Compressive Spectral Video Sensing Using The Convolutional Sparse Coding Framework CSC4D

Crisostomo Barajas-Solano[a], Juan-Marcos Ramirez[b], José Ignacio Martínez Torre[c], Henry Arguello[a,*]

[a] *Universidad Industrial de Santander, Colombia*
[b] *Instituto IMDEA Networks, Spain*
[c] *Universidad Rey Juan Carlos, Spain*

Abstract

Spectral Videos (SV) contain a scene's spatial-spectral-time information. Just as with Spectral Images, SVs require expensive sensing hardware, storage plus high frame ratios. Although Super Resolution techniques improve the quality of low-resolution SVs, Compressive Spectral Video Sensing (CSVS) senses high-quality SVs by extending the Compressive Sensing Image (CSI) techniques. CSI uses the universal Sparse Signal Representation (SSR) model for SVs and SIs despite the limited quality of the recovered signals. On the other hand, dictionaries synthesis models are used successfully for representing SIs, SVs, and in CSI. This work proposes the 4D convolutional sparse representation (CSC4D) for recovering full-resolution SV from CSVS measurements. It is based on a multidimensional formulation of the CSC model, profiting from its robustness without additional optical flow information. Extensive numerical simulations (two CSI architectures and noise models) show that the proposed CSC4D+CSVS improves the state-of-the-art in both quality and border sharpness by up to 1.5dB.

Keywords: Compressive spectral video sensing, convolutional sparse coding, sparse representation, spectral videos.

2010 MSC: 00-01, 99-00

1. Introduction

Spectral Images (SI), 3D datasets which contain information about the 2D spatial features within a scene plus the reflectance at specific electromagnetic wavelengths of such features, have found great acceptance in fields such as medical diagnosis [1], remote sensing [2] and military operations [3], just to mention a few examples. State-of-the-art works have included an additional dimension to the 3D array of an SI: discrete-time variation. This is, we obtain a collection of SIs within a time frame, separated by a discrete time period.

*This work was developed under the Doctorate Scholarship Colciencias 785 (Colombia).

* *Email address:* henarfu@uis.edu.co (Henry Arguello)

Also known as Spectral Videos (SV), have found great acceptance in object or human tracking [4, 5, 6], cancer detection [7], bile duct inspection [8], and several types of surgery [9].

In an SV, a full SI must be captured within a small time-frame to capture the time variations of interest within the observed scene [10]. However, techniques such as push-broom [11] or optical band-pass filters [12] require a considerable sensing time to capture a single spectral frame. These limitations in the available technologies and time restrictions lead to one of two scenarios: either we resort to faster and more expensive sensing equipment to capture the desired SVs within the time limits; or we use available, and not too expensive, technology to capture SVs with low spatial, spectral and/or temporal resolution, within the time limits. In this regard, solutions such as super-resolution techniques [13, 14] offer to recover a high-resolution version of an SV from low-resolution measurements.

A more compelling approach is the application of the Compressive Sensing Imaging (CSI) framework to the SV case, also known as Compressive Spectral Video Sensing (CSVS). CSI states that a full 3D SI can be recovered from a set of 2D encoded projections. This is, a SI is simultaneously scanned and compressed, reducing both the scanning time and storage requirements [15, 16, 17, 18]. CSVS proposes to compress sensing, consecutively, a collection of spectral frames using a collection of 2D coding apertures per spectral frame, and to recover the full SV using an extended version of the Sparse Signal Representation (SSR) model. In [10, 19] they propose to extend the Kronecker basis proposed by Arce *et. al.* [20], which follows the SSR model, by adding the Discrete Cosine Transform (DCT) for temporal compression.

Previous works on CSI as shown that the SSR analytical model can be improved using a synthesis model [21][22][23]. It is possible to learn a collection of features directly from an SI, instead of using a fixed orthonormal basis, to represent it and to improve the reconstruction quality. The Convolutional Sparse Coding (CSC) framework [24] proposes to represent a signal using an overcomplete collection of convolutional dictionary elements and sparse coefficient maps. The former contains the features within the signal, and the latter indicates the position and contributions for each feature. The sum of all the convoluted pair dictionary-coefficient results in a recovered version of the signal of interest. This convolutional representation has an intrinsic local shift-invariant structure and invariance to deformation, robustness to noise and improved border sharpness. In [22] we proposed an extension of the CSC model to represent SIs and to recover them from compressed measurements, the 3D Convolutional Sparse Coding (CSC3D) framework.

For the case of SVs, in [25] we proposed to take the CSC model even further to include the spatial, spectral, and temporal correlations within a moving scene. The result is the CSC4D framework, which uses a single 4D convolution operator for sparsely representing an SV. The previous statement becomes important when reviewing the state-of-the-art. Correa *et. al.* [10] proposed to compress sensing and reconstructing each spectral frame consecutively but independently, missing the temporal correlations between frames; while Lopez *et. al.* [19] proposed to include an additional optical flow term in the reconstruction process to include the temporal variation. The single 4D convolution in CSC4D maintains much of an SV's features and

reduces the necessity of additional terms in the reconstruction problem. Initially, we developed the CSC4D framework to address the super-resolution (SR) problem by increasing the spatial resolution of spectral videos [25]. However, the CSVS problem's compression matrix includes both the spatial and spectral dimensions, increasing complexity.

Nonetheless, this work's main contribution is to use the CSC4D framework within a CSVS framework for recovering high-resolution SVs from compressed measurements, replacing the SSR model as sparsifying basis and profiting on CSC's properties mentioned above. This proposal profits from CSC's properties of local shift-invariant structure, invariance to deformation, robustness to noise, and improved border sharpness. This research work includes extensive numerical simulations to evaluate the performance of the CSC4D model at recovering SVs from compressed measurements, at different compression levels, and in the presence of noise. The CSC4D model was compared to the SSR model and exhibits competitive performance with respect to the reference method at high compression ratios and noise levels. However, it outperforms the SSR model at mid-level compression ratios and noise levels. A modified acquisition scheme, based on a side-information scheme [26, 27], is used for recovering viable versions of SVs from compressive measurements using CSVS architectures with dispersive elements. Overall, the proposed CSC4D+CSVS model improves the recovery quality with respect to the state-of-the-art SSR method, where the recovered frames preserve the edges and textures of the spectral video frames.

This work is organized as follows: Section 2 presents previous work in CSVS, SSR, and CSC for both 3D and 4D. Section 3 includes the integration of the CSC4D framework within a CSVS recovery scheme and the proposed numerical solution. Section 4 presents the performance evaluation and results. Finally, Section 5 includes conclusions and future work.

2. Preliminary Background

2.1. Notation

Table 1 presents a detailed list of the particular notation to be used within this research work:

2.2. Compressive Spectral Video Sensing (CSVS)

SVs consist essentially of a collection of SIs captured during a time frame, $\boldsymbol{\mathcal{S}} = \{\boldsymbol{\mathcal{S}}^t, t = 1, ..., T \mid \boldsymbol{\mathcal{S}}^t \in \mathbb{R}^{M \times N \times L}\}$. This implies that SV sensing has the same limitations and shortcomings as those found in SI sensing [28]. Furthermore, SVs require high-speed sensing. Current state-of-the-art approaches propose an extension of Compressive Sensing Imaging (CSI) to SVs, known as Compressive Spectral Video Sensing (CSVS), to reduce the need for expensive spectral sensing equipment.

CSI states that the spatial-spectral information of a SI can be sensed as a set of 2D encoded projections. A typical CSI architecture encodes the 3D spatial-spectral information using a 2D coded aperture, which

Notation	Description
M, N, L, T, d, M_d, α, β, γ, ρ, λ, σ	real scalars.
$\mathbf{x}$	1D arrays.
$\mathbf{X}$	2D arrays.
$\boldsymbol{\mathcal{X}}$	3D and 4D arrays.
$\|\mathbf{x}\|_0$	ℓ_0 pseudo-norm, calculated as the number of non-zero entries of the vector $\mathbf{x}$.
$\|\mathbf{x}\|_p = (\sum \|\mathbf{x}\|)^{1/p}$	ℓ_p norm for $1 \leq p \leq 2$.
$\otimes$	Kronecker product.
$\overset{n}{*}$	n-dimensional cyclic convolution operation.
$\odot$	Hadamard product.
$\mathbf{a}^t$	$t-th$ temporal index.
$\mathbf{a}^{(j)}, \mathbf{a}^{(j+1)}$	iteration step.
$\mathbf{a^T}$ and $\mathbf{a^H}$	transpose and conjugated transpose, respectively.
$\mathbf{a}_m$	as a single sub index, denote the m-th element of a collection.
$\mathbf{a}_{i,j}$ and $\mathbf{a}_{i+u,j+v}$	as a pair of sub indices, denote spatial coordinates.
$[.]$	horizontal concatenation of two arrays.
$\mathbf{vec}(.)$	rearrangement of an N-dimensional array into a 1D array.
$\mathbf{diag}(\mathbf{x})$	creation of a diagonal matrix with the array $\mathbf{x} \in \mathbb{R}^N$ as its main diagonal.
$\hat{\boldsymbol{\mathcal{X}}} = \mathcal{F}_{ND}(\boldsymbol{\mathcal{X}})$ and $\boldsymbol{\mathcal{X}} = \mathcal{F}_{ND}^{-1}(\hat{\boldsymbol{\mathcal{X}}})$	ND-dimensional Fourier transform and its inverse, respectively.

Table 1: Detailed list of the particular notation to be used within this research work.

is then captured by an intensity detector (see Figure 1). Typically, the coded apertures are block–unblock lithographic masks or spatial light modulators (SLM) that offer good optical sensing properties. Black-or-white, or color-coded, apertures allow or block certain wavelengths of light at specific spatial locations. This process is synthesized in a sensing matrix $\mathbf{H} \in \mathbb{R}^{K \times MNL}$, with $K \ll MNL$. The sensing matrix can be expressed as $\mathbf{H} = \mathbf{SD}\boldsymbol{\Gamma}$, where $\mathbf{S}$ is the integration matrix, $\mathbf{D}$ is the dispersion matrix and $\boldsymbol{\Gamma}$ is the coding matrix [15].

Different CSI architectures generate different sensing matrices $\mathbf{H}$, containing the setting of each CSI system. For CSI architectures with dispersive elements, as CASSI and C-CASSI [20], the matrix $\mathbf{D}$ accounts for the dispersive phenomenon (see Figure 1); while for front-sensing architectures as 3D-CASSI [29] $\mathbf{D} = \mathbf{I}$ (see Figure 2). Matrix $\boldsymbol{\Gamma}$ is a diagonal matrix whose entries are the spectral responses of the optical filters in the coded apertures, which can be black-or-white or color coded. Finally, the matrix $\mathbf{S}$ performs the integration of the encoded and dispersed source. The 2D encoded projections, or compressive spectral measurements, are represented as $\mathbf{y} = \mathbf{Hs}$ where $\mathbf{y} \in \mathbb{R}^K$ and $\mathbf{s} = \mathbf{vec}(\boldsymbol{\mathcal{S}}) \in \mathbb{R}^{MNL}$.

CSVS senses compressively each spectral frame as an independent SI, obtaining only a few measurements per spectral frame. Then, a full version of the SV is recovered from the compressed measurements using a spatial-spectral-temporal orthonormal basis, $\boldsymbol{\Psi}$, following the sparse signal representation model (SSR). Correa-Pugliese *et. al.* in [10] proposes to sparsely represent an SV using the Kronecker basis [20] along with the discrete cosine transform (DCT) for temporal compression, as $\boldsymbol{\Psi}$ = 2D Wavelet $\otimes$ DCT $\otimes$ DCT.

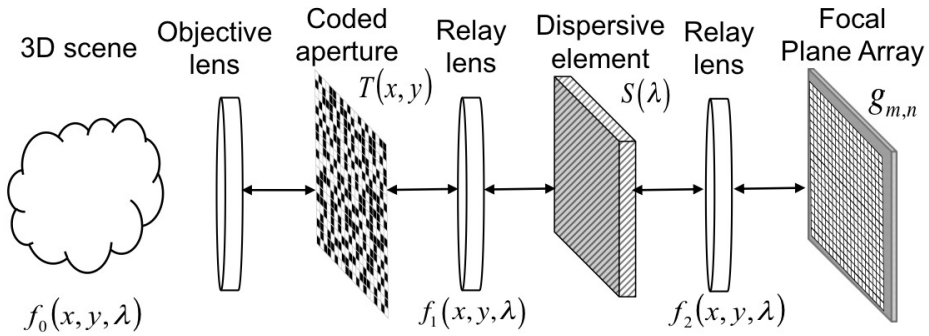

Figure 1: CASSI CSI architecture shceme. Source [15]

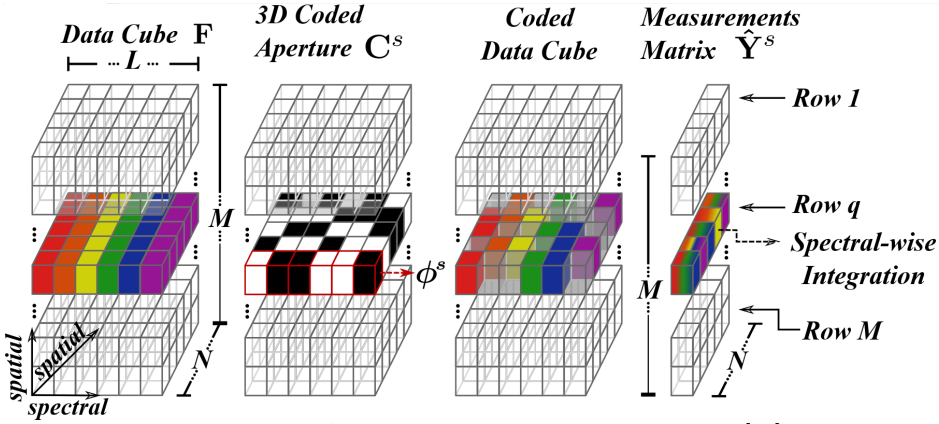

Figure 2: 3D-CASSI CSI architecture scheme. Source [30]

Then, $\mathbf{s} = \boldsymbol{\Psi}\boldsymbol{\theta}$, where $\boldsymbol{\theta} \in \mathbb{R}^{MNLT}$ are the sparse coefficients.

The sensing matrix $\mathbf{H}$ for CSVS is a block diagonal concatenation of independent sensing matrices $\left\{\mathbf{H}^t, t = 1, ..., T \mid \mathbf{H}^t \in \mathbb{R}^{K \times MNL}, K \ll MNL\right\}$, as [19]

$$\begin{bmatrix} \mathbf{y}^1 \\ \vdots \\ \mathbf{y}^t \\ \vdots \\ \mathbf{y}^T \end{bmatrix} = \begin{bmatrix} \mathbf{H}^1 & 0 & \cdots & 0 & 0 \\ & \ddots & & & \\ \vdots & & \mathbf{H}^t & & \vdots \\ & & & \ddots & \\ 0 & 0 & \cdots & 0 & \mathbf{H}^T \end{bmatrix} \begin{bmatrix} \mathbf{s}^1 \\ \vdots \\ \mathbf{s}^t \\ \vdots \\ \mathbf{s}^T \end{bmatrix}, \tag{1}$$

where $\mathbf{H} \in \mathbb{R}^{KT \times MNLT}$ and $\mathbf{y} \in \mathbb{R}^{KT}$. The recovery of an SV from compressed measurements had been typically addressed as the minimization problem

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2} \left\|\mathbf{H}\boldsymbol{\Psi}\boldsymbol{\theta} - \mathbf{y}\right\|_2^2 + \lambda \left\|\boldsymbol{\theta}\right\|_1, \tag{2}$$

where λ is a regularization constant that controls the trade-off between the data-fitting term and the sparsity-

inducing term [20]. To improve the performance of the SSR recovery model, Lopez *et. al.* [19] recently proposed to include an additional regularization term based on the optical flow as

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{H}\boldsymbol{\Psi}\boldsymbol{\theta} - \mathbf{y}\|_2^2 + \lambda \|\boldsymbol{\theta}\|_1 + \beta \|\Delta\|_2^2, \tag{3}$$

where β is a regularization term and $\Delta = \Lambda(\bar{\mathbf{f}}_w)_{i,j} - \Lambda(\bar{\mathbf{f}}_z)_{i+u,j+v}$ is expressed in terms of the horizontal and vertical changes estimated from two upsampled and contiguous frames $\bar{\mathbf{f}}_w$ and $\bar{\mathbf{f}}_z$, obtained from a low-resolution reconstructed version of the original SV.

2.3. Convolutional Sparse Coding (CSC)

The CSC is a synthesis framework for sparsely representing signals using a collection of convolutional dictionary elements and sparse coefficient maps, both learned directly from the signal of interest [31]. This signal specificity allows for higher reconstruction qualities. CSC states that a given signal $\mathbf{s} \in \mathbb{R}^N$ can be represented as the sparse combination of a collection of dictionary elements $\{\mathbf{d}_m,\ m = 1, ..., M_d \mid \mathbf{d}_m \in \mathbb{R}^d\}$ and its corresponding sparse coefficient maps $\{\mathbf{x}_m,\ m = 1, ..., M_d \mid \mathbf{x}_m \in \mathbb{R}^N\}$, as

$$\mathbf{s} = \sum_{m=1}^{M_d} \mathbf{d}_m \overset{1}{*} \mathbf{x}_m + \boldsymbol{\omega}, \tag{4}$$

where $\boldsymbol{\omega}$ denotes a reconstruction error. It is worth noting that the sparse coefficient maps have the same dimension as $\mathbf{s}$, while the dictionary elements are much smaller, $d \ll N$. The structure of the convolution operation makes the representation framework robust to noise, shifting, and deformation of the features within the represented signal. These properties make CSC a useful framework for denoising and machine learning [32].

Wohlberg *et al.* [33] proposes to use the CSC framework for representing gray-scale images $\mathbf{s} \in \mathbb{R}^{M \times N}$ using a collection of dictionary elements (features) $\{\mathbf{D}_m,\ m = 1, ..., M_d \mid \mathbf{D}_m \in \mathbb{R}^{d_M \times d_N}\}$ with $d \lll M, N$, and sparse coefficient maps (abundances) $\{\mathbf{X}_m,\ m = 1, ..., M_d \mid \mathbf{X}_m \in \mathbb{R}^{M \times N}\}$. However, considering the nature of the overcomplete collection of dictionary elements, the collection of sparse coefficient maps cannot be obtained by a direct inversion of the dictionary collection, but utilizing the minimization scheme

$$\underset{\{\mathbf{X}_m\}}{\operatorname{argmin}} \frac{1}{2} \left\| \sum_{m=1}^{M_d} \mathbf{D}_m \overset{2}{*} \mathbf{X}_m - \mathbf{s} \right\|_2^2 + \lambda \sum_{m=1}^{M_d} \|\mathbf{X}_m\|_1 . \tag{5}$$

From this point it is easy to extend to color images where an RGB image is treated as a stack of gray-scale images, $\mathbf{S} \in \mathbb{R}^{M \times N \times 3}$. Wohlberg *et al.* proposed the convolutional basis pursuit denoising (CBPDN) [34] as a low computational cost alternative for the convolutional representation of a stack of gray-scale images

by solving

$$\underset{\{\mathbf{X}_{c,m}\}}{\operatorname{argmin}} \frac{1}{2} \sum_{c=1}^{3} \left\| \sum_{m=1}^{M_d} \mathbf{D}_m \overset{2}{*} \mathbf{X}_{c,m} - \mathbf{S}_c \right\|_F^2 + \lambda \sum_{c=1}^{3} \sum_{m=1}^{M_d} \|\mathbf{X}_{c,m}\|_1 + \mu \|\mathbf{X}_{c,m}\|_{2,1} , \tag{6}$$

where $\{\mathbf{X}_{m,c},\ m = 1, ..., M_d\ ,\ c = 1, 2, 3 \mid \mathbf{X}_{m,c} \in \mathbb{R}^{M \times N}\}$ are the sparse coefficient maps for each dictionary element indexed by m and the c^{th} channel of the RGB image; while $\{\mathbf{D}_m,\ m = 1, ..., M_d \mid \mathbf{D}_m \in \mathbb{R}^{d_M \times d_N}\}$ is a collection of dictionary elements. CBPDN represents the spatial correlation within each color channel in an RGB image but misses the correlation between channels.

For the case of SIs, $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{M \times N \times L}$ datasets with both spatial and spectral correlations, we proposed the CSC3D framework in [22], which uses a single 3D convolution operation to include the spatial-spectral correlations as

$$\boldsymbol{\mathcal{S}} = \sum_{m=1}^{M_d} \boldsymbol{\mathcal{D}}_m \overset{3}{*} \boldsymbol{\mathcal{X}}_m + \boldsymbol{\Omega}, \tag{7}$$

where $\{\boldsymbol{\mathcal{D}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{D}}_m \in \mathbb{R}^{d_M \times d_N \times d_L}\}$ is a collection of 3D convolutional dictionary elements, $\{\boldsymbol{\mathcal{X}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{X}}_m \in \mathbb{R}^{M \times N \times L}\}$ is a collection of 3D sparse coefficient maps, and $\boldsymbol{\Omega}$ represents the reconstruction error.

To include Eq. (7) within a minimization scheme, we proposed to express it as the single linear operation

$$\mathbf{s} = \mathbf{vec}(\boldsymbol{\mathcal{S}}) = \bar{\mathbf{D}}\mathbf{x} + \boldsymbol{\omega}, \tag{8}$$

where $\bar{\mathbf{D}} = [\bar{\mathbf{D}}_1 ... \bar{\mathbf{D}}_{M_d}] \in \mathbb{R}^{MNL \times MNLM_d}$ is the concatenation of the equivalent convolutional matrices $\bar{\mathbf{D}}_m \in \mathbb{R}^{MNL \times MNL}$ and $\mathbf{x} = [\mathbf{vec}(\boldsymbol{\mathcal{X}}_1)^T ... \mathbf{vec}(\boldsymbol{\mathcal{X}}_{M_d})^T]^T \in \mathbb{R}^{MNL}$ is the vertical concatenation of the vectorized coefficient maps. Then, we formulate the minimization scheme as

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\| \bar{\mathbf{D}}\mathbf{x} - \mathbf{s} \right\|_2^2 + \lambda \|\mathbf{x}\|_1 . \tag{9}$$

This new and extended CSC framework has found applications in denoising [21] SIs and CSI, outperforming the SSR model [23]. Now, considering the versatility of the CSC model for representing 1D signals, grayscale and RGB images, and recently 3D SIs, then it leads us to think of a general model for representing multidimensional signals as

$$\boldsymbol{\mathcal{S}} = \sum_{m=1}^{M_d} \boldsymbol{\mathcal{D}}_m \overset{N}{*} \boldsymbol{\mathcal{X}}_m + \boldsymbol{\Omega}, \tag{10}$$

where $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{L_1 \times ... \times L_N}$, $\{\boldsymbol{\mathcal{D}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{D}}_m \in \mathbb{R}^{d_1 \times ... \times d_N}\}$ and $\{\boldsymbol{\mathcal{X}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{X}}_m \in \mathbb{R}^{L_1 \times ... \times L_N}\}$, with $d_i \lll L_i$.

Equation (10) is the basis for the formulation of the CSC4D proposed in [25], a 4D CSC model for representing SV's spatial-spectral-temporal correlations within a single operation. SVs can be considered

as 4D datasets $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{M\times\times N\times L\times T}$ containing the 3D spatial-spectral information for each spectral frame $\boldsymbol{\mathcal{S}}^t \in \mathbb{R}^{M\times N\times L}$, alongside the time variation $t = 1, ..., T$. The CSC formulation for SVs can then be expressed as

$$\boldsymbol{\mathcal{S}} = \sum_{m=1}^{M_d} \boldsymbol{\mathcal{D}}_m \overset{4}{*} \boldsymbol{\mathcal{X}}_m + \boldsymbol{\Omega}, \tag{11}$$

where $\{\boldsymbol{\mathcal{D}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{D}}_m \in \mathbb{R}^{d_M\times d_N\times d_L\times d_T}\}$, is a collection of 4D convolutional dictionary elements containing the collection of features within $\boldsymbol{\mathcal{S}}$, $\{\boldsymbol{\mathcal{X}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{X}}_m \in \mathbb{R}^{M\times N\times L\times T}\}$ is a collection of 4D sparse maps representing the spatial-spectral-temporal contributions for each feature, $\boldsymbol{\Omega}$ represents the reconstruction error, and $d_M, d_N, d_L, d_T \lll M, N, L, T$ respectively.

Just as in the 3D case, the sum of 4D convolutions profit from the commutativity property of the convolution operation to rewrite it as a linear operator

$$\mathbf{s} = \mathbf{vec}(\boldsymbol{\mathcal{S}}) = \sum_{m=1}^{M_d} \bar{\mathbf{D}}_m \mathbf{x}_m = \sum_{m=1}^{M_d} \bar{\mathbf{X}}_m \mathbf{d}_m = \bar{\mathbf{D}}\mathbf{x} = \bar{\mathbf{X}}\mathbf{d} \in \mathbb{R}^{MNLT}, \tag{12}$$

where $\bar{\mathbf{D}}_m, \bar{\mathbf{X}}_m \in \mathbb{R}^{MNLT\times MNLT}$ are equivalent convolutional matrices; $\mathbf{d}_m = \mathbf{vec}(\bar{\mathbf{D}}_m)$ and $\mathbf{x}_m = \mathbf{vec}(\bar{\mathbf{x}}_m) \in \mathbb{R}^{MNLT}$ are their corresponding vectorizations. Operators $\bar{\mathbf{D}}$,$\bar{\mathbf{X}} \in \mathbb{R}^{MNLT\times MNLTM_d}$ are linear operators equivalent to the sum of 4D cyclic convolutions and $\mathbf{x}, \mathbf{d} \in \mathbb{R}^{MNLTM_d}$ are vectorizations. We create the equivalent linear operators as horizontal concatenations, i.e. $\bar{\mathbf{D}} = [\bar{\mathbf{D}}_1\ ...\ \bar{\mathbf{D}}_{M_d}]$, of linear convolutional equivalents; while the vectorizations are vertical concatenations of the M_d vectorizations, i.e. $\mathbf{x} = [\mathbf{vec}(\boldsymbol{\mathcal{X}}_1)^\mathbf{T}\ ...\ \mathbf{vec}(\boldsymbol{\mathcal{X}}_{M_d})^\mathbf{T}]^\mathbf{T}$.

3. CSC4D in a CSVS Framework

The linear representation of CSC4D proposed in [25] can be included within the CSVS recovery minimization in Eq. (2), replacing the SSR model with the CSC model as a representation basis, as

$$\underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \left\|\mathbf{H}\bar{\mathbf{D}}\mathbf{x} - \mathbf{y}\right\|_2^2 + \lambda \left\|\mathbf{x}\right\|_1. \tag{13}$$

Equation 13 aims to learn a set of sparse coefficient maps $\mathbf{x}$ from compressed measurements $\mathbf{y}$, given a fixed convolutional dictionary $\bar{\mathbf{D}}$ and a CSVS sensing matrix $\mathbf{H}$. Considering the specificity of the CSC model, the dictionary elements $\bar{\mathbf{D}}$ must also be learned from the compressed measurements by solving the minimization problem

$$\underset{\mathbf{d}}{\operatorname{argmin}} \frac{1}{2} \left\|\mathbf{H}\bar{\mathbf{X}}\mathbf{d} - \mathbf{y}\right\|_2^2 + \iota_{C_Z}(\mathbf{d}). \tag{14}$$

Eq. (13) and (14) are formulated in a similar and complementary fashion. Eq. (13) is formulated as a ℓ_2-linear restriction and learns a collection of sparse coefficient maps from a fixed collection of dictionary

elements; while Eq. (14) is formulated as a ℓ_2 - linear restriction, and learns a collection of dictionary elements from a fixed collection of sparse coefficient maps. The linear restriction in Eq. (13) is the ℓ_1 norm, while the linear restriction in Eq. (14) is an indicator function based in a constraint set (see Section 3.2.).

Both Eq. (13) and (14) are known as the CSC4D+CSVS model and are solved alternately to recover a full version of the SV of interest from compressed measurements. It is worth noting that $\bar{\mathbf{D}}$ and $\bar{\mathbf{X}}$ should not be considered, under any circumstances, as bases. Due to its rectangular size, its inverse must be obtained through a minimization scheme. Note that the proposed CSC4D+CSVS does not require estimating a low-resolution version of the original SV to include the optical flow information, as in Eq. (3), proposed by Lopez *et. al.* [19]. Having fewer elements in the mathematical formulation simplifies its solution.

Creating the operators $\bar{\mathbf{D}}$ and $\bar{\mathbf{X}}$ is computationally expensive and non-optimal, requiring a more convenient alternative as explained in [23]. Due to the general N-dimensional CSC model in Eq. (10), Eqs. (13) and (14) are similar to the formulation of CSC3D in [23]. However, the new proposed operators $\bar{\mathbf{D}}$ and $\bar{\mathbf{X}}$ in Eq. (12) contain four dimensions instead of the three dimensions used in CSC3D. The additional dimension taxes heavily the numerical cost of the CSC4D+CSVS formulation, requiring new optimization routines instead of simply scaling up CSC3D's routines. Subsections 3.1 and 3.2 expand on the analytical solution of Eq. (13) and (14), respectively.

The compress sensing matrix $\mathbf{H}$ in CSC4D+CSVS seems similar to the decimation matrix in CSD4D+SR [25]. However, there are some considerations to take into account which differentiate one formulation from the other, to mention:

- About the domain of operation, the SR decimation matrix operates only on the spatial dimension, with $\mathbf{H} \in \mathbb{R}^{M_s N_s LT \times MNLT}$, $M_s = M/\alpha$, $N_s = N/\alpha$ and $\alpha \in \mathbb{N}^*$ being a decimation factor; while the CSVS sensing matrix can operate over both spatial and spectral dimensions, according to the CSVS architecture used, with $\mathbf{H} \in \mathbb{R}^{KT \times MNLT}$ and $K \ll MNL$.

- About the compression size, the SR case was tested successfully for $\alpha = 2$, which results in $\mathbf{H}$'s rows being a quarter of the number of columns; this means a compression of 25%. For the CSVS case, the compression was tested successfully for as low as 18.75%.

- About the mathematical properties, for the CSVS case we can profit on $\mathbf{H}\mathbf{H}^{\mathbf{T}} = \mathbf{I}$, which is not true for the SR case.

Matrix $\mathbf{H}$ is the main difference between both CSC4D+SR and CSC4D+CSVS formulations, though they seem similar on paper. Matrix $\mathbf{H}$ is responsible for generating either the low resolution SV $\dot{\mathbf{s}}$ or the compressed measurements $\mathbf{y}$, respectively, which are the second main difference between both formulations. These two elements force to change half the solutions presented bellow.

3.1. Reconstruction Update Problem (RU)

Eq. (13) is called the Reconstruction Update problem (RU) and aims to learn the spatial-spectral-temporal contributions of each one of the elements in a given convolutional dictionary collection. Considering the nature and size of the product $\mathbf{H}\bar{\mathbf{D}}$, we propose to solve it using the alternating directions multiplier method, ADMM [35], by introducing two auxiliary variables as

$$\begin{aligned} \underset{\mathbf{x},\mathbf{u},\mathbf{v}}{\operatorname{argmin}} & \frac{1}{2}\left\|\mathbf{H}\mathbf{u}-\mathbf{y}\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1, \\ \text{s.t.: } & \mathbf{u}=\bar{\mathbf{D}}\mathbf{x}, \\ & \mathbf{v}=\mathbf{x}. \end{aligned} \tag{15}$$

Considering the model for the N-dimensional CSC framework in Eq. (10), we propose to profit on CSC3D formulation [23] and extend it on the required update steps. Then, considering the augmented Lagrangian for Eq. (15) given by

$$\mathcal{L}\{\mathbf{x},\mathbf{u},\mathbf{v},\mathbf{f},\mathbf{g}\}=\frac{1}{2}\left\|\mathbf{H}\mathbf{u}-\mathbf{y}\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1+\frac{\rho}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{u}+\mathbf{f}\right\|_2^2+\frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}+\mathbf{g}\right\|_2^2, \tag{16}$$

we obtain the update steps

$$\mathbf{x}^{(j+1)}:=\underset{\mathbf{x}}{\operatorname{argmin}}\ \frac{\rho}{2}\left\|\bar{\mathbf{D}}\mathbf{x}-\mathbf{u}^{(j)}+\mathbf{f}^{(j)}\right\|_2^2+\frac{\rho}{2}\left\|\mathbf{x}-\mathbf{v}^{(j)}+\mathbf{g}^{(j)}\right\|_2^2, \tag{17}$$

$$\mathbf{u}^{(j+1)}:=\underset{\mathbf{u}}{\operatorname{argmin}}\ \frac{1}{2}\left\|\mathbf{H}\mathbf{u}-\mathbf{y}\right\|_2^2+\frac{\rho}{2}\left\|\bar{\mathbf{D}}\mathbf{x}^{(j+1)}-\mathbf{u}+\mathbf{f}^{(j)}\right\|_2^2, \tag{18}$$

$$\mathbf{v}^{(j+1)}:=\underset{\mathbf{v}}{\operatorname{argmin}}\ \frac{\rho}{2}\left\|\mathbf{x}^{(j+1)}-\mathbf{v}+\mathbf{g}^{(j)}\right\|_2^2+\lambda\left\|\mathbf{v}\right\|_1, \tag{19}$$

$$\mathbf{f}^{(j+1)}=\mathbf{f}^{(j)}+\bar{\mathbf{D}}\mathbf{x}^{(j+1)}-\mathbf{u}^{(j+1)}, \tag{20}$$

$$\mathbf{g}^{(j+1)}=\mathbf{g}^{(j)}+\mathbf{x}^{(j+1)}-\mathbf{v}^{(j+1)}, \tag{21}$$

where $\mathbf{f}$ and $\mathbf{g}$ are the so called dual variables, and the super indexes (j) and $(j+1)$ refer to the iteration steps.

Solutions to Eqs. (17) to (19) are presented bellow according to the derivations presented in Appendix A to Appendix C

$$\hat{\mathbf{x}}^{(j+1)}=\left[\hat{\mathbf{b}}-\hat{\bar{\mathbf{D}}}^{\mathbf{H}}\left(\mathbf{I}+\hat{\bar{\mathbf{D}}}\hat{\bar{\mathbf{D}}}^{\mathbf{H}}\right)^{-1}\hat{\bar{\mathbf{D}}}\hat{\mathbf{b}}\right], \tag{22}$$

$$\mathbf{u}^{(j+1)}=\frac{1}{\rho}\left[\mathbf{b}-\left(\frac{1}{\rho+1}\right)\mathbf{H}^{\mathbf{T}}\mathbf{H}\mathbf{b}\right], \tag{23}$$

$$\mathbf{v}^{(j+1)}=\mathcal{S}_{\frac{\lambda}{\rho}}\left(\mathbf{x}^{(j+1)}+\mathbf{g}^{(j)}\right). \tag{24}$$

3.2. Feature Extraction Problem (FE)

Just as the sparse coefficient maps, the collection of convolutional dictionary elements is also learned from the compressed measurements by solving the minimization in Eq. (14) called the Feature Extraction problem (FE). However, instead of the ℓ_1 restriction, FE aims to obtain a collection of 1-norm small dictionary elements using the indicator function $\iota_{\mathcal{C}_Z}$, as explained in Appendix D. Given the nature and size of the product $\mathbf{H}\bar{\mathbf{X}}$, Eq. (14) is also solved using ADMM by introducing two auxiliary variables

$$\begin{aligned} &\underset{\mathbf{d},\mathbf{p},\mathbf{q}}{\operatorname{argmin}} \frac{1}{2}\left\|\mathbf{H}\mathbf{p}-\mathbf{y}\right\|_2^2 + \iota_{\mathcal{C}_Z}(\mathbf{q}), \\ &\text{s.t.: } \mathbf{p} = \bar{\mathbf{X}}\mathbf{d}, \\ &\qquad \mathbf{q} = \mathbf{d}, \end{aligned} \tag{25}$$

with augmented Lagrangian

$$\mathcal{L}\{\mathbf{d},\mathbf{p},\mathbf{q},\mathbf{r},\mathbf{t}\} = \frac{1}{2}\left\|\mathbf{H}\mathbf{p}-\mathbf{y}\right\|_2^2 + \iota_{\mathcal{C}_Z}(\mathbf{q}) + \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{p}+\mathbf{r}\right\|_2^2 + \frac{\sigma}{2}\left\|\mathbf{d}-\mathbf{q}+\mathbf{t}\right\|_2^2, \tag{26}$$

and updates

$$\mathbf{d}^{(j+1)} := \underset{\mathbf{d}}{\operatorname{argmin}} \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}-\mathbf{p}^{(j)}+\mathbf{r}^{(j)}\right\|_2^2 + \frac{\sigma}{2}\left\|\mathbf{d}-\mathbf{q}^{(j)}+\mathbf{t}^{(j)}\right\|_2^2, \tag{27}$$

$$\mathbf{p}^{(j+1)} := \underset{\mathbf{p}}{\operatorname{argmin}} \frac{1}{2}\left\|\mathbf{H}\mathbf{p}-\mathbf{y}\right\|_2^2 + \frac{\sigma}{2}\left\|\bar{\mathbf{X}}\mathbf{d}^{(j+1)}-\mathbf{p}+\mathbf{r}^{(j)}\right\|_2^2, \tag{28}$$

$$\mathbf{q}^{(j+1)} := \underset{\mathbf{q}}{\operatorname{argmin}} \frac{\sigma}{2}\left\|\mathbf{d}^{(j+1)}-\mathbf{q}+\mathbf{t}^{(j)}\right\|_2^2 + \iota_{\mathcal{C}_Z}(\mathbf{q}), \tag{29}$$

$$\mathbf{r}^{(j+1)} = \mathbf{r}^{(j)} + \bar{\mathbf{X}}\mathbf{d}^{(j+1)} - \mathbf{p}^{(j)}, \tag{30}$$

$$\mathbf{t}^{(j+1)} = \mathbf{t}^{(j)} + \mathbf{d}^{(j+1)} - \mathbf{t}^{(j)}. \tag{31}$$

where $\mathbf{r}$ and $\mathbf{t}$ are the so called dual variables, and the superindexes j and $j+1$ refer to the iteration step.

Solutions to Eqs. (27) to (29) are presented bellow according to the derivations presented in Appendix E to Appendix G

$$\hat{\mathbf{d}}^{(j+1)} = \left[\hat{\mathbf{b}} - \hat{\bar{\mathbf{X}}}^{\mathbf{H}}\left(\mathbf{I} + \hat{\bar{\mathbf{X}}}\hat{\bar{\mathbf{X}}}^{\mathbf{H}}\right)^{-1}\hat{\bar{\mathbf{X}}}\hat{\mathbf{b}}\right]. \tag{32}$$

$$\mathbf{p}^{(j+1)} = \frac{1}{\sigma}\left[\mathbf{b} - \left(\frac{1}{\sigma+1}\right)\mathbf{H}^{\mathbf{T}}\mathbf{H}\mathbf{b}\right], \tag{33}$$

$$\mathbf{q}_m^{(j+1)} = \frac{\mathbf{Z}_p\mathbf{Z}_p^{\mathbf{T}}(\mathbf{d}_m^{(j+1)} + \mathbf{t}_m^{(j)})}{\left\|\mathbf{Z}_p\mathbf{Z}_p^{\mathbf{T}}(\mathbf{d}_m^{(j+1)} + \mathbf{t}_m^{(j)})\right\|_2}, \tag{34}$$

3.3. Proposed CSC4D+CSVS Algorithm

The CSC framework achieves improved reconstruction qualities by learning both collections of features and spatial-spectral-temporal contributions directly from one desired signal. The State-of-the-art SSR model uses a predefined analysis model, limiting the versatility range of the recovered signals; CSC4D solves this limitation by solving both RU and FE problems alternately, with one's updates feeding the other.

Algorithm 1 Convolutional Spectral Coding for Compressive Spectral Video Sensing - CSC4D+CSVS

Require: $\{\boldsymbol{\mathcal{X}}_m^{(0)} \in \mathbb{R}^{M\times N\times L\times T}\}$ as zeros and $\{\boldsymbol{\mathcal{D}}_m^{(0)} \in \mathbb{R}^{d_M\times d_N\times d_L\times d_T}\}$ as random $\forall m = 1, ..., M_d$; M, N, L, T, M_d, d_M, $d_N, d_L, d_T, \rho_0, \lambda$ and σ_0.

Set $j = 0$.
Set $\{\boldsymbol{\mathcal{V}}_m^{(j)}\} = \{\boldsymbol{\mathcal{X}}_m^{(j)}\}$ and build the vectorization $\mathbf{v}^{(j)}$.
Set $\{\boldsymbol{\mathcal{Q}}_m^{(j)}\} = \{\boldsymbol{\mathcal{D}}_m^{(j)}\}$ and build the vectorization $\mathbf{q}^{(j)}$.
Solve $\boldsymbol{\mathcal{S}}^{(j)} = \sum_{m=1}^{M_d} \boldsymbol{\mathcal{D}}_m^{(j)} \overset{4}{*} \boldsymbol{\mathcal{X}}_m^{(j)}$.
Set $\boldsymbol{\mathcal{U}}^{(j)} = \boldsymbol{\mathcal{P}}^{(j)} = \boldsymbol{\mathcal{S}}^{(j)}$ and build the respective vectorizations $\mathbf{u}^{(j)}$ and $\mathbf{p}^{(j)}$.
Set $\mathbf{f}^{(j)} = \mathbf{g}^{(j)} = \mathbf{r}^{(j)} = \mathbf{t}^{(j)} =$ zeros
Build $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from $\{\boldsymbol{\mathcal{D}}_m^{(j)}\}$.
Build $\hat{\mathbf{x}}^{(j)}$ from $\{\boldsymbol{\mathcal{X}}_m^{(j)}\}$
Set $\rho^{(j)} = \rho_0$ and $\sigma^{(j)} = \sigma_0$
repeat
RU
Solve $\hat{\mathbf{x}}^{(j+1)}$ using Eq. (22) and build $\mathbf{x}^{j+1}$ from $\hat{\mathbf{x}}^{(j+1)}$
Solve $\mathbf{u}^{(j+1)}$ using Eq. (23)
Solve $\mathbf{v}^{(j+1)}$ using Eq. (24)
Update $\mathbf{f}^{(j+1)}$ and $\mathbf{g}^{(j+1)}$ using Eq. (20) and (21), respectively
Update $\rho^{(j+1)}$ according to [35], section 3.3
Fold $\mathbf{v}^{(j+1)}$ into $\{\boldsymbol{\mathcal{V}}_m^{(j+1)}\}$ and build $\bar{\mathbf{X}}$ and $\hat{\bar{\mathbf{X}}}$ from it.
FE
Solve $\hat{\mathbf{d}}^{(j+1)}$ using Eq. (E.1) and build $\mathbf{d}^{j+1}$ from $\hat{\mathbf{d}}^{(j+1)}$
Solve $\mathbf{p}^{(j+1)}$ using Eq. (33)
Solve $\mathbf{q}^{(j+1)}$ using Eq. (34)
Update $\mathbf{r}^{(j+1)}$ and $\mathbf{t}^{(j+1)}$ using Eq. (30) and (31), respectively
Update $\sigma^{(j+1)}$ according to [35], section 3.3
Fold $\mathbf{q}^{(j+1)}$ into $\{\boldsymbol{\mathcal{Q}}_m^{(j+1)}\}$ and build $\bar{\mathbf{D}}$ and $\hat{\bar{\mathbf{D}}}$ from it
until a stopping criteria is satisfied
return Collection of sparse coefficient maps $\{\boldsymbol{\mathcal{V}}_m^{(j+1)}\}$ and convolutional dictionary elements $\{\boldsymbol{\mathcal{Q}}_m^{(j+1)}\}$.

3.4. Estimated Numerical Complexity

We will now estimate the numerical complexity of the more complex subproblems in RU and FE. Subproblem (19)'s solution, Eq. (24), is obtained from a soft-thresholding problem, and subproblem (29)'s solution, Eq. (34), is akin to a hard-thresholding problem. Both complexities are negligible.

Subproblems (18) and (28) have the same solution structure where the inversion of $\mathbf{H^T H} + \alpha\mathbf{I} \in \mathbb{R}^{MNLT \times MNLT}$ represents the highest complexity. According to Appendix A of [23], their complexity can be reduced from $\mathcal{O}((MNLT)^3)$ to $\mathcal{O}(\tilde{M}T(MNLT)^2)$, with $\tilde{M} \ll MNL$ and $\mathbf{H} \in \mathbb{R}^{\tilde{M}T \times MNLT}$, by profiting from $\mathbf{H}$'s block diagonal structure and $\mathbf{H}\mathbf{H}^T = \mathbf{I}$.

One of the main sources of numerical complexity is the 4D convolutional operation, with complexity $\mathcal{O}((MNLT)^2)$, considering that the dictionary elements are zero-padded to match the dimensions of the sparse coefficient maps. The 4D convolution complexity is reduced to a fraction by expressing it as a Hadamard product, profiting from the Discrete Fourier Transform (DFT) Theorem, reducing the cost to $\mathcal{O}(MNLT\log(MNLT))$.

Finally, the greatest source of numerical complexity is the inversions in Eqs. (A.3) and (E.1), as solutions to subproblems (17) and (27). Again, the canonical complexity for inverting $\hat{\mathbf{A}}^H\hat{\mathbf{A}} + \alpha\mathbf{I} \in \mathbb{R}^{MNLTM_d \times MNLTM_d}$ is $\mathcal{O}((MNLTM_d)^3)$. However, by profiting on the concatenated diagonal structure of $\hat{\hat{\mathbf{D}}}$ and $\hat{\hat{\mathbf{X}}}$ and the dimensions rearrangement exposed in Appendix B of [23], the inversion complexity falls to $\mathcal{O}(MNLTM_d)$. Table 2 summarizes the different complexities for the stated subproblems and their solutions.

Eq.	Complexity	Solution
(19) and (29)	Negible	Implemented
(18) and (28)	$\mathcal{O}((MNLT)^3)$	Original
(23) and (33)	$\mathcal{O}(\tilde{M}T(MNLT)^2)$	Implemented
(17) and (27)	$\mathcal{O}((MNLTM_d)^3)$	Original
(A.3) and (E.1)	$\mathcal{O}(MNLTM_d)$	Implemented

Table 2: Complexity review of the proposed CSC4D+CSVS algorithm.

For the SSR case, the complexity is set by the matrix $\mathbf{\Psi}$ = 2D Wavelet $\otimes$ DCT $\otimes$ DCT. The complexity of the 2D wavelet transform can be estimated, in the worst-case scenario for a whole set of filters and levels, as $\mathcal{O}((MN)^2)$. The spectral DCT has complexity $\mathcal{O}(L\log(L))$ and the temporal DCT has complexity $\mathcal{O}(T\log(T))$. The numerical complexity of the whole Kronecker product can be estimated as $\mathcal{O}((MN)^2LT\log(L)\log(T))$.

The complexity of CSC4D+CSVS is determined by the solutions to Eqs. (23) and (33) $\mathcal{O}(\tilde{M}T(MNLT)^2)$, plus the fact that the CSC framework demands the solution of two (2) minimization schemes instead of one (1) for the SSR case. However, experimental results evidence that, although being more complex to solve, the proposed CSC4D+CSVS converges quicker to a viable solution.

4. Performance Evaluation

This research work tested the performance of the proposed CSC4D in CSVS following a two-step scheme:

1. *Recovery quality from compressed measurements:* The first step was to assess the recovery quality of the CSC4D from compressed measurements. For this case we use two collections of independent sensing matrices from two CSI architectures, 3D-CASSI [29] and C-CASSI [20], without the presence of noise, and compared to the SSR model. Specifically, we compared CSC4D to the classical CSVS recovery model, without the additional optical flow regularization term, considering that CSC4D does not use the optical flow information either.
2. *Robustness to acquisition noise:* The second step was to assess the recovery quality of the CSC4D from compressed measurements in the presence of noise. We use two noise models, Gaussian white noise and Poisson noise, at different intensities, the latter to simulate sensing noise. Again, we compared the performance of the proposed CSC4D to the classical CSVS model.

The quality of the sparse representation of SVs using a 4D CSC model was tested successfully in [25], improving by up to 20dB PSNR the SSR model at sparsity levels below 10%, as shown in Figure 3. For this reason, this research work does not include that particular performance evaluation.

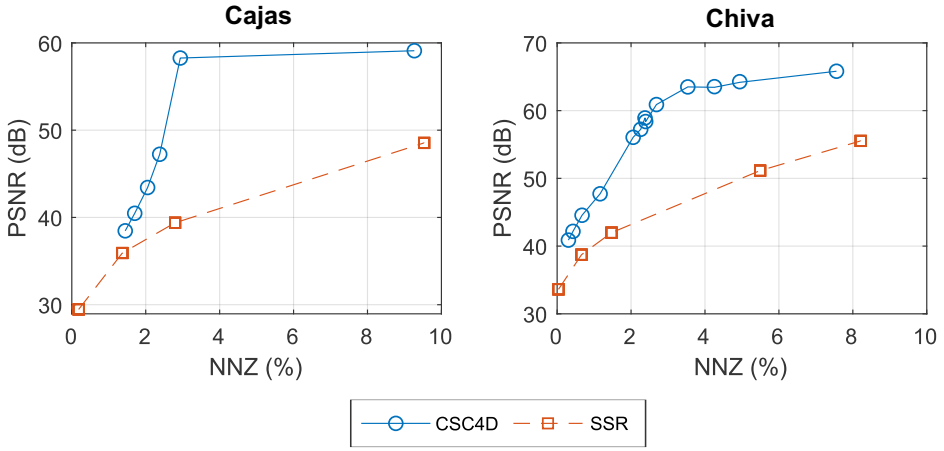

Figure 3: Reconstruction quality performance (PSNR) for various levels of sparsity (NNZ), for the proposed CSC4D and SSR model. Taken from [25].

The two-step test scheme was conducted using two laboratory-captured spectral videos, Cajas and Chiva [19] (see Fig 4). Both data sets have spatial resolution 128×128, 16 spectral bands, and 8 frames. Considering that convolutional dictionaries have a low performance for representing low-frequency components of multidimensional signals [36], this work uses the high-frequency versions of the original datasets, which are obtained by performing a high-pass filtering stage to the image data. For illustrative purposes, we add the low-frequency components to evaluate visually the recovered SVs.

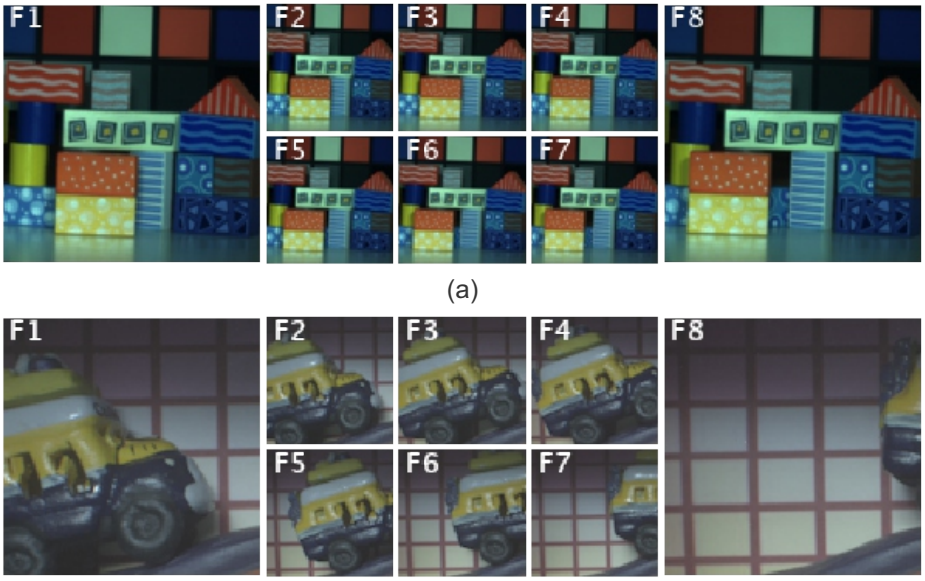

Figure 4: False RGB of the test datasets (a) Cajas and (b) Chiva SVs.

The CSC approach is compared against the state-of-the-art SSR approach in all three steps of the performance test using the following metrics:

- the peak signal-to-noise ratio (PSNR) for measuring the overall reconstruction quality,
- the structural similarity index (SSIM) for edge sharpness, and
- the percentage of non-zero elements (NNZ) for measuring sparsity.

Considering that the convolutional coefficient maps are a collection of M_d sparse tetrahedrons $\in \mathbb{R}^{M\times N\times L\times T}$, compared to the single sparse tetrahedron of the SSR model, then the sparsity of the convolutional coefficient maps will be measured as

$$\text{sparsity} = \max_{m=1}^{M_d} \frac{\|\boldsymbol{\mathcal{X}}_m\|_0}{MNLT}. \tag{35}$$

This is, the sparsity of the convolutional solutions will be the maximum sparsity of the individual coefficient maps. Following a series of previous experimental results, the sizes for the dictionary collection were fixed to $d_M = d_N = d_L = 8$, $d_T = 3$, and $M_d = 30$, for the entire scheme test.

4.1. Performance Using Noiseless Measurements from the 3D-CASSI and C-CASSI CSI Architecture

The first step of the test scheme is to assess the performance of CSC4D at recovering full versions of compressed sensed SVs. First, we test CSC4D's using compressed measurements obtained using the 3D-CASSI CSI architecture. For this experiment we create collections of independently generated sensing matrices $\left\{\mathbf{H}^t, t = 1, ..., 8 \mid \mathbf{H}^t \in \mathbb{R}^{K.128.128\times 128.128.16}\right\}$, where $K = \{3, 4\}$ are the number of shots per spectral

frame, resulting in a compression of 18.75% and 25% respectively for both scenarios. Then, the sensing matrix can be defined as $\mathbf{H} \in \mathbb{R}^{K.128.128.8 \times 128.128.16.8}$, according to Eq. (1). Table 3 presents the results for both the CSC4D+CSVS and the SSR model. CSC4D outperforms the SSR model at all compression ratios with both datasets.

Dataset	K	Metric	CSC4D+CSVS	SSR
Cajas	3	PSNR	**36,30**	35,00
		SSIM	**0,960**	0,929
	4	PSNR	**36,53**	35,58
		SSIM	**0,961**	0,943
Chiva	3	PSNR	**51,83**	50,76
		SSIM	**0,996**	0,974
	4	PSNR	**54,60**	52,91
		SSIM	**0,997**	0,994

Table 3: Mean PSNR and mean SSIM of the reconstructed datasets using the 3D-CASSI, for both CSC4D+CSVS and SSR framework, and two compression ratios. The standard deviations for five repetitions were well bellow 1% for both mean PSNR and mean SSIM, for this reason they are not shown.

Figures 5 and 6 show some example reconstructed frames as false RGB from compressed measurements taken with $K = 4$. Figures 7 and 8 show the error between the original SVs and the reconstructed versions from both models. Note the overall errors of the SSR model at recovering both textures and border details; while the frames recovered by the CSC4D+CSVS framework exhibit a higher overall quality and increased border sharpness, although still exist some missing specific details.

Second, we test the performance of the proposed CSC4D at recovering SVs from compressed measurements sensed using the C-CASSI CSI architecture. Again, we create a collection of independently generated sensing matrices $\left\{\mathbf{H}^t, t = 1, ..., 8 \mid \mathbf{H}^t \in \mathbb{R}^{K.128.(128+16-1) \times 128.128.16}\right\}$, where $K = \{3, 4\}$ are the number of shots per spectral frame, giving a compression of 20.94% and 27.92%, respectively, for both scenarios, according to $\gamma = KM(N + L - 1)/MNL$ [20]. Then, the sensing matrix can be defined as $\mathbf{H} \in \mathbb{R}^{K.128.(128+16-1).8 \times 128.128.16.8}$, according to Eq. (1). Preliminary experiments showed that CSC4D performed poorly when integrated with the C-CASSI CSI architecture. We observed the same behavior with the CSC3D framework and the C-CASSI [23].

An exhaustive analysis of the minimization routines yielded one possible cause: the interaction between the C-CASSI sensing matrix and the linear operators $\bar{\mathbf{D}}$ and $\bar{\mathbf{X}}$. Both linear operators are matrix representations of the cyclic convolution operation, therefore they have very defined structures which include the circular shifting effect. Consider the band-shifting structure of a single C-CASSI sensing matrix $\mathbf{H}^T$, depicted in Figure 9, and the circular shifting effect of matrix $\bar{\mathbf{D}}$. When interacting with the optical dispersion element represented in C-CASSI's sensing matrix, the circular shifting effect of the equivalent cyclic convolutional matrices is replicated off-site, as shown in Figure 10, adding noise to the recovery routine.

To solve this issue, we recreated the side information scheme used in [23]. The proposed side information system creates a gray-scale version of the original SV and replicates each gray-scale image at each band,

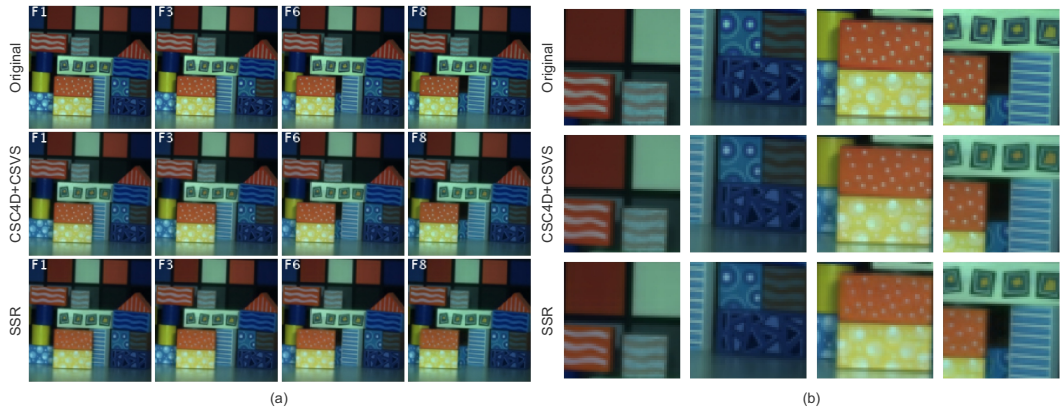

Figure 5: Example recovered frames from 3D-CASSI compressed measurements, at $K = 4$, with the CSC4D and SSR methods for the dataset (a) Cajas and (b) some close-up details.

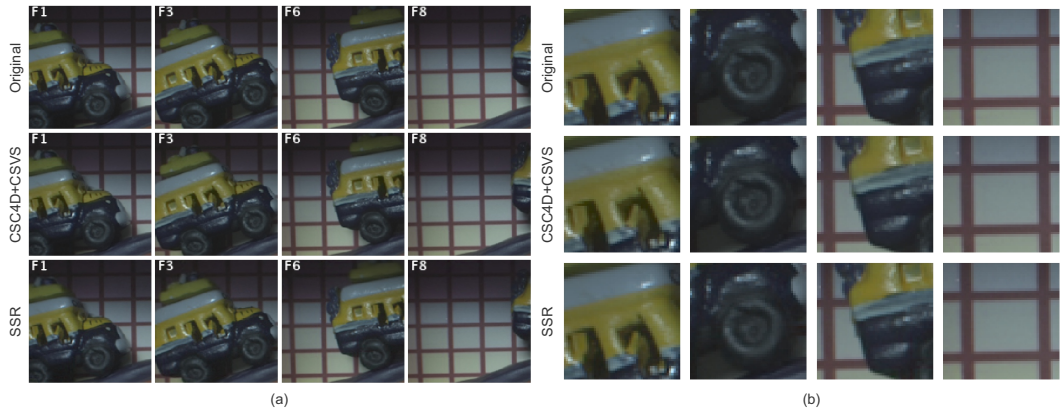

Figure 6: Example recovered frames from 3D-CASSI compressed measurements, at $K = 4$, with the CSC4D and SSR methods for the dataset (a) Chiva and (b) some close-up details.

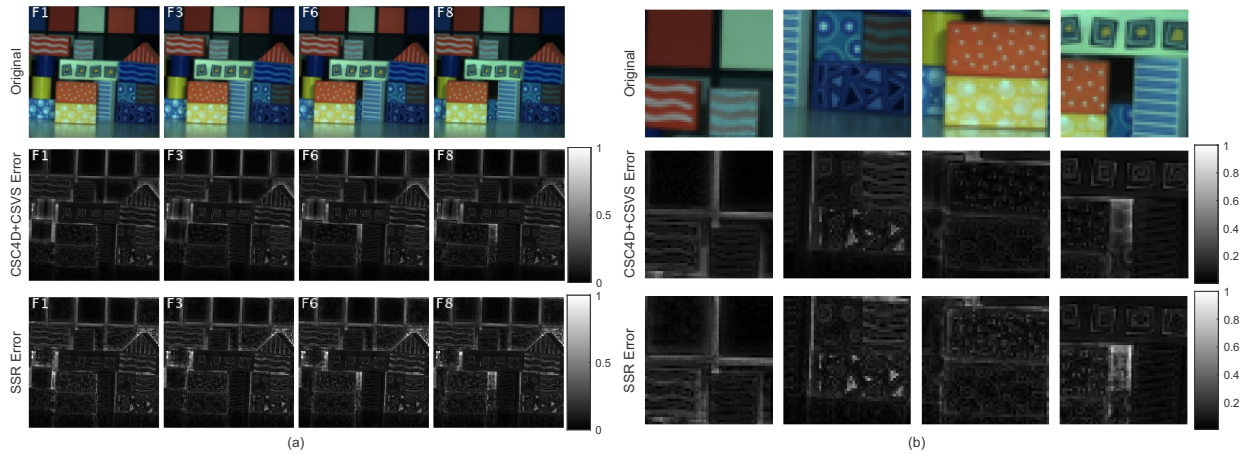

Figure 7: Reconstructed error frames recovered from 3D-CASSI compressed measurements, at $K = 4$, with the CSC4D and SSR methods for the dataset (a) Cajas and (b) some close-up details.

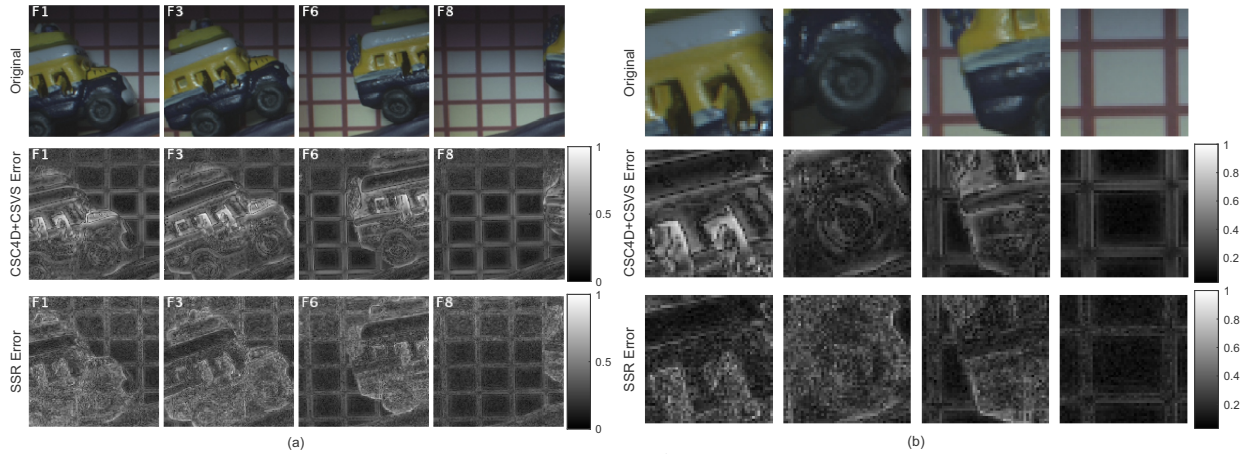

Figure 8: Reconstructed error frames recovered from 3D-CASSI compressed measurements, at $K = 4$, with the CSC4D and SSR methods for the dataset (a) Chiva and (b) some close-up details.

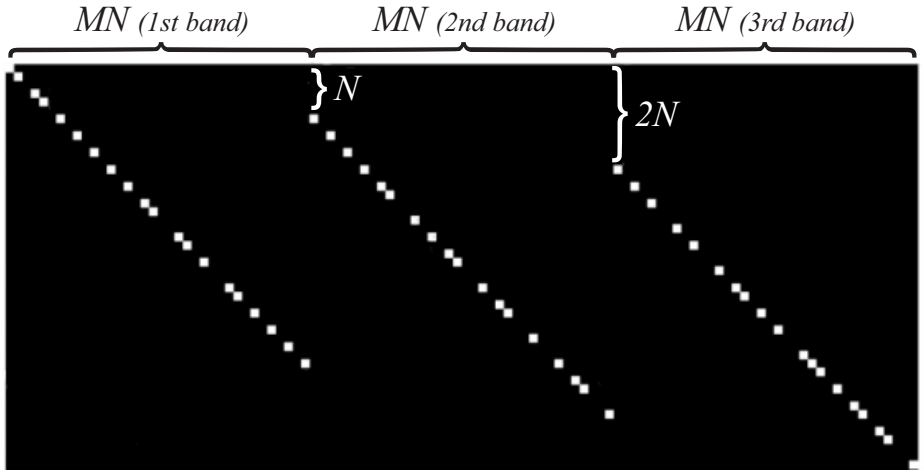

Figure 9: Detail of the band shifting for a single C-CASSI sensing matrix $\mathbf{H}^t$. Taken from [23]

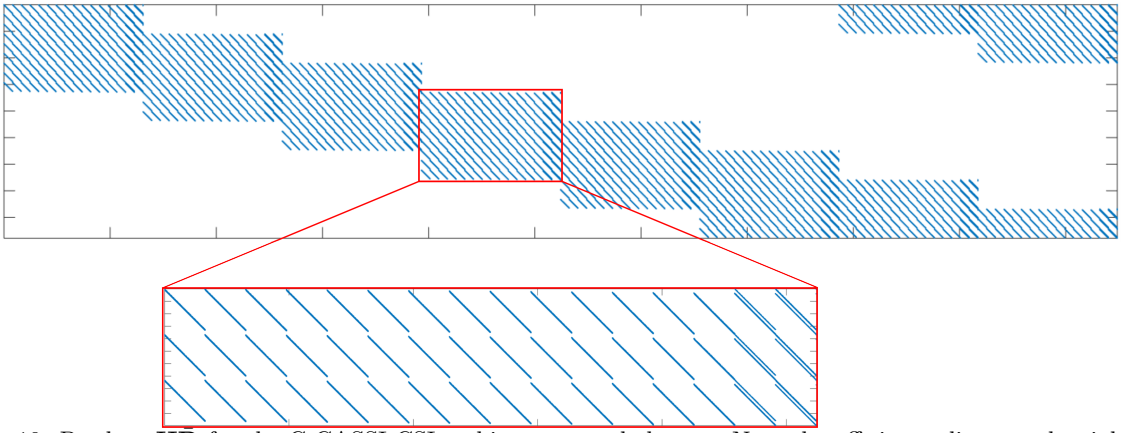

Figure 10: Product $\mathbf{HD}$ for the C-CASSI CSI architecture, and closeup. Note the off-site replicas on the right side of the closeup.

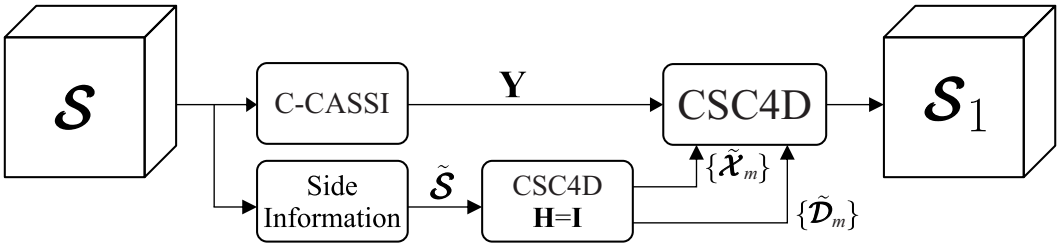

Figure 11: Side information scheme. Taken from [23]

Dataset	K	Metric	CSC4D	SSR
Cajas	3	PSNR	**36,2**1	34,30
		SSIM	**0,935**	0,905
	4	PSNR	**36,4**5	35,49
		SSIM	**0,957**	0,919
Chiva	3	PSNR	**45,53**	39,11
		SSIM	**0,97**4	0,928
	4	PSNR	**46,75**	41,69
		SSIM	**0,981**	0,951

Table 4: Mean PSNR of the reconstructed datasets using the C-CASSI, for both CSC4D and SSR framework and two compression ratios. The standard deviations for five repetitions were well bellow 1% for both mean PSNR and mean SSIM, thus are not shown.

so $\tilde{\mathcal{S}} \in \mathbb{R}^{M\times N\times L\times T}$. Note that $\tilde{\mathcal{S}}$ contains all the spatial patterns, structures, and correlations of the original SV, but no spectral information. This additional information helps overcome the noise generated by the dispersive element in C-CASSI. Optimal initial collections $\left\{\tilde{\mathcal{D}}_m\right\}$ $\left\{\tilde{\mathcal{X}}_m\right\}$ are created from $\tilde{\mathcal{S}}$ and used as initializations for recovering $\mathcal{S}_1$, as shown in Figure 11. The results of recovering both datasets using C-CASSI plus the proposed side information scheme are shown in Table 4, where the CSC4D model outperforms the state-of-the-art model.

4.2. Robustness to acquisition noise

The last step in the two-step test scheme is to assess the performance of the CSC4D algorithm in the presence of noise. This research work used two noise models: Gaussian white noise and Poisson noise, the latter representing acquisition noise. We added three different noise levels to the compressed measurements: 10dB, 20dB, and 30dB PSNR for Gaussian; 18dB, 23dB, and 28dB for Poisson. Again, both CSI architectures were used, with two compression ratios, for both SVs. The side information scheme was coupled again with the C-CASSI architecture. The results of the performance tests are shown in Figures 12, 13 and Tables 5 to 8.

Just as in the CSC3D framework, the CSC4D model down-performs in the presence of excessive noise (10dB Gaussian and 18dB Poisson), matching the performance of the SSR model. However, it outperforms the state-of-the-art model at medium and low noise levels, at all compression ratios, up to 4dB. On the other hand, the SSIM values for the CSC4D model are higher than SSR's SSIM values at almost all compression

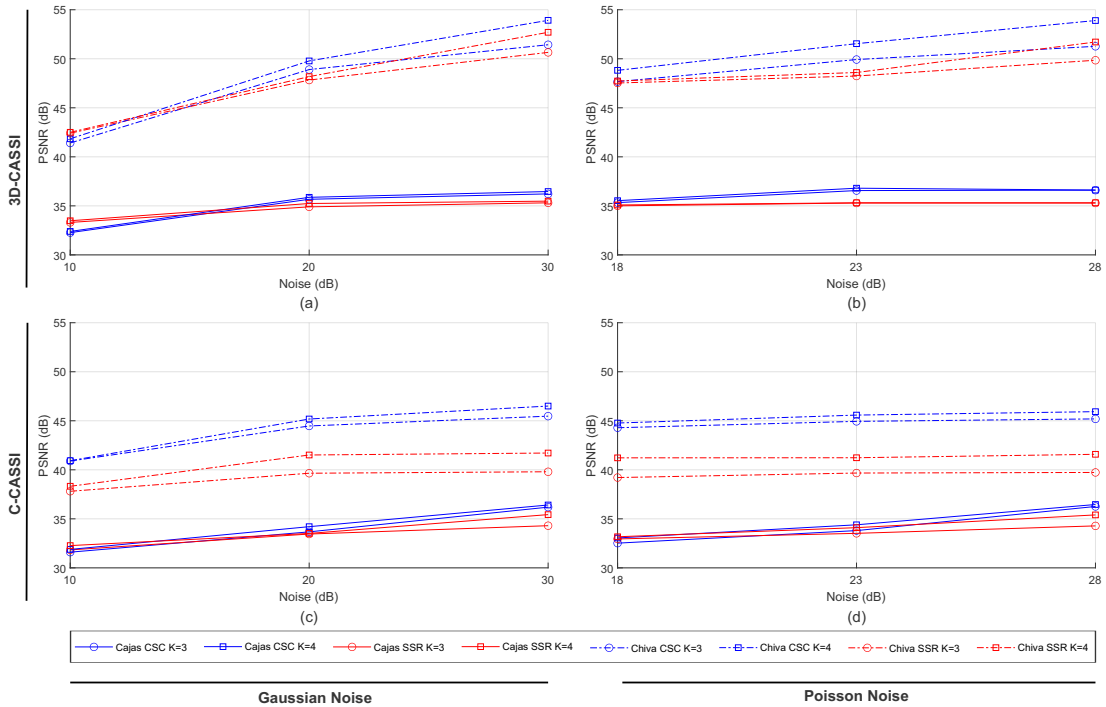

Figure 12: PSNR comparison for the Cajas (solid line) and Chiva (dotted line), *CSC* framework (blue line), *SSR* framework (red line), K=3 (□) and K=4 (○), for two CSVS techniques and two noise types.

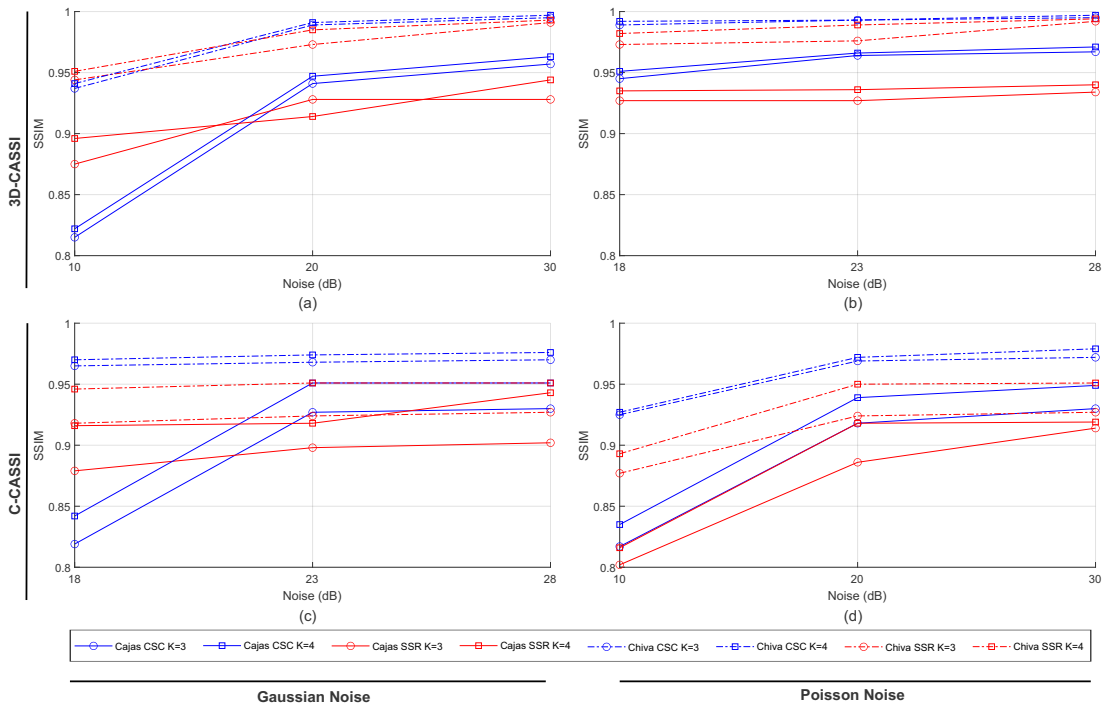

Figure 13: SSIM comparison for the Cajas (solid line) and Chiva (dotted line), *CSC* framework (blue line), *SSR* framework (red line), K=3 (□) and K=4 (○), for two CSVS techniques and two noise types.

Dataset	Noise (dB)	K	Metric	CSC4D	SSR
Cajas	10	3	PSNR	32,27	**33,31**
			SSIM	0,815	**0,875**
		4	PSNR	32,39	**33,48**
			SSIM	0,822	**0,896**
	20	3	PSNR	**35,67**	34,99
			SSIM	**0,941**	0,928
		4	PSNR	**35,87**	35,24
			SSIM	**0,947**	0,914
	30	3	PSNR	**36,22**	35,31
			SSIM	**0,957**	0,928
		4	PSNR	**36,46**	35,48
			SSIM	**0,963**	0,944
Chiva	10	3	PSNR	41,41	**42,41**
			SSIM	0,937	**0,944**
		4	PSNR	41,84	**42,50**
			SSIM	0,941	**0,951**
	20	3	PSNR	**48,90**	47,83
			SSIM	**0,989**	0,973
		4	PSNR	**49,78**	48,18
			SSIM	**0,991**	0,985
	30	3	PSNR	**51,43**	50,65
			SSIM	**0,995**	0,991
		4	PSNR	**53,92**	52,70
			SSIM	**0,997**	0,993

Table 5: Mean PSNR and SSIM of the reconstructed datasets using the 3D-CASSI in presence of Gaussian white noise, for both CSC4D and SSR framework and two compression ratios.

Dataset	Noise (dB)	K	Metric	CSC4D	SSR
Cajas	18	3	PSNR	**35,33**	35,00
			SSIM	**0,945**	0,927
		4	PSNR	**35,53**	35,09
			SSIM	**0,951**	0,935
	23	3	PSNR	**36,56**	35,29
			SSIM	**0,964**	0,927
		4	PSNR	**36,80**	35,31
			SSIM	**0,966**	0,936
	28	3	PSNR	**36,60**	35,29
			SSIM	**0,967**	0,934
		4	PSNR	**36,61**	35,31
			SSIM	**0,971**	0,940
Chiva	18	3	PSNR	**47,67**	47,53
			SSIM	**0,989**	0,973
		4	PSNR	**48,82**	47,73
			SSIM	**0,992**	0,982
	23	3	PSNR	**49,92**	48,24
			SSIM	**0,993**	0,976
		4	PSNR	**51,54**	48,60
			SSIM	**0,996**	0,989
	28	3	PSNR	**51,27**	49,85
			SSIM	**0,995**	0,992
		4	PSNR	**53,90**	51,71
			SSIM	**0,997**	0,994

Table 6: Mean PSNR and SSIM of the reconstructed datasets using the 3D-CASSI in presence of Poisson noise, for both CSC4D and SSR framework and two compression ratios.

Dataset	Noise (dB)	K	Metric	CSC4D	SSR
Cajas	10	3	PSNR	31,60	**31,83**
			SSIM	**0,817**	0,802
		4	PSNR	31,88	**32,27**
			SSIM	**0,835**	0,816
	20	3	PSNR	**33,67**	33,45
			SSIM	**0,918**	0,886
		4	PSNR	**34,20**	33,51
			SSIM	**0,939**	0,918
	30	3	PSNR	**36,19**	34,30
			SSIM	**0,930**	0,904
		4	PSNR	**36,41**	35,43
			SSIM	**0,949**	0,919
Chiva	10	3	PSNR	**40,89**	37,81
			SSIM	**0,925**	0,877
		4	PSNR	**40,93**	38,32
			SSIM	**0,927**	0,893
	20	3	PSNR	**44,47**	39,65
			SSIM	**0,969**	0,924
		4	PSNR	**45,18**	41,51
			SSIM	**0,972**	0,950
	30	3	PSNR	**45,47**	39,80
			SSIM	**0,972**	0,927
		4	PSNR	**46,50**	41,71
			SSIM	**0,979**	0,951

Table 7: Mean PSNR and SSIM of the reconstructed datasets using the C-CASSI in presence of Gaussian white noise, for both CSC4D and SSR framework and two compression ratios.

Dataset	Noise (dB)	K	Metric	CSC4D	SSR
Cajas	18	3	PSNR	32,52	**32,95**
			SSIM	0,819	**0,879**
		4	PSNR	33,05	**33,17**
			SSIM	0,842	**0,916**
	23	3	PSNR	**33,80**	33,52
			SSIM	**0,927**	0,898
		4	PSNR	**34,39**	34,10
			SSIM	**0,951**	0,918
	28	3	PSNR	**36,26**	34,28
			SSIM	**0,930**	0,902
		4	PSNR	**36,46**	35,40
			SSIM	**0,950**	0,943
Chiva	18	3	PSNR	**44,29**	39,21
			SSIM	**0,965**	0,918
		4	PSNR	**44,77**	41,22
			SSIM	**0,970**	0,946
	23	3	PSNR	**44,94**	39,67
			SSIM	**0,968**	0,924
		4	PSNR	**45,58**	41,23
			SSIM	**0,974**	0,951
	28	3	PSNR	**45,19**	39,73
			SSIM	**0,970**	0,927
		4	PSNR	**45,93**	41,58
			SSIM	**0,976**	0,951

Table 8: Mean PSNR and SSIM of the reconstructed datasets using the C-CASSI in presence of Poisson noise, for both CSC4D and SSR framework and two compression ratios.

The standard deviations for five repetitions, in Tables 3 to 7, were well bellow 1% for both mean PSNR and mean SSIM, thus are not shown.

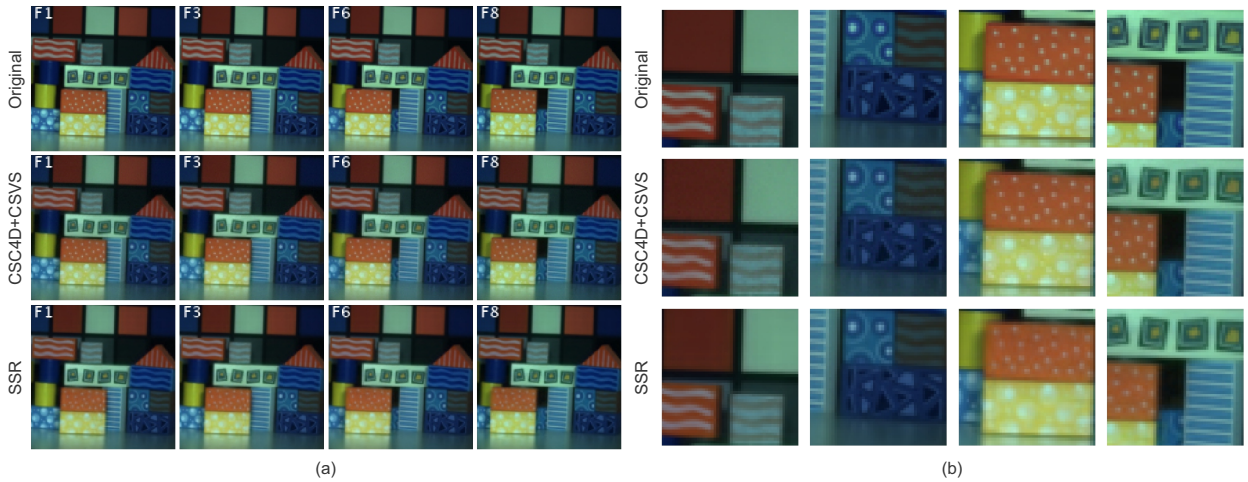

Figure 14: Example reconstructed frames recovered from 3D-CASSI compressed measurements, at $K = 4$ and 20dB SNR Gauss, with the CSC4D and SSR methods for the dataset (a) Cajas and (b) and some close-up details.

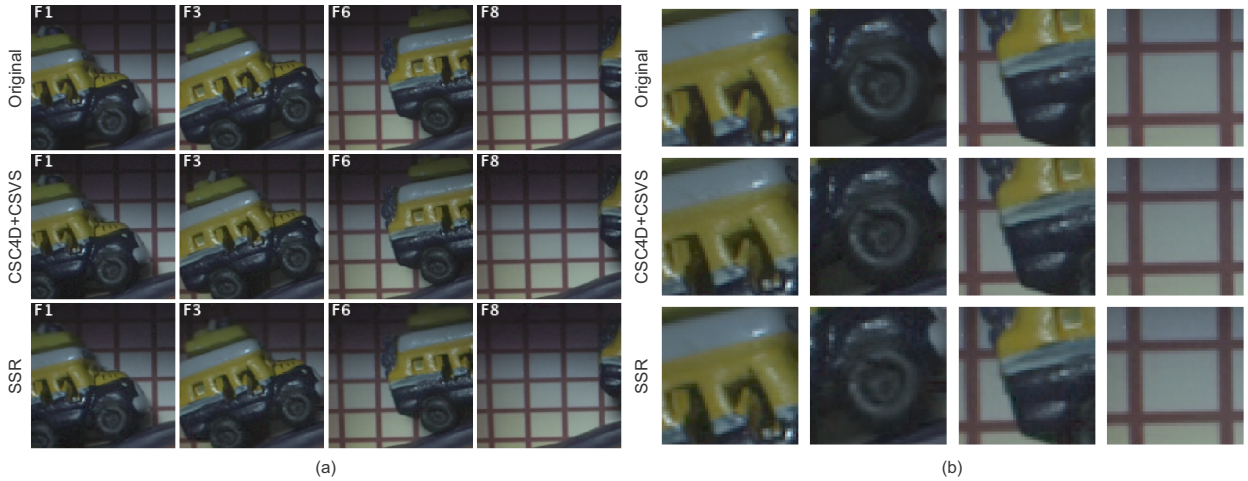

Figure 15: Example reconstructed frames recovered from 3D-CASSI compressed measurements, at $K = 4$ and 20dB SNR Gauss, with the CSC4D and SSR methods for the dataset (a) Chiva and (b) and some close-up details.

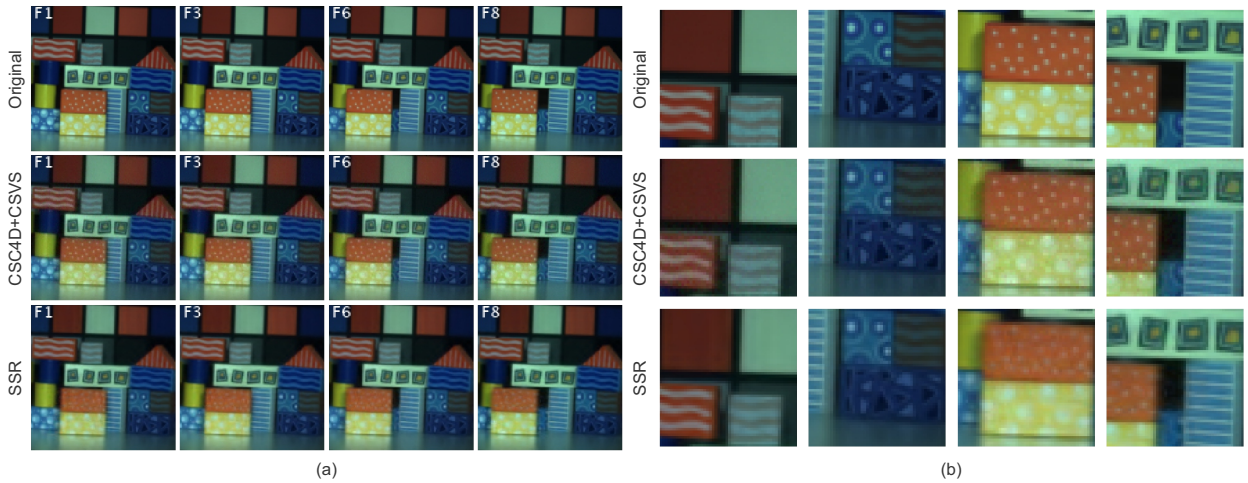

Figure 16: Example reconstructed frames recovered from C-CASSI compressed measurements, at $K = 4$ and 23dB SNR Poisson, with the CSC4D and SSR methods for the dataset (a) Cajas and (b) some close-up details.

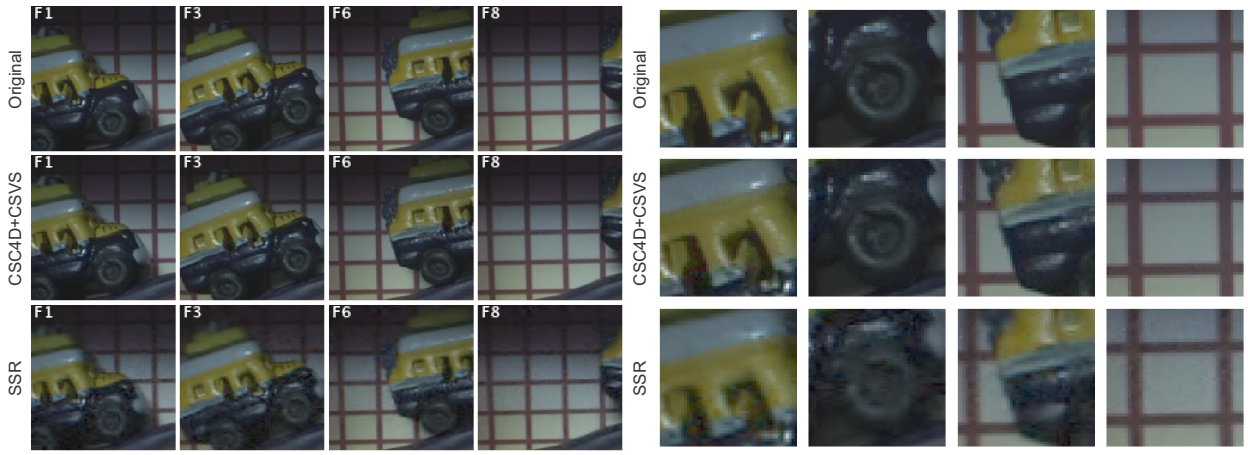

Figure 17: Example reconstructed frames recovered from C-CASSI compressed measurements, at $K = 4$ and 23dB SNR Poisson, with the CSC4D and SSR methods for the dataset (a) Chiva and (b) some close-up details.

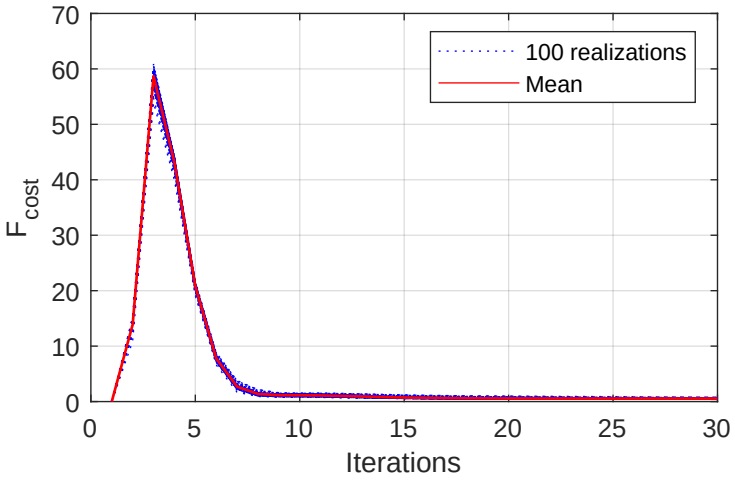

Figure 18: Mean behavior of the cost function after 100 realizations.

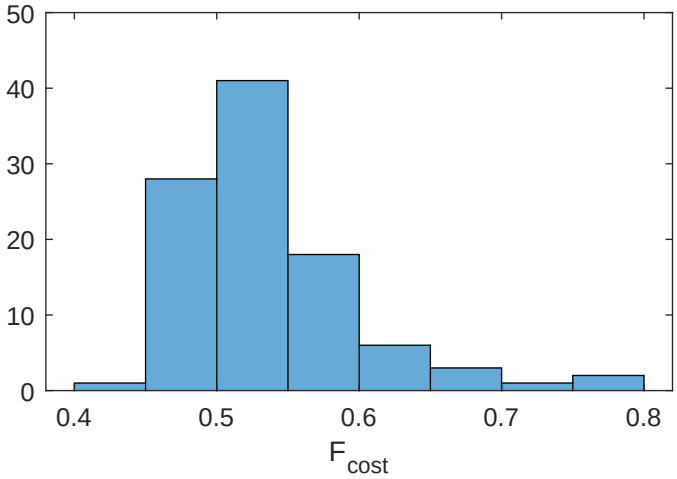

Figure 19: Histogram of the cost function at random initializations

and noise levels. This means that, although CSC4D fails to recover all the SV's spatial-spectral-temporal features in the presence of noise, the recovered SV has sharper and more defined borders than the SSR model.

Figures 14 to 17 show some example reconstructed frames as false RGB, recovered from compressed measurements taken with $K = 4$ and 20dB SNR Gauss and 23dB SNR Poisson noise, respectively. Note the overall quality of the recovered individual spectral frames, and the border sharpness, of the CSC4D model when compared to the SSR model.

4.3. Algorithm Convergence

To illustrate the good convergence of Algorithm 1, we selected the evolution of the cost function (13) as function of the number of iterations. Considering that the indicator function in (14) only has values 0 or ∞, it is not included in the estimation of the cost function. The high-frequency version of the Cajas dataset was used for this experiment, and the compressive measurements were simulated with the 3D-CASSI system, with 20dB SNR white Gaussian noise and $K = 4$. Fig. 18 confirms the convergence of the algorithm to a critical point of the objective function.

To analyze the sensitivity to initialization, we ran the proposed algorithm with 100 different random initializations for the collection of dictionary elements, and all-zero coefficient maps. The histogram of the corresponding values of the objective function is shown in Fig. 19, presenting a single mode, confirming the convexity of the proposed algorithm. The mode of the objective function histogram corresponds to a PSNR of 35.61dB.

5. Conlusions

The proposed Convolutional Sparse Coding (CSC4D) was successfully integrated within a CSVS recovery scheme, giving as result the CSC4D+CSVS algorithm. The proposed algorithm was tested using two different CSI SV architectures. When integrated with a 3D-CASSI CSVS architecture, the CSC4D+CSVS model is able to match and outperform the state-of-the-art approach at different compression ratios and noise levels, for a couple of test SVs. On the other hand, when the CSC4D+CSVS model is integrated with the C-CASSI CSVS architecture, it is not able to unmix the C-CASSI band-shifting; it requires a better initial set of convolutional dictionary elements for preserving spatial features. With the suggested C-CASSI Side Information approach, the CSC4D+CSVS model is able to outperform the state-of-the-art SSR approach at various compression ratios and noise levels. The CSC4D+CSVS algorithm proved to be robust when dealing with Poisson acquisition noises, improving the state-of-the-art SSR model.

6. Acknowledgements

Ph.D. (c) Crisostomo Barajas-Solano is supported by the MINCIENCIAS scholarship #785.

References

[1] G. Lu, B. Fei, Medical hyperspectral imaging: A review, J. Biomed. Opt 19 (2014) 10901.

[2] G. A. Shaw, H. K. Burke, Spectral imaging for remote sensing, Lincoln laboratory journal 14 (2003) 3–28.

[3] D. Manolakis, D. Marden, G. A. Shaw, Hyperspectral image processing for automatic target detection applications, Lincoln Lab. J 14 (2003) 79–116.

[4] S. Y. Cheng, S. Park, M. M. Trivedi, Multi-spectral and multiperspective video arrays for driver body tracking and activity analysis, Comput. Vis. Image Underst. 106 (2007) 245–257.

[5] H. Van-Nguyen, A. Banerjee, R. Chellappa, Tracking via object reflectance using a hyperspectral video camera, 2010, pp. 44–51.

[6] A. Banerjee, P. Burlina, J. Broadwater, Hyperspectral video for illumination-invariant tracking, 2009.

[7] R. Leitner, M. De-Biasio, T. Arnold, C. V. Dinh, M. Loog, R. P. W. Duin, Multi-spectral video endoscopy system for the detection of cancerous tissue, Pattern Recognition Letters 34 (2013) 85–93.

[8] K. J. Zuzak, S. C. Naik, G. Alexandrakis, D. Hawkins, K. Behbehani, E. Livingston, Intraoperative bile duct visualization using nearinfrared hyperspectral video imaging, 2013, pp. 145–150.

[9] D. Yi, L. Kong, F. Wang, F. Liu, S. Sprigle, A. Adibi, Instrument an off-shelf ccd imaging sensor into a handheld multispectral video camera, Photonics Technology Letters, IEEE 23 (2011) 606–608.

[10] C. V. Correa, C. A. Hinojosa, G. R. Arce, H. Arguello, Multiple snapshot colored compressive spectral imager, Optical Engineering 56 (2016) 041309:1–041309:10.

[11] R. G. Sellar, G. D. Boreman, Classification of imaging spectrometers for remote sensing applications, Opt. Eng. 44 (2005) 013602–1 – 013602–2.

[12] N. Gat, Imaging spectroscopy using tunable filters, 2000, pp. 50–64.

[13] B. Buttingsrud, B. K. Alsberg, Superresolution of hyperspectral images, Chemometrics and Intelligent Laboratory Systems 84 (2006) 62–68. doi:10.1016/j.chemolab.2006.04.014.

[14] C. Kwan, J. H. Choi, S. Chan, J. Zhou, B. Budavari, Resolution enhancement for hyperspectral images: A super-resolution and fusion approach, 2017, pp. 6180–6184. doi:10.1109/ICASSP.2017.7953344.

[15] C. V. Correa, H. Arguello, G. R. Arce, Snapshot colored compressive spectral imager, JOSA A 32 (2015) 1754–1763.

[16] H. Arguello, G. R. Arce, Rank minimization code aperture design for spectrally selective compressive imaging, IEEE Transactions on Image Processing 22 (2013) 941–954. doi:10.1109/TIP.2012.2222899.

[17] A. Barducci, D. Guzzi, C. Lastri, P. Marcoionni, V. Nardino, I. Pippi, Compressive sensing and hyperspectral imaging, 2012, p. 105642Z.

[18] Y. Wang, Q. Yao, J. T. Kwok, L. M. Ni, Scalable online convolutional sparse coding, IEEE Transactions on Image Processing 27 (2018) 4850–4859. doi:10.1109/TIP.2018.2842152.

[19] K. M. Leon-Lopez, L. V. G. Carreno, H. A. Fuentes, Temporal colored coded aperture design in compressive spectral video sensing, IEEE Transactions on Image Processing 28 (2019) 253–264. doi:10.1109/TIP.2018.2867171.

[20] G. R. Arce, D. J. Brady, L. Carin, H. Arguello, D. S. Kittle, Compressive coded aperture spectral imaging: An introduction, IEEE Signal Processing Magazine 31 (2014) 105–115. doi:10.1109/MSP.2013.2278763.

[21] C. Barajas-Solano, J. M. Ramirez, H. Garcia, H. Arguello, Tridimensional convolutional sparse coding of spectral images, Vol. 2019, 2019. doi:10.1364/hise.2019.htu3b.5.

[22] C. Barajas-Solano, H. Garcia, H. Arguello, Convolutional basis pursuit denoising of spectral images using a tri-dimensional sparse representation, IEEE, 2019, pp. 1–5. doi:10.1109/STSIVA.2019.8730285.

[23] C. Barajas-Solano, J.-M. Ramirez, H. Arguello, Convolutional sparse coding framework for compressive spectral imaging, Journal of Visual Communication and Image Representation 66 (2019) 1–15. doi:10.1016/j.jvcir.2019.102690.
URL https://doi.org/10.1016/j.jvcir.2019.102690

[24] V. Papyan, Y. Romano, J. Sulam, M. Elad, Theoretical foundations of deep learning via sparse representations: A multilayer sparse model and its connection to convolutional neural networks, IEEE Signal Processing Magazine 35 (2018) 72–89.

[25] C. Barajas-Solano, J.-M. Ramirez, H. A. Fuentes, Spectral video compression using convolutional sparse coding, 2020.

[26] L. Galvis, D. Lau, X. Ma, H. Arguello, G. R. Arce, Coded aperture design in compressive spectral imaging based on side information, Applied Optics 56 (2017) 6332–6340.

[27] X. Yuan, T. han Tsai, R. Zhu, P. Llull, D. Brady, L. Carin, Compressive hyperspectral imaging with side information, IEEE Journal of Selected Topics in Signal Processing 9 (2015) 964–976. doi:10.1109/JSTSP.2015.2411575.

[28] C. V. Correa, H. Arguello, G. R. Arce, Compressive spectral imaging with coloredpatterned detectors, 2014, pp. 7789–7793.

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

[30] C. Hinojosa, J. Bacca, H. Arguello, Coded aperture design for compressive spectral subspace clustering, IEEE Journal of Selected Topics in Signal Processing 12 (2018) 1589–1600.

[31] A. M. Bruckstein, D. L. Donoho, M. Elad, From sparse solutions of systems of equations to sparse modeling of signals and images, SIAM Review 51 (2009) 34–81.

[32] V. Papyan, J. Sulam, M. Elad, Working locally thinking globally: Theoretical guarantees for convolutional sparse coding, IEEE Transactions on Signal Processing 65 (2017) 5687–5701. doi:10.1109/TSP.2017.2733447.

[33] B. Wohlberg, Efficient algorithms for convolutional sparse representations, IEEE Transactions on Image Processing 25 (2016) 301–315.

[34] B. Wohlberg, Convolutional sparse representation of color images, 2016, pp. 57–60.

[35] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, Foundations and Trends in Machine Learning 3 (2010) 1–122.

[36] B. Wohlberg, Boundary handling for convolutional sparse representations, 2016, pp. 1833–1837.

[37] H. V. Henderson, S. R. Searle, On deriving the inverse of a sum of matrices, Siam Review 23 (1981) 53–60.

[38] R. T. Rockafellar, Convex Analysis, Princeton University Press, 1970.

Appendix A. Coefficients Maps Update

It should be noted that solving Eq. (17) requires the costly creation of operator $\bar{\mathbf{D}} \in \mathbb{R}^{MNLT \times MNLTM_d}$, and its transpose. By profiting on the Discrete Fourier Transform (DFT) theorem for the 4D case

$$\sum_{m=1}^{M_d} \boldsymbol{\mathcal{D}}_m \overset{4}{*} \boldsymbol{\mathcal{X}}_m = \mathcal{F}_{4D}^{-1}\left(\sum_{m=1}^{M_d} \mathcal{F}_{4D}(\boldsymbol{\mathcal{D}}_m) \odot \mathcal{F}_{4D}(\boldsymbol{\mathcal{X}}_m)\right), \tag{A.1}$$

we propose to solve Eq. (17) in the Fourier domain as

$$\hat{\mathbf{x}}^{(j+1)} := \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \frac{\rho}{2}\left\|\hat{\bar{\mathbf{D}}}\hat{\mathbf{x}} - \hat{\mathbf{z}}^{(j)}\right\|_2^2 + \frac{\rho}{2}\left\|\hat{\mathbf{x}} - \hat{\mathbf{w}}^{(j)}\right\|_2^2, \tag{A.2}$$

by performing the following transformations:

- $\mathbf{x} \in \mathbb{R}^{MNLTM_d}$ is folded into $\boldsymbol{\mathcal{X}} = \{\boldsymbol{\mathcal{X}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{X}}_m \in \mathbb{R}^{M \times N \times L \times T}\}$, calculate the collection of Fourier transforms $\{\hat{\boldsymbol{\mathcal{X}}}_m\} = \{\mathcal{F}_{4D}(\boldsymbol{\mathcal{X}}_m)\}$ and vectorized as $\hat{\mathbf{x}} = [\mathbf{vec}(\hat{\boldsymbol{\mathcal{X}}}_1)^{\mathbf{H}} \ ... \ \mathbf{vec}(\hat{\boldsymbol{\mathcal{X}}}_{M_d})^{\mathbf{H}}]^{\mathbf{H}} \in$

$\mathbb{C}^{MNLTM_d}$.

- $\mathbf{w}^{(j)} = \mathbf{v}^{(j)} - \mathbf{g}^{(j)} \in \mathbb{R}^{MNLTM_d}$ is folded into $\boldsymbol{\mathcal{W}} = \{\boldsymbol{\mathcal{W}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{W}}_m \in \mathbb{R}^{M\times N\times L\times T}\}$, calculate the collection of Fourier transforms $\{\hat{\boldsymbol{\mathcal{W}}}_m\} = \{\mathcal{F}_{4D}(\boldsymbol{\mathcal{W}}_m)\}$ and vectorized as $\hat{\mathbf{w}} = [\mathbf{vec}(\hat{\boldsymbol{\mathcal{W}}}_1)^{\mathbf{H}}$ ... $\mathbf{vec}(\hat{\boldsymbol{\mathcal{W}}}_{M_d})^{\mathbf{H}}]^{\mathbf{H}} \in \mathbb{C}^{MNLTM_d}$.
- $\mathbf{z}^{(j)} = \mathbf{u}^{(j)} - \mathbf{f}^{(j)} \in \mathbb{R}^{MNLT}$ is folded into $\boldsymbol{\mathcal{Z}}^{(j)} \in \mathbb{R}^{M\times N\times L\times T}$, calculate its Fourier transform $\hat{\boldsymbol{\mathcal{Z}}}^{(j)} = \mathcal{F}_{4D}(\boldsymbol{\mathcal{Z}}^{(j)}) \in \mathbb{C}^{M\times N\times L\times T}$, and vectorized $\hat{\mathbf{z}}^{(j)} = \mathbf{vec}(\hat{\boldsymbol{\mathcal{Z}}}^{(j)}) \in \mathbb{C}^{MNLT}$.
- Calculate the collection of Fourier transforms $\{\hat{\boldsymbol{\mathcal{D}}}_m\} = \{\mathcal{F}_{4D}(\boldsymbol{\mathcal{D}}_m)\}$, create a collection of diagonal matrices $\{\hat{\bar{\mathbf{D}}}_m \in \mathbb{C}^{MNLT\times MNLT}\} = \{\mathbf{diag}(\mathbf{vec}(\hat{\boldsymbol{\mathcal{D}}}_m))\}$ and build the horizontal concatenation $\hat{\bar{\mathbf{D}}} = [\hat{\bar{\mathbf{D}}}_1$ ... $\hat{\bar{\mathbf{D}}}_{M_d}] \in \mathbb{C}^{MNLT\times MNLTM_d}$.

The original $\bar{\mathbf{D}}$ operator must be created as a concatenation of equivalent convolutional matrices with defined structure; while the operator $\hat{\bar{\mathbf{D}}}$ can be created as the concatenation of diagonal matrices, which is much easier to build. Although there is an associated cost of $\mathcal{O}(MNLT\log(MNLT))$ for each 4D Fourier Transform, the memory savings and simplicity of operating simple diagonal matrices make up for this cost.

Finally, Eq. (A.2) has closed solution

$$\hat{\mathbf{x}}^{(j+1)} = \left(\hat{\bar{\mathbf{D}}}^{\mathbf{H}}\hat{\bar{\mathbf{D}}} + \mathbf{I}\right)^{-1}\left(\hat{\bar{\mathbf{D}}}^{\mathbf{H}}\hat{\mathbf{z}}^{(j)} + \hat{\mathbf{w}}^{(j)}\right). \tag{A.3}$$

Eq. (A.3) resembles Eq. (21) in [23], but with higher dimensions. We propose to extend Appendix B in [23], profiting on Woodbury's matrix identity [37]

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}\left(\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U}\right)^{-1}\mathbf{VA}^{-1}, \tag{A.4}$$

by rearranging Eq. (A.3) as

$$\hat{\mathbf{x}}^{(j+1)} = \left(\mathbf{I} + \hat{\bar{\mathbf{D}}}^{\mathbf{H}}\hat{\bar{\mathbf{D}}}\right)^{-1}\hat{\mathbf{b}}, \tag{A.5}$$

with $\hat{\mathbf{b}} = \hat{\bar{\mathbf{D}}}^{\mathbf{H}}\hat{\mathbf{z}}^{(j)} + \hat{\mathbf{w}}^{(j)} \in \mathbb{C}^{MNLTM_d}$, and solution

$$\hat{\mathbf{x}}^{(j+1)} = \left[\hat{\mathbf{b}} - \hat{\bar{\mathbf{D}}}^{\mathbf{H}}\left(\mathbf{I} + \hat{\bar{\mathbf{D}}}\hat{\bar{\mathbf{D}}}^{\mathbf{H}}\right)^{-1}\hat{\bar{\mathbf{D}}}\hat{\mathbf{b}}\right]. \tag{A.6}$$

Eq. (A.6) can be solved using the dimensions rearrangements proposed in Appendix B of [23], minding the increase in the dimensions. Finally, $\hat{\mathbf{x}}^{(j+1)} \in \mathbb{C}^{MNLM_d}$ is folded into $\{\hat{\boldsymbol{\mathcal{X}}}_m^{(j+1)},\ m = 1, ..., M_d \mid \hat{\boldsymbol{\mathcal{X}}}_m^{(j+1)} \in \mathbb{C}^{M\times N\times L\times T}\}$, we calculate the collection of inverse 4D Fourier transforms $\{\boldsymbol{\mathcal{X}}_m^{(j+1)}\} = \{\mathcal{F}_{4D}^{-1}(\hat{\boldsymbol{\mathcal{X}}}_m^{(j+1)})\}$ and the update is the vectorization $\mathbf{x}^{(j+1)} = [\mathbf{vec}(\boldsymbol{\mathcal{X}}_1^{(j+1)})^{\mathbf{T}}$ ... $\mathbf{vec}(\boldsymbol{\mathcal{X}}_{M_d}^{(j+1)})^{\mathbf{T}}]^{\mathbf{T}} \in \mathbb{R}^{MNLTM_d}$.

Appendix B. Temporal Recovery From Compressed Measurements Update

Eq. (18) has closed solution in the spatial domain as

$$\mathbf{u}^{(j+1)} = \left(\mathbf{H}^{\mathbf{T}}\mathbf{H} + \rho\mathbf{I}\right)^{-1}\left(\mathbf{H}^{\mathbf{T}}\mathbf{y} + \rho\mathbf{z}^{(j)}\right), \tag{B.1}$$

and can also be solved using Woodbury's matrix identity as

$$\mathbf{u}^{(j+1)} = \frac{1}{\rho}\left[\mathbf{b} - \mathbf{H}^{\mathbf{T}}\left(\rho\mathbf{I} + \mathbf{H}\mathbf{H}^{\mathbf{T}}\right)^{-1}\mathbf{H}\mathbf{b}\right], \tag{B.2}$$

with $\mathbf{b} = \mathbf{H}^{\mathbf{T}}\mathbf{y} + \rho\mathbf{z}^{(j)}$. Considering that $\mathbf{H}\mathbf{H}^{\mathbf{T}} = \mathbf{I}$, then the update $\mathbf{u}^{(j+1)}$ can be factorized as

$$\mathbf{u}^{(j+1)} = \frac{1}{\rho}\left[\mathbf{b} - \left(\frac{1}{\rho+1}\right)\mathbf{H}^{\mathbf{T}}\mathbf{H}\mathbf{b}\right]. \tag{B.3}$$

Eq. (B.3) can be solved optimally by performing the products right-to-left. This is, start by solving the matrix-to-vector product $\mathbf{H}\mathbf{b}$, and continuing left wise, instead of solving the matrix-to-matrix product $\mathbf{H}^{T}\mathbf{H}$ first.

Appendix C. Sparse Coefficient Maps Update

Eq. (19), has a closed form solution via soft thresholding [38] as

$$\mathbf{v}^{(j+1)} = \mathcal{S}_{\frac{\lambda}{\rho}}\left(\mathbf{x}^{(j+1)} + \mathbf{g}^{(j)}\right). \tag{C.1}$$

Appendix D. Indicator Function for Dictionary Size Restriction

Using the previously defined zero-padding operator $\mathbf{Z}_p$ and its transpose, we define the constraint set

$$\mathcal{C}_{Z_p} = \left\{\mathbf{x} \in \mathbb{R}^{MNLT} : (\mathbf{I} - \mathbf{Z}_p\mathbf{Z}_p^{\mathbf{T}})\mathbf{x} = 0, \|\mathbf{x}\|_2 = 1\right\}, \tag{D.1}$$

which guarantees the desired dictionary elements. Besides, we introduce an indicator function of the constrained set as

$$\iota_{\mathcal{C}_Z}(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \in \mathcal{C}_{Z_p} \\ \infty & \text{if } \mathbf{x} \notin \mathcal{C}_{Z_p} \end{cases}, \tag{D.2}$$

and applied over each vectorized individual convolutional element $\mathbf{d}_m \in \mathbb{R}^{MNL}$. For simplicity, the notation will be applied over the whole collection of dictionary elements.

Appendix E. Convolutional Dictionary Update

Eq. (27) and Eq. (17) have analog structures, thus based on the mathematical development of Appendix A to solve Eq. (27) as

$$\hat{\mathbf{d}}^{(j+1)} = \left(\hat{\mathbf{X}}^{\mathbf{H}}\hat{\mathbf{X}} + \mathbf{I}\right)^{-1}\left(\hat{\mathbf{X}}^{\mathbf{H}}\hat{\mathbf{z}}^{(j)} + \hat{\mathbf{w}}^{(j)}\right), \tag{E.1}$$

where

- $\mathbf{d} \in \mathbb{R}^{MNLTM_d}$ is folded into $\boldsymbol{\mathcal{D}} = \{\boldsymbol{\mathcal{D}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{D}}_m \in \mathbb{R}^{M\times N\times L\times T}\}$, calculate the collection of Fourier transforms $\{\hat{\boldsymbol{\mathcal{D}}}_m\} = \{\mathcal{F}_{4D}(\boldsymbol{\mathcal{D}}_m)\}$ and vectorized as $\hat{\mathbf{d}} = [\mathbf{vec}(\hat{\boldsymbol{\mathcal{D}}}_1)^{\mathbf{H}}\ ...\ \mathbf{vec}(\hat{\boldsymbol{\mathcal{D}}}_{M_d})^{\mathbf{H}}]^{\mathbf{H}} \in \mathbb{C}^{MNLTM_d}$.
- $\mathbf{w}^{(j)} = \mathbf{q}^{(j)} - \mathbf{t}^{(j)} \in \mathbb{R}^{MNLTM_d}$ is folded into $\boldsymbol{\mathcal{W}} = \{\boldsymbol{\mathcal{W}}_m,\ m = 1, ..., M_d \mid \boldsymbol{\mathcal{W}}_m \in \mathbb{R}^{M\times N\times L\times T}\}$, calculate the collection of Fourier transforms $\{\hat{\boldsymbol{\mathcal{W}}}_m\} = \{\mathcal{F}_{4D}(\boldsymbol{\mathcal{W}}_m)\}$ and vectorized as $\hat{\mathbf{w}} = [\mathbf{vec}(\hat{\boldsymbol{\mathcal{W}}}_1)^{\mathbf{H}}\ ...\ \mathbf{vec}(\hat{\boldsymbol{\mathcal{W}}}_{M_d})^{\mathbf{H}}]^{\mathbf{H}} \in \mathbb{C}^{MNLTM_d}$.
- $\mathbf{z}^{(j)} = \mathbf{p}^{(j)} - \mathbf{r}^{(j)} \in \mathbb{R}^{MNLT}$ is folded into $\boldsymbol{\mathcal{Z}}^{(j)} \in \mathbb{R}^{M\times N\times L\times T}$, calculate its Fourier transform $\hat{\boldsymbol{\mathcal{Z}}}^{(j)} = \mathcal{F}_{4D}(\boldsymbol{\mathcal{Z}}^{(j)}) \in \mathbb{C}^{M\times N\times L\times T}$, and vectorized $\hat{\mathbf{z}}^{(j)} = \mathbf{vec}(\hat{\boldsymbol{\mathcal{Z}}}^{(j)}) \in \mathbb{C}^{MNLT}$.
- Calculate the collection of Fourier transforms $\{\hat{\boldsymbol{\mathcal{X}}}_m\} = \{\mathcal{F}_{4D}(\boldsymbol{\mathcal{X}}_m)\}$, create a collection of diagonal matrices $\{\hat{\mathbf{D}}_m \in \mathbb{C}^{MNLT\times MNLT}\} = \{\mathbf{diag}(\mathbf{vec}(\hat{\boldsymbol{\mathcal{X}}}_m))\}$ and build the horizontal concatenation $\hat{\mathbf{X}} = [\hat{\mathbf{X}}_1\ ...\ \hat{\mathbf{X}}_{M_d}] \in \mathbb{C}^{MNLT\times MNLTM_d}$.

The same numerical rearrangements from Appendix B in [23] also apply, minding the increase in the dimensions. Eq. (E.1) has closed solution

$$\hat{\mathbf{d}}^{(j+1)} = \left[\hat{\mathbf{b}} - \hat{\mathbf{X}}^{\mathbf{H}}\left(\mathbf{I} + \hat{\mathbf{X}}\hat{\mathbf{X}}^{\mathbf{H}}\right)^{-1}\hat{\mathbf{X}}\hat{\mathbf{b}}\right]. \tag{E.2}$$

with $\hat{\mathbf{b}} = \hat{\mathbf{X}}^{\mathbf{H}}\hat{\mathbf{z}}^{(j)} + \hat{\mathbf{w}}^{(j)} \in \mathbb{C}^{MNLTM_d}$. Finally, $\mathbf{d}^{(j+1)}$ is built from $\hat{\mathbf{d}}^{(j+1)}$ just as $\mathbf{x}^{(j+1)}$ is built from $\hat{\mathbf{x}}^{(j+1)}$.

Appendix F. Temporal Recovery From Compressed Measurements Update

Eq. (28) is analog to Eq. (18), thus based on the mathematical development of Appendix B to solve Eq. (28) as

$$\mathbf{p}^{(j+1)} = \frac{1}{\sigma}\left[\mathbf{b} - \left(\frac{1}{\sigma+1}\right)\mathbf{H}^{\mathbf{T}}\mathbf{H}\mathbf{b}\right], \tag{F.1}$$

where $\mathbf{b} = \mathbf{H}^{\mathbf{T}}\mathbf{y} + \sigma\mathbf{z}^{(j)}$ and $\mathbf{z}^{(j)} = \bar{\mathbf{X}}\mathbf{d}^{(j+1)} + \mathbf{r}^{(j)}$.

Appendix G. Desired Convolutional Dictionary Update

Eq. (29) can be solved via proximal for each m-4D dictionary element as

$$\mathbf{q}_m^{(j+1)} = \frac{\mathbf{Z}_p \mathbf{Z}_p^{\mathbf{T}} (\mathbf{d}_m^{(j+1)} + \mathbf{t}_m^{(j)})}{\left\| \mathbf{Z}_p \mathbf{Z}_p^{\mathbf{T}} (\mathbf{d}_m^{(j+1)} + \mathbf{t}_m^{(j)}) \right\|_2}, \tag{G.1}$$