\mathcal{A}$, our design algorithm uses $\mathcal{E}$ as a baseline CA seed that is replicated K times in a particular manner within the high resolution SCA $\mathcal{C}$. The inputs of Algorithm 1 are $\mathbf{G}$, $\mathcal{A}$, and $\mathcal{E}$. Initially, the algorithm computes the indices $\mathbf{p}$, $\mathbf{q}$ such that

$$(p_l, q_l) = (i, j), \quad \text{s.t } \mathbf{G}_{i,j,k} = 1, \tag{6}$$

where $\mathbf{p}$ and $\mathbf{q}$ represent the row and column indices of the non-zero entries in $\mathbf{G}$–similar to the use of MATLAB's `find()` or Python's `where()` functions. We also define $\mathbf{e}$ as a cyclic index vector that iterates over the temporal masks in $\mathcal{E}$ from the first to the last mask such that

$$e_l = \text{mod}(l, B) + 1, \quad l = \{0, 1, \ldots, L-1\}, \tag{7}$$

where $L = KB$. Using the index vectors, the SCA is constructed iteratively (see lines 13–14): for each index i, the kernel $\mathcal{A}_{:,:,p_i}$ expands the corresponding submask $\mathcal{E}_{:,:,e_i}$, and the result is assigned to the mask $\mathcal{C}_{:,:,q_i}$. A detailed version of the algorithm can be found in the Supplementary Material.

III. Reconstruction

After completing the sampling process and obtaining the compressive measurement $\mathbf{Y}$, we observe that the projection is temporally decoupled into K groups, each defined by a kernel from $\mathcal{A}$. Therefore, to start the reconstruction process, we need to recover each compressed group within $\mathbf{Y}$. In the following, we define the decimated versions of $\mathbf{Y}^{(k)}$ and $\mathcal{X}^{(k)}$ that are used by the compressive reconstruction algorithms of subsection III-A. Each decimated measurement is given by

$$\mathbf{Y}^{(k)} = \mathbf{D}^{(k)}_{\text{row}} \mathbf{Y} \left(\mathbf{D}^{(k)}_{\text{col}}\right)^{\top}, \tag{8}$$

where $\mathbf{D}^{(k)}_{\text{row}} \in \{0,1\}^{m\times M}$ and $\mathbf{D}^{(k)}_{\text{col}} \in \{0,1\}^{n\times N}$ are selection matrices that extract the corresponding decimated rows and columns. Since the decimation must be performed differently for each group–depending on the kernel defined in

Algorithm 1 The compact version of the SCA algorithm

Inputs: $\mathbf{G}$, $\mathcal{A}$, $\mathcal{E}$ ▷ Grouping matrix, stacked kernels, and submask.
Output: $\mathcal{C}$

1: **function** CreateSCA($\mathbf{G}, \mathcal{A}, \mathcal{E}$)
2: Set m, n, B as the size of each dimension of $\mathcal{E}$ ▷ Extract dimensions of submask.
3: Set K, T as the size of each dimension of $\mathbf{G}$ ▷ Extract dimensions of grouping matrix.
4: $L \leftarrow KB$
5: $l \leftarrow 0$
6: **for** $k = 1$ to K **do** ▷ Construct $\mathbf{p}$ and $\mathbf{q}$ as in Eq. (6).
7: **for** $t = 1$ to T **do**
8: **if** $G_{k,t} = 1$ **then**
9: $(p_l, q_l) \leftarrow (t, k)$
10: $l \leftarrow l + 1$
11: **for** $l = 0$ to $L - 1$ **do** ▷ Construct $\mathbf{e}$ as in Eq. (7).
12: $e_l \leftarrow \text{mod}(l, B) + 1$
13: **for** $i = 1$ to L **do** ▷ Construct SCA.
14: $\mathcal{C}_{:,:,\mathbf{q}_i} \leftarrow \mathcal{C}_{:,:,\mathbf{q}_i} + \left(\mathcal{E}_{:,:,\mathbf{e}_i} \otimes \mathcal{A}_{:,:,\mathbf{p}_i}\right)$
return $\mathcal{C}$ ▷ Output of the function.

$\mathcal{A}$–a distinct pair of selection matrices are constructed for each group k. These matrices are defined as

$$\begin{aligned} \left(D^{(k)}_{\text{row}}\right)_{i, s_k + (i-1)\sqrt{K}} &= 1, \quad i = \{1, \ldots, m\}, \\ \left(D^{(k)}_{\text{col}}\right)_{j, g_k + (j-1)\sqrt{K}} &= 1, \quad j = \{1, \ldots, n\}, \end{aligned} \tag{9}$$

where $\mathbf{s}$ and $\mathbf{g}$ are vectors containing the leading pixel positions obtained from $\mathcal{A}$ such that

$$(s_k, g_k) = (i, j), \quad \text{s.t } \mathcal{A}_{i,j,k} = 1. \tag{10}$$

Therefore, (s_k, g_k) is any pair of indices for which the entry $\mathcal{A}_{s_k, g_k, k}$ is non-zero. Finally, to match each decimated measurement $Y^{(k)}$ with its corresponding groundtruth signal $\mathcal{X}^{(k)}$, the high-resolution video tensor $\mathcal{X}$ must also be decimated. Nevertheless, in contrast to the decimation performed in Eq. (8), this must be done in groups of B frames per each grouping index k. Therefore, the decimated video tensor is defined as

$$\mathcal{X}^{(k)} = \mathbf{D}^{(k)}_{\text{row}} \mathcal{X}_{:,:,t+\alpha(k-1)} \left(\mathbf{D}^{(k)}_{\text{col}}\right)^{\top}, \; t = \{1, \ldots, B\}, \tag{11}$$

where α is the temporal dispersion previously defined in Eq. (3). With this in mind, the corresponding submask of the decimated datacube is the same CA $\mathcal{E}$ across all subproblems, therefore, Eq. (1) can be decomposed as

$$\mathbf{Y}^{(k)} = \left(\sum_{b=1}^{B} (\mathcal{X}^{(k)}_{:,:,b} \odot \mathcal{E}_{:,:,b})\right) + \Omega^{(k)}, \tag{12}$$

where each subproblem can be solved independently with a lower computational burden and a lower memory requirement.

The proposed method comprises two main steps to obtain the video from the measurements: The first stage (see Fig. 5) involves compressively reconstructing each subproblem $\mathbf{Y}^{(k)}$

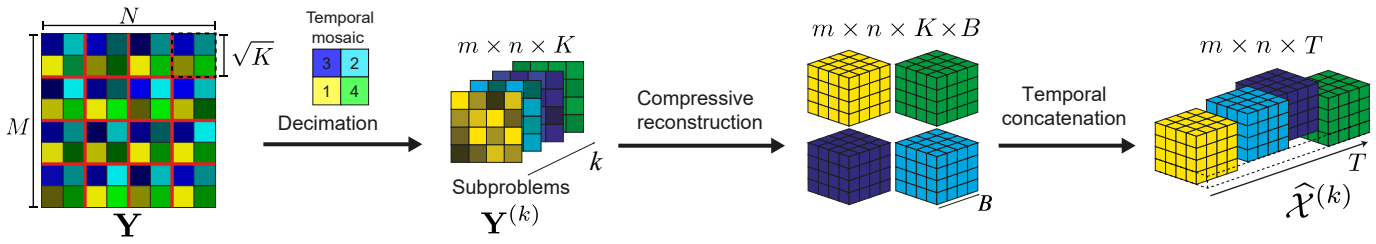


Fig. 5. Compressive reconstruction stage from a single compressive projection. First, the measurement is decimated based on the kernels $\mathcal{A}$. Next, each subproblem is solved using compressive reconstruction algorithms. Each resulting cube corresponds to the signal in different time windows, and therefore, concatenation along the temporal axis is applied.

to obtain the datacube $\widehat{\mathcal{X}}^{(k)}$. This reconstruction can proceed sequentially or in parallel, since there is no interdependence among the subproblems.

Subsequently, each reconstructed datacube obtained in the demultiplexing stage must be mapped onto a high-resolution video grid with T frames. Due to spatial decimation, it becomes necessary to fill in the missing information. Fig. 6 illustrates the pipeline for the reconstruction options. A detailed description of each process is provided in subsections III-A and III-B.

A. Compressive reconstruction algorithm

In compressive reconstruction techniques, it is useful to describe Eq. (12) in the form $\mathbf{x}^{(k)} = \text{Vec}(\mathcal{X}^{(k)})$, the measurement $\mathbf{y}^{(k)} = \text{Vec}(\mathbf{Y}^{(k)})$, and the sensing matrix $\mathbf{H} = [\text{Diag}(\text{Vec}(\mathcal{E}_{:,:,1})) \dots \text{Diag}(\text{Vec}(\mathcal{E}_{:,:,B}))]$, for $k = \{1, ..., K\}$ and $\boldsymbol{\omega} = \text{Vec}(\boldsymbol{\Omega})$, where $\text{Vec}(\cdot)$ performs matrix and tensor vectorization. It should be noted that this configuration allows the division of the entire measurement into smaller segments, regardless of the dimensions N, M and T. These segments can be processed using any SCV reconstruction algorithm. The strategy divides the high-resolution datacube $N \times M \times T$ into more manageable $m \times n \times B$ subproblems, ensuring compliance with computational limitations. Thus, Eq. (12) can be generalized to the subproblem k as

$$\mathbf{y}^{(k)} = \mathbf{H}\mathbf{x}^{(k)} + \boldsymbol{\omega}^{(k)}, \tag{13}$$

the underlying k^{th} datacube can be recovered by solving the following SCV inverse problem

$$\widehat{\mathbf{x}}^{(k)} = \underset{\mathbf{x}^{(k)}}{\text{argmin}} \ \frac{1}{2}\|\mathbf{y}^{(k)} - \mathbf{H}\mathbf{x}^{(k)}\|_2^2 + \lambda \, \text{Reg}(\mathbf{x}^{(k)}), \tag{14}$$

where λ is a parameter used to balance the fidelity term $\|\mathbf{y}^{(k)} - \mathbf{H}\mathbf{x}^{(k)}\|_2^2$ and the regularization term $\text{Reg}(\cdot)$. Eq. (14) can be solved iteratively [9]–[13], [29] or by a DL model [15]–[18], [21]–in some cases as unrolling networks [20], [30]. On the other hand, the regularizer $\text{Reg}(\cdot)$ is usually the norm l_1, aimed at promoting sparsity on some representation basis such as Wavelets [31] or discrete cosine transform (DCT). Another option is to use total variation (TV) [32], [33], VBM4D [34] or WNNM [35] as denoisers. In recent years, a series of DL-based denoisers have been developed, such as FFDNet [36] and fastDVDNet [37].

Since the subproblems are independent, these algorithms can be executed in parallel to reduce the reconstruction time. Once the decimated videos $\widehat{\mathcal{X}}^{(k)}$ are reconstructed, the subsequent step involves merging each subproblem to create our final high-resolution video $\widehat{\mathcal{X}}$ using the upscaling process outlined in the following section.

B. Upscaling stage

After finishing the compressive reconstruction of the K subproblems the decimated videos $\widehat{\mathcal{X}}^{(k)}$ obtained by solving Eq. (14), must be mapped into the high-resolution video $\bar{\mathcal{X}} \in \mathbb{R}^{N \times M \times T}$, however, due to the decimation introduced during sampling, we must fill in the missing information to obtain the final reconstruction $\widehat{\mathcal{X}}$. As shown in Fig. 6, the task has two alternatives, super resolution (SR) and tensor completion.

First, we may start by locating the reconstructed pixels at their corresponding subpixel positions in the high-dimensional datacube $\bar{\mathcal{X}}$, which is determined by the stacked kernels $\mathcal{A}$, which was used in the decimation for each subproblem, and then the missing data where filled to obtain our reconstructed video $\widehat{\mathcal{X}}$. This can be achieved using tensor completion (TC) algorithms; however, the main difficulty arises from the large amount of missing information within the space-time datacube. Often, tensor completion algorithms (both iterative and DL-based) require more than 5% of known data % [38]–[40]. To ensure a successful reconstruction, the following criteria must be met: $\frac{B}{\alpha(K-1)+B} \geq 5\%$ if $\beta = 0$ or $\frac{B}{K\alpha} \geq 5\%$ otherwise.

On the other hand, we may use SR algorithms to recover the original sampling resolution. However, depending on the TM order, can lead to video jitter caused by subpixel aliasing. Our testing shows that out of the four designs presented, the zigzag approach clearly minimizes jitter upon visual inspection, although there is some noticeable loss of information

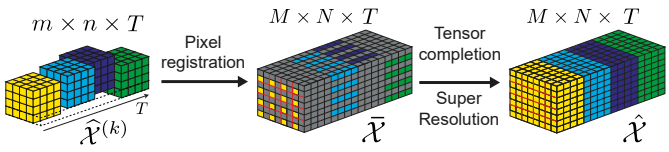


Fig. 6. Upscaling from concatenated datacubes involves mapping low-dimensional to high-dimensional datacubes via pixel registration, resulting in an incomplete $\bar{\mathcal{X}}$. Two methods to complete missing information presented: tensor completion and superresolution.

near the borders. Although SR is a viable option for filling in missing samples in reconstructed high-resolution videos, additional post-processing steps are required to prevent any jitter caused by misregistration of subpixel positions.

IV. Simulation Results

By utilizing the forward model of Eq. (1) and the SCA design outlined in Algorithm 1, we conduct simulations of the acquisition and reconstruction processes using a set of high-speed videos that are publicly accessible. To have consistent comparisons across all results we fix the resolution of the submaks $\mathcal{E}$ to $512 \times 512 \times 32$ and by changing the other design parameter we achieve different target resolutions. We used 28 datasets in total, each with a spatial resolution of 2048×2048 pixels and a temporal resolution of at least 1000 frames. The reconstruction methods are implemented on a desktop computer equipped with a single RTX 1080Ti GPU card.

To test several compressive reconstruction methods, we use the iterative algorithm GAP-TV [13] (all runs with 100 iterations) along with DL-based techniques such as RevSCI [21], SCI3D [20], and the more recent method STT [41]. Following the reconstruction of the subproblems $\widehat{\mathcal{X}}^{(k)}$, we utilize various upscaling options, including interpolation algorithms 2D interpolation (2DI) and 3D interpolation (3DI), as well as three SR networks: EDSR [42], VRT [43] and VRT++ [44]. All DL-based methods were implemented using PyTorch.

A. Analysis of α and its trade-offs

The parameter α is tested in several simulations with values ranging from 2 to 8. As discussed in Sec. II-A, $\alpha = 0$ is equivalent to a spatially upsampled version of the traditional CACTI and it is not considered. The results for scaling factor $K = 4$ and a variety of values of α are shown in Table I, which compares the performance of different combinations of reconstruction algorithms and upsampling methods. Regardless of the value of α or the combination of algorithms, the reconstruction results achieve a minimum of 32 dB on average. For $\alpha = 2$, the performance can reach nearly 39 dB by combining the latest SCI3D with 2DI or EDSR. In general, increasing α from 2 to 8–indicating more temporal dispersion– results in a loss of up to 2 dB in PSNR for any combined method. Nevertheless, this trade-off might be justifiable in enhancing the temporal resolution (or compression) of the compressive sampling design. When α is small, additional information is available to reconstruct each frame, resulting in a total of 14 frames for $K = 4$. On the other hand, by increasing the value of α, we reconstruct up to 32 frames ($\alpha = 8$, without overlap) with the same computational cost, although the image quality is clearly reduced.

Table I shows that when using TC, performance is often 2 dB lower than interpolation. Since TC algorithms may be affected when the percentage of known pixels is low, we will also investigate whether the use of asynchronous sensing adjustment can be advantageous. In Table II, we present simulation results with and without AS. It is clear that having more samples leads to progressively better results. However, this comes at the expense of a reduction in temporal resolution. When applying AS, the total frames decrease from 14 to 8 for $\alpha = 2$, from 17 to 12 for $\alpha = 3$, and from 20 to 16 for $\alpha = 4$.

TABLE I
AVERAGE PSNR—SSIM FOR 28 VIDEOS WITH $B = 8$, $K = 4$, AND VARYING ALPHA VALUES. FOR THIS SETTING N AND M ARE BOTH 512, AND THE VALUES OF T ARE 14, 20, 26, AND 32. TO ENSURE A FAIR COMPARISON, ONLY THE FIRST 14 FRAMES WERE CONSIDERED IN ALL CASES. FOR THE TC ALGORITHM, THE PERCENTAGES OF AVAILABLE PIXELS PER FRAME IS 57%, 40%, 30%, AND 25%, RESPECTIVELY.

Method, | α	2	4	6	8
GAP-TV + 2DI	34.69—0.9135	33.94—0.9092	33.38—0.9054	32.83—0.9012
GAP-TV + EDSR	34.40—0.9114	33.54—0.9056	32.84—0.9000	32.21—0.8946
GAP-TV + TC	32.65—0.8643	32.71—0.8763	32.68—0.8801	32.29—0.8764
RevSCI + 2DI	36.51—0.9282	35.97—0.9264	35.46—0.9242	34.97—0.9213
RevSCI + EDSR	36.57—0.9292	35.97—0.9271	35.41—0.9244	34.89—0.9212
RevSCI + TC	34.13—0.8906	33.96—0.8885	33.04—0.8621	31.94—0.8376
SCI3D + 2DI	38.84—0.9546	**38.14—0.9520**	**37.41—0.9489**	**36.83—0.9460**
SCI3D + EDSR	**38.87—0.9554**	37.93—0.9514	36.96—0.9463	36.23—0.9419
SCI3D + TC	35.70—0.9157	35.45—0.9158	35.02—0.9080	34.05—0.8907

TABLE II
AVERAGE PSNR—SSIM VALUES FROM 28 DATASETS WHEN AS IS USED. THE PARAMETERS USED ARE $K = 4$, THE TM IS THE ZIGZAG PATTERN (FIG.2(D)), AND THE RECONSTRUCTION METHOD USED IS SCI3D + TC.

Method, | α	2	3	4
Without AS	35.70—0.9158	35.63—0.9183	35.45—0.9158
With AS	36.32—0.9181	35.97—0.9189	36.32—0.9196

B. Performance at different scales

To test the capability of the proposed sampling scheme at its maximum demand, and thus the maximum spatial decimation, we choose to sample without any overlap by setting $\alpha = B$. Furthermore, we selected three levels of superpixels $K = 4$, 16, and 64 which resulted in the sampling and reconstruction of video datacubes with resolutions of $512 \times 512 \times 32$, $1024 \times 1024 \times 128$, and $2048 \times 2048 \times 512$, respectively. For every sample scheme tested, only one snapshot is obtained during one exposure time using the forward model in Eq. (1), which leads to compression ratios of 3%, 0.7% and 0.2%, respectively. In Table III, we present the average results of the reconstruction performances obtained using various combinations of reconstruction and SR methods.

As we increase the sampled resolution or compression ratio, there is a noticeable decrease in quality. In general, reconstructions at $K = 4$ (3% compression) achieve a performance of more than 32 dB regardless of the reconstruction method employed. This performance decreases to 28 dB for $K = 16$ (0.7% compression) and 25 dB for $K = 64$ (0.2% compression). The worst performance is often observed when using traditional inversion algorithms such as GAP-TV for the demultiplexing stage or employing TC methods for the upsampling process. Furthermore, modern deep neural networks achieve performances as high as 37 dB for a low compression ratio and as low as 32 dB for the most aggressive compression ratio.

Although interpolation methods, such as 2DI, match the numerical performance of the other upsampling methods based

TABLE III
AVERAGE PERFORMANCE METRICS, PSNR—SSIM, FOR THE RECONSTRUCTION OF 28 SCENES AT DIFFERENT SPACE-TIME RESOLUTIONS USING COMBINATIONS OF DIFFERENT RECONSTRUCTION AND SR METHODS. THE BEST VALUE IN EACH COLUMN IS HIGHLIGHTED IN BOLD.

Method	$512 \times 512 \times 32$	$1024 \times 1024 \times 128$	$2048 \times 2048 \times 512$
GAP-TV + 2DI	32.10 ± 6.38—0.8736 ± 0.1103	30.12 ± 5.16—0.8800 ± 0.0947	29.60 ± 5.09—0.9214 ± 0.0559
GAP-TV + 3DI	32.03 ± 6.44—0.8748 ± 0.1086	30.04 ± 5.22—0.8789 ± 0.0951	29.50 ± 5.13—0.9200 ± 0.0565
GAP-TV + EDSR	32.03 ± 6.43—0.8746 ± 0.1085	30.05 ± 5.20—0.8797 ± 0.0946	29.50 ± 5.11—0.9207 ± 0.0561
GAP-TV + VSR	32.03 ± 6.43—0.8744 ± 0.1084	30.03 ± 5.21—0.8794 ± 0.0948	29.49 ± 5.11—0.9203 ± 0.0562
GAP-TV + VSR++	32.03 ± 6.43—0.8747 ± 0.1082	30.04 ± 5.21—0.8796 ± 0.0946	29.48 ± 5.11—0.9203 ± 0.0561
GAP-TV + TC	31.86 ± 6.26—0.8567 ± 0.1199	27.91 ± 4.12—0.8527 ± 0.0947	25.42 ± 4.89—0.8095 ± 0.0557
RevSCI + 2DI	36.77 ± 6.57—0.9322 ± 0.0670	33.57 ± 5.45—0.9122 ± 0.0751	32.40 ± 5.24—0.9343 ± 0.0511
RevSCI + 3DI	36.80 ± 6.60—0.9348 ± 0.0651	33.44 ± 5.52—0.9116 ± 0.0754	32.22 ± 5.29—0.9331 ± 0.0519
RevSCI + EDSR	36.80 ± 6.56—0.9344 ± 0.0653	33.47 ± 5.48—0.9126 ± 0.0747	32.25 ± 5.26—0.9341 ± 0.0511
RevSCI + VSR	36.75 ± 6.55—0.9339 ± 0.0654	33.44 ± 5.48—0.9122 ± 0.0749	32.22 ± 5.26—0.9338 ± 0.0512
RevSCI + VSR++	36.70 ± 6.59—0.9328 ± 0.0659	33.33 ± 5.58—0.9078 ± 0.0799	32.12 ± 5.33—0.9293 ± 0.0567
RevSCI + TC	35.09 ± 7.00—0.8903 ± 0.0997	28.06 ± 3.54—0.8546 ± 0.0911	27.12 ± 4.53—0.8522 ± 0.0851
SCI3D + 2DI	37.16 ± 6.75—0.9355 ± 0.0657	33.79 ± 5.55—0.9143 ± 0.0742	32.58 ± 5.29—0.9354 ± 0.0505
SCI3D + 3DI	37.23 ± 6.77—0.9389 ± 0.0630	33.66 ± 5.62—0.9140 ± 0.0743	32.41 ± 5.34—0.9341 ± 0.0514
SCI3D + EDSR	37.22 ± 6.72—0.9383 ± 0.0633	33.69 ± 5.58—0.9150 ± 0.0734	32.43 ± 5.30—0.9353 ± 0.0503
SCI3D + VSR	37.09 ± 6.75—0.9367 ± 0.0640	33.63 ± 5.61—0.9140 ± 0.0742	32.38 ± 5.31—0.9346 ± 0.0506
SCI3D + VSR++	37.12 ± 6.73—0.9371 ± 0.0637	33.57 ± 5.64—0.9116 ± 0.0762	32.31 ± 5.35—0.9316 ± 0.0533
SCI3D + TC	35.54 ± 7.35—0.8931 ± 0.0989	28.14 ± 3.58—0.8565 ± 0.0900	27.54 ± 3.32—0.8515 ± 0.0831
STT + 2DI	37.35 ± 6.57—0.9403 ± 0.0587	**33.93 ± 5.45**—0.9169 ± 0.0714	**32.69 ± 5.22—0.9364 ± 0.0496**
STT + 3DI	**37.45 ± 6.61—0.9439 ± 0.0554**	33.80 ± 5.54—0.9166 ± 0.0714	32.52 ± 5.27—0.9351 ± 0.0506
STT + EDSR	37.41 ± 6.53—0.9433 ± 0.0557	33.82 ± 5.51—**0.9176 ± 0.0704**	32.55 ± 5.26—0.9363 ± 0.0494
STT + VSR	37.24 ± 6.49—0.9422 ± 0.0561	33.62 ± 5.50—0.9155 ± 0.0714	32.38 ± 5.20—0.9342 ± 0.0500
STT + VSR++	37.18 ± 6.53—0.9413 ± 0.0570	33.42 ± 5.58—0.9116 ± 0.0742	32.13 ± 5.25—0.9288 ± 0.0535
STT + TC	35.84 ± 7.15—0.9021 ± 0.0897	28.27 ± 3.40—0.8605 ± 0.0861	27.82 ± 3.49—0.8535 ± 0.0822

on SR, after the visual inspection shown in Fig. 7 (see matching video in Visualization 1), we observe that diagonal lines exhibit less aliasing when using the EDSR method. Therefore, we can argue that the best SR method in qualitative terms seems to be EDSR. Visualizations 2 and 3 provide samples of reconstructed videos of $2048 \times 2048 \times 512$ from a single SCA compressive measurement using STT+EDSR. Visualization 2 is a 15 seconds segment of a movie trailer rendered at the standard 24 fps, while Visualization 3 is an all-sky timelapse that captures a night-long recording at 1 frame per minute.

C. Mask design comparison

We continue by comparing the proposed SCA scheme with three traditional CA mask designs options to perform SCV reconstruction. The first mask follows the traditional CACTI implementation (shown in Fig.1(a)) and consists of a random mask with transmittance of 50%. The second mask is also a random CACTI mask, now constrained with a transmittance of 12% and 3%, equivalent to the corresponding SCA for videos of $512\times512\times32$ and $1024\times1024\times128$, respectively. The third mask is a subsampling mask of the full video, enforcing a non-overlapping condition in both space and time to prevent any multiplexing of temporal information, as shown in Fig.1(b).

Table IV presents a comparative analysis of the reconstruc-

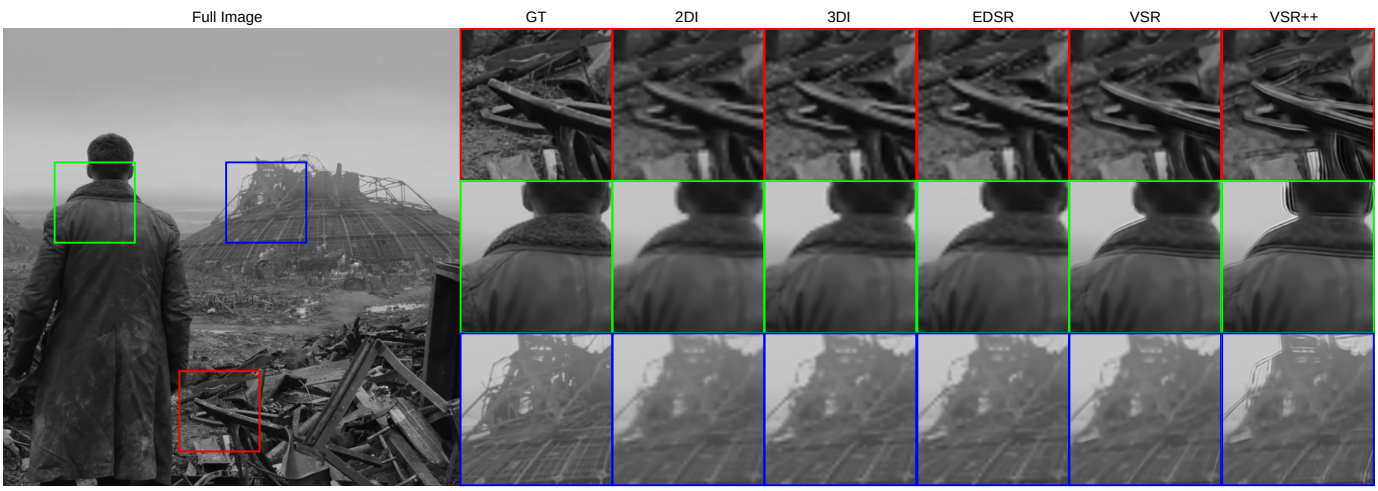


Fig. 7. Comparison of reconstruction quality for different SR methods. The sampling mask is designed with $K = 64$, and the demultiplexing method used is STT. One out of 512 reconstructed frames of a 2048×2048 movie trailer video is displayed. A slow-motion playback of the fully reconstructed video is available in Visualization 1.

tion quality in terms of PSNR and SSIM for the different masks and resolutions. For this comparison, we use GAP-TV to reconstruct the video frames, given its speed and flexibility, as well as its ability to handle arbitrary mask designs without requiring CNN model training. Since GAP-TV requires the full mask and the video to be loaded into memory, we were unable to test using the highest resolution $2048 \times 2048 \times 512$, constraining the comparison to masks of dimensions $512 \times 512 \times 32$ and $1024 \times 1024 \times 128$.

TABLE IV
PSNR AND SSIM RESULTS COMPARING DIFFERENT SAMPLING METHODS, EACH DATAPOINT IS THE AVERAGE OF 28 VIDEOS, TR DENOTES TRANSMITTANCE OF THE CODED APERTURE.

Mask, | Resolution	$512 \times 512 \times 32$	$1024 \times 1024 \times 128$
Random Tr={50%, 50%}	26.14—0.7844	22.83—0.7436
Random Tr={12%, 3%}	27.13—0.7858	22.65—0.7494
Subsampling Tr={3.12%, 0.78%}	27.63—0.8512	26.31—0.8449
Proposed mask Tr={12%, 3%}	**31.85—0.8793**	**29.97—0.8765**

As shown in Table IV, the proposed mask consistently outperforms all other methods in terms of both PSNR and SSIM for different resolutions, achieving an advantage on the average PSNR over 4.2dB and 3.6dB for the lower and higher resolution datacubes, respectively. We also present a comparative result of one of the reconstructed frames in Fig. 8, where it is evident that the proposed mask leads to a superior visual quality consistent with the quantitative results. When inspecting the companion reconstructed video available in Visualization 4, we can remarkably notice fewer artifacts when using the proposed SCA scheme.

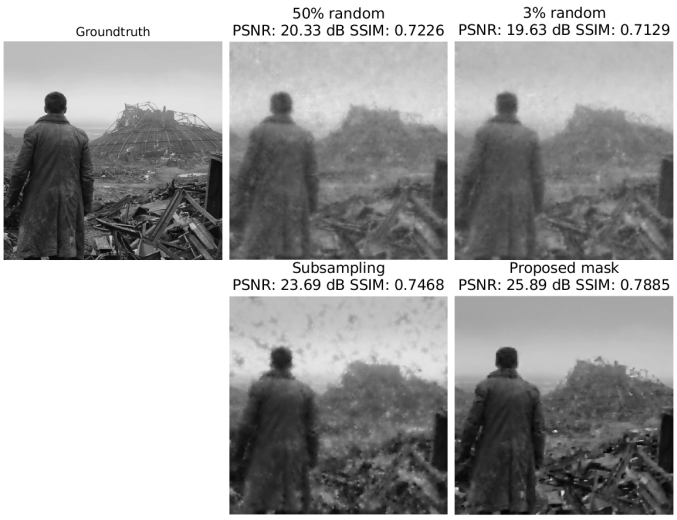


Fig. 8. Comparison of reconstruction quality for different sampling masks for one out of the 128 reconstructed frames of the 1024×1024 movie trailer video. All methods used GAP-TV for reconstruction and our proposed mask used the 2DI method for its final upscale stage. A playback of the reconstructed video is available in Visualization 4.

D. Computational complexity analysis

We analyze the computational complexity of the proposed SCA method by dividing the process in three main stages: measurement preprocessing, subproblem demultiplexing, and frame upscaling. The first stage–measurement preprocessing–involves a series of indexing and concatenation operations, which are relatively fast. In our setup, this step takes approximately 0.1 seconds and it can be scaled up without a major increase in time. For the second stage, we can recall from Eq. (14) that by decimating the measurement, the demultiplexing stage can be solved by K equal SCV reconstructions, using any of the recovery methods at the resolution of the submask seed $\mathcal{E}$, in this case at $256 \times 256 \times 8$. This is the classical approach of divide-and-conquer. Thus, if t_d is the time it takes to solve Eq. (14), then the complete demultiplexing time using a single core scales linearly in the form Kt_d, where t_d is heavily dependent on the nature of the algorithm used (iterative or neural estimation). Nevertheless, note that the whole process can be parallelized in K cores, if available. Finally, for the upscaling stage there are two possible ways to perform it according to the chosen approach. One option is super-resolved frame-by-frame for 2D upscaling while the other option is super-resolved stack-by-stack performing 3D upscaling. To limit the memory consumption for low-power systems we keep the stack with a limit of 8 frames. Then if t_u is the processing time for one frame or stack, then our upscaling stage increases linearly as Tt_u for frame-by-frame or $Tt_u/8$ for stack-by-stack. The exception to the rule are TC methods given the gigantic memory requirements, restricting the maximum resolution possible to be upscaled by our hardware at once to $512 \times 512 \times 8$, making this particular method to scale as $TKt_u/32$.

Table V summarizes the total processing time for different demultiplexing and upscaling methods, evaluated independently. When comparing the time it took to solve each demultiplexing subproblem and the time taken to solve the full problem at the target resolution, we can verify that demultiplexing methods scale linearly. In contrast, we can notice that upscaling methods are resolution-dependent and exhibit higher computational costs as the target resolution increases. Using different combinations of demultiplexing and upscaling stages, we can obtain reconstructions at the highest resolution under 3 minutes for CPU methods (GAP-TV + 2DI) and under 10 minutes for GPU methods (STT + EDSR), which can be further improved if using parallelization and current state-of-the-art GPUs.

V. EXPERIMENTAL RESULTS

The traditional CACTI test setup uses external optics—such as shifting masks or DMDs—to temporally modulate the scene before detection. Even though DMDs may offer resolutions such as FullHD or even 4k, they also introduce complex calibration and optical alignment issues which hinders the spatial resolution of the acquired compressive measurement. Alternatively, pixel-wise programmable exposure sensors (PE-CMOS) [45] or the SCAMP-5 [46] could serve as an electronic coded aperture, yet current prototypes remain immature and expensive while not offering the high pixel density needed for an effective demonstration of the proposed SCA scheme. To overcome these limitations, we implemented the SCA

TABLE V
PROCESSING TIME OF EACH DEMULTIPLEXING AND UPSCALING METHODS IN SECONDS. COLUMNS INDICATE THE TIME FOR A SINGLE SUBPROBLEM, FRAME OR STACK AND THE FULL PROBLEM FOR DIFFERENT TARGET RESOLUTIONS.

Method		$512 \times 512 \times 32$ $K = 4$	$1024 \times 1024 \times 128$ $K = 16$	$2048 \times 2048 \times 512$ $K = 64$
Demultiplexing	t_d			
GAP-TV (CPU)	2.66	11.08	42.07	164.36
RevSCI (GPU)	0.72	4.27	9.67	31.29
SCI3D (GPU)	2.15	10.48	31.94	117.80
STT (GPU)	2.25	11.04	33.34	122.91
Upscaling	t_u			
2DI (CPU)	0.07	0.88	6.68	71.34
3DI (CPU)	0.01	1.23	7.30	84.17
EDSR (GPU)	0.42	7.27	51.76	325.26
VSR (GPU)	0.63	170.49	600.10	2594.10
VSR++ (GPU)	0.06	11.23	47.18	313.29
TC (GPU)	1.80	481.02	7217.30	115659.34

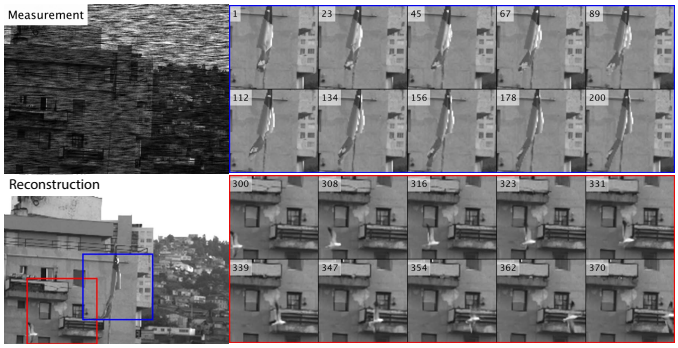


Fig. 9. Sample of an experimental result and reconstruction. (top-left) measurement acquired using the proposed sampling scheme with $K = 64$; (bottom-left) one frame of 1936×1464 pixels out of the 512 frames reconstructed with STT+EDSR; (right) sequence of reconstructed portions within the scene. The fully reconstructed video, played in slow motion, can be viewed in Visualization 5.

mask directly in the host computer RAM, using a frame grabber to acquire buffers of high-speed images from a CMOS sensor captured at nearly 500fps. The SCA binary mask is loaded into memory and used to selectively integrate only the "active" pixels per frame, while discarding inactive pixels (mask values of 0). In practice, the camera streams batches of 32 8-bit frames per buffer, each contributing only its active pixels toward a single 16-bit compressed image that is also being stored in memory. One of the advantages is that after the integration the buffer is immediately discarded and rewritten with a new batch of frames that are integrated until 512 frames have been compressed, minimizing memory requirements at the host computer. By using the selective integration of pixels in memory we can perform real-time, agnostic, and lightweight video compression. Although optical integration offers superior performance in terms of noise–for instance, one readout instead of several–we still believe the proposed implementation may handle measurement noise by tuning the compressive reconstruction algorithm for this situation. We may think about implementing the SCA codec into the frame grabber hardware or camera firmware in the future, creating a self-contained compressive camera, although in this work we concentrate on demonstrating the usefulness of the proposed SCA design using the aforementioned software solution as a proof of concept.

Experimental validation was performed using an Emergent HB-2800 camera equipped with a Sony IMX421 detector containing 1936×1464 pixels that can be read out at 410 frames per second. We preloaded the binary codes for $K = 64$ onto the camera hardware, which is a $2048 \times 2048 \times 512$ mask design, cropped to match the detector resolution. Every value of "1" in the code indicates which pixels within the frame are being integrated into the buffer. Therefore, every new pixel that is integrated may add 1 bit to the final integrated measurement. However, since we only used $B = 8$ pixels that could be multiplexed per spatial position, we might only need 16 bits for the final compressed images if the frames were quantized at 8 bits. Once the last code (frame 512 in this case) is used, the compressed image is measured and saved, and the buffer is reset to start coding and integrate the next compressed frame. In this way, we can end up with a sequence of compressed images that can be decoded and concatenated to create longer reconstructed videos.

For the reconstruction, we utilize the STT+EDSR method as the optimal combination in both quantitative and qualitative aspects, as demonstrated in the previous simulation results, and we use subpixel correction. Based on the measurements, we reconstruct 64 subproblems of size $256 \times 256 \times 8$. Subsequently, these subproblems are used to populate the superresolved image of size $2048 \times 2048 \times 512$ using EDSR.

Fig. 9 shows a compressive measurement and a reconstructed frame, along with two sequences of the reconstructed regions of the scene. The proposed method can reconstruct 512 frames at the detector resolution from a single shot under normal illumination conditions. We can observe a variety of motions in the scene, such as the minimal movement of the flag due to the low wind and the crossing of a seagull. Nevertheless, in the zoomed-in sequences, we can also observe a slight loss in spatial resolution due to the high value of $\sqrt{K}$ used, resulting in a high compression ratio and significant spatial decimation. The dynamics of the scene can be further inspected in Visualization 5. It is important to note that the scene is reproduced in slow motion and had to be compressed using MP4 algorithms.

As a final validation, we conducted a more extensive acquisition, this time capturing a sequence of 3 compressed measurements containing information equivalent to nearly 4 seconds of real-time video. We reconstructed 1536 frames as shown in the slow-motion video of Visualization 6. Fig. 10 shows one of the compressed measurements and one of the reconstructed frames. The recording was taken from our lab's window, in which we captured a variety of movements at different speeds and directions, including people and vehicles. Note that in the reconstructed videos, we not only see issues stemming from reconstruction or poor spatial sampling in some areas but also from artifacts resulting from the additional MP4 compression applied for visualization purposes.

In Fig. 10, the proposed design successfully compresses 512 frames of high-resolution, high-speed videos into a single shot. Moreover, the coding scheme is completely agnostic to the data and can be implemented in current CACTI systems

or as digital real-time codecs–similar to MP4, although way more lightweight–that can be easily integrated into the camera hardware. This integration enables the capture of sequences of compressed frames, suitable for very long videos or high-speed recordings with minimal memory usage. As an example, using this method, we can compress a 90-minute full-HD video sampled at 30 fps with only 360 compressed measurements. However, even though increasing the parameter K improves the compression ratio and scalability toward higher spatial and temporal resolutions, it also intensifies the decimation of the video datacube, resulting in a degradation of spatial resolution.

VI. Conclusion

We proposed a simple, yet effective scalable coded aperture design that enables scalable compressive temporal imaging. By merging time division multiplexing with traditional compressive video coding, we devised the design of a temporal mosaic that allowed for unprecedented spatial and temporal scalability during the acquisition process by segmenting spatially organized measurement pixels at specific intervals along the temporal axis. The SCA can be implemented in any existing CACTI system or even in the camera hardware as a lightweight codec. The usage of the proposed SCA makes the reconstruction also scalable, so it can be decomposed into a set of lower-resolution subproblems that can be efficiently handled by existing reconstruction methods for smaller datacubes–even already trained deep learning-methods–that may be implemented in parallel. Finally, the reconstructed datacubes from the demultiplexed subproblems are populated into a high-resolution grid that further undergoes interpolation, tensor completion, or super-resolution algorithms to fill the missing samples and generate the high-resolution video in space and time.

By adjusting the parameters K, B, α and β of the SCA design, we can increase or decrease the amount of compression with great flexibility. It is important to note that the scale parameter K has the most significant impact on scalability gains, but it also affects the potential loss of spatial resolution. Although the parameters α and β can help by increasing the available samples, and therefore offer better reconstructions and enable the use of tensor completion methods, in our analysis, the results of tensor completion were not competitive enough to compensate for the decrease in reconstructed frames.

From the presented simulations, we can state that even with an increased scalability and compression ratio, the reconstructions never fell below 30 dB, regardless of the combination of reconstruction methods used (excluding tensor completion methods that require a larger sampling density). However, the combination of state-of-the-art deep neural networks for the compressive reconstruction and/or interpolation (super resolution) stages, with the best combination being STT+EDSR, leads to average qualities of nearly 38 dB for 3% of compression, 34 dB for 0.7% of compression, and 33 dB for 0.2% of compression. The latter allows for superb reconstruction quality, as demonstrated in the companion experimental results, where we were able to compress 512 frames of a 1936×1464 video into a single snapshot. Although there is an evident loss of spatial resolution in the most aggressive compression ratios–which also happens in traditional video compression methods at similar compression levels, super resolution methods can still enhance the results if they are re-trained, especially when considering the precise subpixel position of each subproblem. However, the focus of this study has been on the versatility of the coding design rather than on the quality of the reconstruction provided by a specific algorithm.

In conclusion, the proposed SCA coding scheme for CACTI systems simplifies data acquisition by reducing the amount of temporal multiplexing allowed per pixel, and hence increasing compression in the temporal domain. This also eases the reconstruction process by dividing it into smaller subproblems, leading to remarkable scalability in both sensing and reconstruction, which may enable 4K optical or electronic video coding of hundreds of frames with a minimal hardware burden. We are currently investigating the addition of spatial multiplexing to enhance the spatial quality of the reconstructed frames and explore potential adaptations for compressive camera arrays [47].

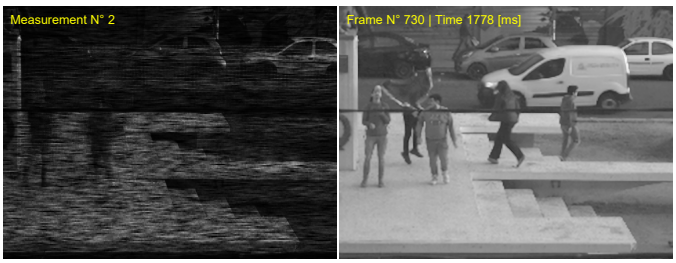


Fig. 10. Sample of another experimental result and reconstruction. (left) compressed measurement sequence using the proposed sampling scheme with $K = 64$; (right) reconstructed frame sequence at 1936×1464. The reconstructed video sequence, played in slow motion, can be viewed in Visualization 6.

Acknowledgements

We are grateful to the Optoelectronics Lab staff at PUCV for joyfully participating in the final demonstration videos.

References

[1] A. Konno, R. Uchikura, T. Ishihara, T. Tsujita, T. Sugimura, J. Deguchi, M. Koyanagi, and M. Uchiyama, "Development of a high speed vision system for mobile robots," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 1372–1377.

[2] P. Drews, G. Williams, B. Goldfain, E. A. Theodorou, and J. M. Rehg, "Vision-based high-speed driving with a deep dynamic observer," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1564–1571, 2019.

[3] S.-C. Lin, Y. Zhang, C.-H. Hsu, M. Skach, M. E. Haque, L. Tang, and J. Mars, "The architectural implications of autonomous driving: Constraints and acceleration," in *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, 2018, pp. 751–766.

[4] M. Versluis, "High-speed imaging in fluids," *Experiments in fluids*, vol. 54, pp. 1–35, 2013.

[5] N. M. Law, C. D. Mackay, and J. E. Baldwin, "Lucky imaging: high angular resolution imaging in the visible from the ground," *Astronomy & Astrophysics*, vol. 446, no. 2, pp. 739–745, 2006.

[6] S. Hertegård, H. Larsson, and T. Wittenberg, "High-speed imaging: applications and development," *Logopedics Phoniatrics Vocology*, vol. 28, no. 3, pp. 133–139, 2003.

[7] 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.

[8] 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.

[9] 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.

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

[11] 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 Transactions on Image Processing*, vol. 24, no. 1, pp. 106–119, 2014.

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

[13] Y. Liu, X. Yuan, J. Suo, D. J. 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, 2018.

[14] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[15] M. Iliadis, L. Spinoulas, and A. K. Katsaggelos, "Deep fully-connected networks for video compressive sensing," *Digital Signal Processing*, vol. 72, pp. 9–18, 2018.

[16] M. Qiao, Z. Meng, J. Ma, and X. Yuan, "Deep learning for video compressive sensing," *Apl Photonics*, vol. 5, no. 3, p. 030801, 2020.

[17] Z. Wang, H. Zhang, Z. Cheng, B. Chen, and X. Yuan, "Metasci: Scalable and adaptive reconstruction for video compressive sensing," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2083–2092.

[18] 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*. Springer, 2020, pp. 258–275.

[19] 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, no. 2, pp. B18–B29, 2021.

[20] Z. Wt, J. Zhangt, and C. Mou, "Dense deep unfolding network with 3d-cnn prior for snapshot compressive imaging," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 4872–4881.

[21] 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 Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 246–16 255.

[22] X. Yuan, Y. Liu, J. Suo, and Q. Dai, "Plug-and-play algorithms for large-scale snapshot compressive imaging," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1447–1457.

[23] L. Wang, M. Cao, and X. Yuan, "Efficientsci: Densely connected network with space-time factorization for large-scale video snapshot compressive imaging," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 18 477–18 486.

[24] G. Bub, M. Tecza, M. Helmes, P. Lee, and P. Kohl, "Temporal pixel multiplexing for simultaneous high-speed, high-resolution imaging," *Nature methods*, vol. 7, no. 3, pp. 209–211, 2010.

[25] W. Feng, F. Zhang, X. Qu, and S. Zheng, "Per-pixel coded exposure for high-speed and high-resolution imaging using a digital micromirror device camera," *Sensors*, vol. 16, no. 3, p. 331, 2016.

[26] M. Zhai, H. Li, J. Zhang, L. Tao, S. Wang, and H. Mao, "Temporal multiplexing of the scientific grade camera for hyper-frame-rate imaging," *Optics Express*, vol. 26, no. 16, pp. 20 813–20 822, 2018.

[27] N. Diaz, A. Alvarado, P. Meza, F. Guzmán, and E. Vera, "Multispectral filter array design by optimal sphere packing," *IEEE Transactions on Image Processing*, vol. 32, pp. 3634–3649, 2023.

[28] E. Vera, F. Guzmán, and N. Díaz, "Shuffled rolling shutter for snapshot temporal imaging," *Optics Express*, vol. 30, no. 2, pp. 887–901, 2022.

[29] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "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, 2011.

[30] C. Yang, S. Zhang, and X. Yuan, "Ensemble learning priors driven deep unfolding for scalable video snapshot compressive imaging," in *European Conference on Computer Vision*. Springer, 2022, pp. 600–618.

[31] S. Mallat, *A wavelet tour of signal processing*. Elsevier, 1999.

[32] 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.

[33] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, "An iterative regularization method for total variation-based image restoration," *Multiscale Modeling & Simulation*, vol. 4, no. 2, pp. 460–489, 2005.

[34] M. Maggioni, G. Boracchi, A. Foi, and K. Egiazarian, "Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms," *IEEE Transactions on image processing*, vol. 21, no. 9, pp. 3952–3966, 2012.

[35] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 2862–2869.

[36] K. Zhang, W. Zuo, and L. Zhang, "Ffdnet: Toward a fast and flexible solution for cnn-based image denoising," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4608–4622, 2018.

[37] 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*, 2020, pp. 1354–1363.

[38] Z. Long, Y. Liu, L. Chen, and C. Zhu, "Low rank tensor completion for multiway visual data," *Signal processing*, vol. 155, pp. 301–316, 2019.

[39] Y. Zheng and A.-B. Xu, "Tensor completion via tensor qr decomposition and l2, 1-norm minimization," *Signal Processing*, vol. 189, p. 108240, 2021.

[40] X.-L. Zhao, W.-H. Xu, T.-X. Jiang, Y. Wang, and M. K. Ng, "Deep plug-and-play prior for low-rank tensor completion," *Neurocomputing*, vol. 400, pp. 137–149, 2020.

[41] L. Wang, M. Cao, Y. Zhong, and X. Yuan, "Spatial-temporal transformer for video snapshot compressive imaging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[42] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[43] J. Liang, J. Cao, Y. Fan, K. Zhang, R. Ranjan, Y. Li, R. Timofte, and L. Van Gool, "Vrt: A video restoration transformer," *arXiv preprint arXiv:2201.12288*, 2022.

[44] K. C. Chan, S. Zhou, X. Xu, and C. C. Loy, "Basicvsr++: Improving video super-resolution with enhanced propagation and alignment," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 5972–5981.

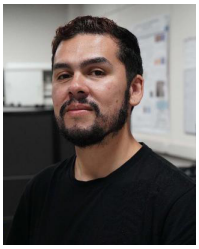
[45] J. Zhang, J. Newman, Z. Wang, Y. Qian, P. Feliciano-Ramos, W. Guo, T. Honda, Z. S. Chen, C. Linghu, R. Etienne-Cummings *et al.*, "Pixel-wise programmability enables dynamic high-snr cameras for high-speed microscopy," *Nature Communications*, vol. 15, no. 1, p. 4480, 2024.

[46] P. Dudek, T. Richardson, L. Bose, S. Carey, J. Chen, C. Greatwood, Y. Liu, and W. Mayol-Cuevas, "Sensor-level computer vision with pixel processor arrays for agile robots," *Science robotics*, vol. 7, no. 67, p. eabl7755, 2022.

[47] F. Guzmán, P. Meza, and E. Vera, "Compressive temporal imaging using a rolling shutter camera array," *Opt. Express*, vol. 29, no. 9, pp. 12 787–12 800, Apr 2021.

Felipe Guzmán He received the B.Sc. degree in electronic engineering in 2018 and an M.S. degree in electrical engineering in 2019 from the Pontificia Universidad Católica de Valparaíso (PUCV), Chile. He earned his Ph.D degree in 2024 working on compressed sensing strategies for high-speed imaging. He is currently a Postdoctoral Researcher at PUCV, Chile. His research interests include computational imaging, deep learning, and wavefront sensing. He is a member of SPIE and OPTICA.

Nelson Diaz (M'15) received the B.S. and M.S. in computer science, and the Ph.D. in engineeering, all from the Universidad Industrial de Santander, in Colombia, in 2012, 2015 and 2019. In 2018, he was an intern with Telecommunications for Space and Aeronautics Laboratory (TéSA), France. His research interest are compressive sensing and computational imaging. Currently, he holds a postdoctoral position at the Pontificia Universidad Católica de Valparaíso, Chile. He is a member of OPTICA.



Bastián Romero He received a B.S. and M.S. degree in electronics engineering from the Pontificia Universidad Católica de Valparaíso, Valparaíso, Chile, in 2021 and 2024, where he is currently pursuing a Ph.D. degree in electrical engineering. His research interests include adaptive optics, deep learning and computational imaging. He is a Student Member of Optica and holds a National Phd Scholarship funded by ANID.

Esteban Vera (M'98,SM'24) received the Engineering diploma in electronics engineering, and the M.S. and Ph.D. degrees in electrical engineering from Universidad de Concepción, Chile, in 1999, 2003, and 2010. From 2001 to 2007, he worked for the Paranal and Gemini Observatories. In 2010, he was a postdoctoral researcher with the University of Arizona, and then a research scientist at Duke University from 2013. In 2016, he joined the School of Electrical Engineering at the Pontificia Universidad Católica de Valparaíso, Chile, where he is an associate professor. His research interests span the fields of computational imaging, compressed sensing, machine learning, optical computing, and adaptive optics. He is an associate editor for the IEEE Open Journal of Signal Processing and Optics Express. He is a member of the SPIE and a senior member of OPTICA.