signals

MDPI

Article

Optimizing Dynamic Mode Decomposition for Video Denoising via Plug-and-Play Alternating Direction Method of Multipliers †

Hyoga Yamamoto, Shunki Anami and Ryo Matsuoka *

Faculty of Environmental Engineering, The University of Kitakyushu, Fukuoka 808-0135, Japan
* Correspondence: r-matsuoka@kitakyu-u.ac.jp
† This paper is an extended version of our paper published in Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Tokyo, Japan, 14–17 December 2021.

Abstract: Dynamic mode decomposition (DMD) is a powerful tool for separating the background and foreground in videos. This algorithm decomposes a video into dynamic modes, called DMD modes, to facilitate the extraction of the near-zero mode, which represents the stationary background. Simultaneously, it captures the evolving motion in the remaining modes, which correspond to the moving foreground components. However, when applied to noisy video, this separation leads to degradation of the background and foreground components, primarily due to the noise-induced degradation of the DMD mode. This paper introduces a novel noise removal method for the DMD mode in noisy videos. Specifically, we formulate a minimization problem that reduces the noise in the DMD mode and the reconstructed video. The proposed problem is solved using an algorithm based on the plug-and-play alternating direction method of multipliers (PnP-ADMM). We applied the proposed method to several video datasets with different levels of artificially added Gaussian noise in the experiment. Our method consistently yielded superior results in quantitative evaluations using peak-signal-to-noise ratio (PSNR) and structural similarity (SSIM) compared to naive noise removal methods. In addition, qualitative comparisons confirmed that our method can restore higher-quality videos than the naive methods.

Keywords: dynamic mode decomposition; noise removal; plug-and-play; alternating direction method of multipliers; total variation; BM3D

Citation: Yamamoto, H.; Anami, S.; Matsuoka, R. Optimizing Dynamic Mode Decomposition for Video Denoising via Plug-and-Play Alternating Direction Method of Multipliers. *Signals* **2024**, *5*, 202–215. https://doi.org/10.3390/signals5020011

Academic Editor: Cataldo Guaragnella

Received: 5 February 2024
Revised: 13 March 2024
Accepted: 19 March 2024
Published: 1 April 2024

1. Introduction

1.1. Background

Video processing is critical in surveillance and in-vehicle systems and specifically includes essential tasks such as noise removal, foreground/background separation, and object detection. The foreground/background separation process helps detect, identify, track, and recognize objects within a video sequence.

Dynamic mode decomposition (DMD) is useful for separating background and foreground components in various applications [1–9]. Initially applied in fluid dynamics, DMD has evolved into a powerful tool for analyzing the dynamics of nonlinear systems, as shown in research such as [10–12]. In background/foreground separation, the DMD method identifies a static background by performing a spatiotemporal decomposition of video frames. It effectively distinguishes between static modes and the remaining dynamic modes, separating a static background from a dynamic foreground.

In low-light conditions and using high-sensitivity settings, the captured video exhibits noticeable noise levels due to amplified sensor noise. This amplification occurs because camera sensors operating at high sensitivity are more susceptible to capturing and amplifying random electrical signals. However, sensor noise often deteriorates the DMD mode when attempting to separate foreground and background components from

noisy videos using the DMD algorithm. This issue has also been mentioned in the field of fluid analysis, where sensor noise can introduce bias errors and reduce the accuracy of the analysis of fluids.

To address this limitation, some researchers have proposed the total-least-squares DMD (tlsDMD) algorithm to mitigate bias errors due to sensor noise [13,14]. However, tlsDMD-based methods are ineffective at removing spatial noise to separate a noisy video into foreground and background because they lack prior knowledge that promotes image smoothness.

1.2. Related Work

In [10], Schmid introduced the basic DMD algorithm and explained its relevance to standard methods used in fluid analysis for atmospheric or oceanographic data. The potential of the DMD algorithm was demonstrated through several scenarios, including a plane channel flow, flow over a two-dimensional cavity, wake flow behind a flexible membrane, and a jet passing between two cylinders. The demonstrations showed the ability of this algorithm to analyze fluid flows and identify critical physical mechanisms that govern them, highlighting its power and versatility.

In [13,14], the vulnerability of the DMD algorithm to sensor noise was mentioned. The basic DMD algorithm does not take sensor noise into account. When decomposing snapshots degraded by sensor noise, the estimated eigenvalues deviate from the ideal values due to noise bias. Hemati et al. proposed the tlsDMD algorithm, which estimates the bias due to sensor noise in forward and backward DMD mode estimation of snapshots [14]. This algorithm calculates DMD modes and their eigenvalues while excluding the estimated bias. As a result, the bias caused by sensor noise can be removed, and simulation experiments showed that the estimated eigenvalues are close to those calculated for snapshots without sensor noise. Dawson et al. analytically derived a formula that explicitly shows how DMD is affected by noise, assuming that sensor noise is uncorrelated with system dynamics [13]. They complemented the derivation of the tlsDMD algorithm. However, this algorithm aims to remove the sensor noise bias, and the noise removal accuracy of the DMD mode is insufficient. In addition, they do not consider the reconstruction error of snapshots. Therefore, when applied to videos for foreground/background separation applications, it is impossible to remove sensor noise from reconstructed video frames sufficiently and their DMD modes.

Various noise removal methods have been proposed for images and videos, including optimization-based approaches such as total variation (TV) regularization [15–21] and filter-based noise removal methods such as block matching 3D (BM3D) [22–25]. The TV is designed to represent the total magnitude of the vertical and horizontal discrete gradients of an image and promotes the local smoothness property in optimization [15–19]. In the case of noise removal, the TV effectively reduces noise by emphasizing spatial smoothness while preserving edges and structures. Although BM3D was proposed over a decade ago, it remains one of the most advanced methods for denoising images and videos [22,23]. It works by partitioning the image into blocks, searching for similar blocks, and then thresholding their noise in the 3D-transformed domain using the discrete cosine transform (DCT). BM3D uses local and non-local similarities in the image to effectively reduce noise while preserving image structure and texture.

However, these noise removal methods do not explicitly account for the spatial smoothness and texture of the DMD mode. Improving the spatial smoothness and texture of both the video frames and their DMD modes is critical to obtaining reliable results for practical video analysis in the presence of noise. Additionally, for foreground/background separation applications, noise removal on the DMD mode obtained by its decomposition must be considered.

1.3. Contribution

This paper introduces a novel noise removal method for the DMD mode obtained by applying DMD to noisy videos. We formulate a minimization problem within the plug-and-play framework that aims to simultaneously reduce the noise in DMD modes and its reconstructed videos. To solve the proposed problem, we introduce an algorithm based on the plug-and-play alternating direction method of multipliers (PnP-ADMM). The experimental results demonstrate the effectiveness of the proposed method by comparing it with naive noise removal methods. The main contributions of this paper are as follows:

1. Introducing a novel minimization problem that simultaneously removes noise from DMD modes and improves their reconstructed video quality. This problem includes two implicit regularization terms for the DMD modes and their reconstructed video, along with two constraints on the reconstructed video: one for reconstruction error and the other to ensure real numbers.
2. The development of the PnP-ADMM algorithm is based on the plug-and-play framework and Gaussian denoisers. This algorithm solves the proposed minimization problem and aims to obtain optimal DMD modes capable of reconstructing a smooth and noiseless video.
3. Two advanced noise removal methods, the total variation (TV) algorithm and BM3D, are employed as Gaussian denoisers to implicitly regularize the DMD modes and their reconstructed video within the optimization algorithm.

In the previous study [26], we used the TV denoiser with the PnP-ADMM algorithm to remove noise from the DMD modes obtained by decomposing the observed noisy videos. Since the DMD modes are complex numbers, the reconstructed video may have values in the imaginary part. Although the reconstructed video must contain real numbers, such constraints were not explicitly considered when formulating the optimization problem. In the proposed method, we replaced the TV denoiser with the BM3D denoiser to improve the noise removal performance. Additionally, we added a constraint to restrict the reconstructed video obtained by optimal DMD mode to real numbers in the optimization problem.

The remainder of this paper is organized as follows. In Section 2, we present mathematical preliminaries, a DMD algorithm, a PnP-ADMM algorithm, some proximal tools, and total variation regularization. Section 3 introduces the proposed minimization problem for noise removal of the DMD mode. In Section 4, several examples are presented and compared with some naive noise removal methods to verify the effectiveness of the proposed method. Finally, Section 5 concludes the paper.

2. Preliminaries

Throughout this paper, bold-faced lowercase and uppercase letters indicate vectors and matrices, respectively. The notations $\mathbb{R}^N$ and $\mathbb{C}^N$ denote real- and complex-valued vector spaces of N dimensions, respectively. We define the notations $\mathbb{R}^{N\times M}$ and $\mathbb{C}^{N\times M}$ as the set of $N \times M$ real-valued and complex-valued matrices, respectively. The symbols $(\cdot)^\top$ and $(\cdot)^*$ denote the operations of non-conjugate and conjugate transpose of vectors and matrices, respectively. The symbol diag(X) denotes the operation of extracting the diagonal components of a diagonal matrix X and converting it into a column vector.

2.1. Dynamic Mode Decomposition

The DMD algorithm is defined for pairs of N-dimentional data $\{\mathbf{x}_i, \mathbf{y}_i\}$ satisfying $\mathbf{y}_i = \mathbf{A}\mathbf{x}_i \quad (i = 1, \ldots, M)$, for some matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$. These vectors are sampled by equispaced snapshots of a dynamical system. However, the matrix $\mathbf{A}$ is not completely determined by the snapshots. The DMD algorithm estimates $\mathbf{A}$ such that satisfying $\mathbf{Y} \approx \mathbf{A}\mathbf{X}$, where $\mathbf{Y} := [\mathbf{y}_1, \ldots, \mathbf{y}_M]$ and $\mathbf{X} := [\mathbf{x}_1, \ldots, \mathbf{x}_M]$. Several methods have been proposed to compute DMD [10,13,14,27,28].

In this paper, we use the basic DMD algorithm [10] described as follows:

(i) Calculate the (reduced) singular value decomposition (SVD) of the matrix $\mathbf{X}$ as $\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^*$, where $\mathbf{U} \in \mathbb{C}^{N\times r}$, $\mathbf{S} \in \mathbb{C}^{r\times r}$, and $\mathbf{V} \in \mathbb{C}^{M\times r}$, with the rank r.
(ii) Let $\tilde{\mathbf{A}}$ be defined by $\tilde{\mathbf{A}} = \mathbf{U}^*\mathbf{Y}\mathbf{V}\mathbf{S}^{-1}$.
(iii) Compute the eigenvalue decomposition of $\tilde{\mathbf{A}}$ as $\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\mathbf{\Lambda}$, where $\mathbf{W} := [\mathbf{w}_1, \ldots, \mathbf{w}_r]$ is a matrix configured by arranging the eigenvectors $\mathbf{w}_i \in \mathbb{C}^r$ $(i = 1, \ldots, r)$ and $\mathbf{\Lambda}$ is a diagonal matrix having eigenvalues λ_i $(i = 1, \ldots, r)$ as the diagonal elements.
(iv) The DMD mode $\mathbf{\Phi} := [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_r]$ $(\boldsymbol{\phi}_i \in \mathbb{C}^N)$ is obtained by $\mathbf{\Phi} = \mathbf{U}\mathbf{W}$.
(v) Then, we define $\mathbf{\Sigma} \in \mathbb{C}^{r\times M}$ as

$$\mathbf{\Sigma} := [\mathrm{diag}(\mathbf{\Lambda}^0)\ \mathrm{diag}(\mathbf{\Lambda}^1)\ \cdots\ \mathrm{diag}(\mathbf{\Lambda}^{M-1})]. \tag{1}$$

(vi) Estimate the diagonal matrix $\mathbf{B} \in \mathbb{C}^{r\times r}$ by minimizing the cost function

$$E(\mathbf{B}) := \|\mathbf{X} - \mathbf{\Phi}\mathbf{B}\mathbf{\Sigma}\|_F^2. \tag{2}$$

(vii) Finally, $\mathbf{X}$ is represented by $\mathbf{\Phi}\mathbf{B}\mathbf{\Sigma}$ as

$$\mathbf{X} \approx \mathbf{\Phi}\mathbf{B}\mathbf{\Sigma}. \tag{3}$$

In this manner, the DMD algorithm decomposes $\mathbf{X}$ into $\mathbf{\Phi}$, $\mathbf{B}$, and $\mathbf{\Sigma}$, where $\mathbf{\Phi}$ is the set of dynamic modes of observed dynamical systems, each diagonal element of $\mathbf{B}$ is the amplitude of each mode, and each row of $\mathbf{\Sigma}$ is a Vandermonde matrix describing the temporal evolution of each mode.

2.2. Plug-and-Play Alternating Direction Method of Multipliers

The alternating direction method of multipliers (ADMM) [29] is a proximal splitting algorithm for convex optimization problems of the form

$$\min_{\mathbf{x}\in\mathbb{R}^{N_1},\, \mathbf{z}\in\mathbb{R}^{N_2}} F(\mathbf{x}) + G(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{z} = \mathbf{L}\mathbf{x}, \tag{4}$$

where F and G are usually assumed to be a quadratic and proximable function, respectively, and $\mathbf{L} \in \mathbb{R}^{N_2\times N_1}$ is a matrix with full-column rank. For any $\mathbf{x}^{(0)} \in \mathbb{R}^{N_1}$, $\mathbf{z}^{(0)} \in \mathbb{R}^{N_2}$, $\mathbf{b}^{(0)} \in \mathbb{R}^{N_2}$ and $\rho > 0$, the ADMM algorithm is given by

$$\begin{cases} \mathbf{x}^{(t+1)} = \arg\min_{\mathbf{x}} \left\{ F(\mathbf{x}) + \dfrac{\rho}{2}\|\mathbf{z}^{(t)} - \mathbf{L}\mathbf{x} - \mathbf{b}^{(t)}\|_2^2 \right\}, \\ \mathbf{z}^{(t+1)} = \arg\min_{\mathbf{z}} \left\{ G(\mathbf{z}) + \dfrac{\rho}{2}\|\mathbf{z} - \mathbf{L}\mathbf{x}^{(t+1)} - \mathbf{b}^{(t)}\|_2^2 \right\}, \\ \mathbf{b}^{(t+1)} = \mathbf{b}^{(t)} + \mathbf{L}\mathbf{x}^{(t+1)} - \mathbf{z}^{(t+1)}, \end{cases} \tag{5}$$

where the superscript (t) denotes the iteration number. The sequence generated by Equation (5) converges quickly to an optimal solution of Equation (4).

In PnP-ADMM [30,31], the solution of the sub-problem with respect to $\mathbf{z}$ (assuming $\mathbf{L}$ is the identity matrix) is replaced by an off-the-shelf noise removal algorithm, to yield

$$\mathbf{z}^{(t+1)} = \mathcal{D}_\sigma\left(\mathbf{x}^{(t+1)} + \mathbf{b}^{(t)}\right), \tag{6}$$

where $\mathcal{D}_\sigma$ denotes the Gaussian denoiser and σ is the standard deviation of the assumed additive white Gaussian noise (AWGN).

2.3. Proximal Tools

The proximity operator [32] is a key tool of proximal splitting techniques. Let $\mathbf{x} \in \mathbb{R}^N$ be an input vector. For any $\gamma > 0$, the proximity operator of f over $\mathbb{R}^N$ is defined by

$$\mathrm{prox}_{\gamma f}(\mathbf{x}) := \arg\min_{\mathbf{y}\in\mathbb{R}^N} f(\mathbf{y}) + \frac{1}{2\gamma}\|\mathbf{x} - \mathbf{y}\|^2. \tag{7}$$

For a given nonempty closed convex set $\mathcal{C}$, the indicator function of $\mathcal{C}$ is defined by

$$\iota_{\mathcal{C}}(\mathbf{x}) := \begin{cases} 0, & \text{if } \mathbf{x} \in \mathcal{C}, \\ +\infty, & \text{otherwise.} \end{cases} \tag{8}$$

The proximity operator of $\iota_{\mathcal{C}}$ is expressed as

$$\mathrm{prox}_{\gamma\iota_{\mathcal{C}}}(\mathbf{x}) := \arg\min_{\mathbf{y}\in\mathbb{R}^N} \iota_{\mathcal{C}}(\mathbf{x}) + \frac{1}{2\gamma}\|\mathbf{x}-\mathbf{y}\|_2^2. \tag{9}$$

The solution of $\mathrm{prox}_{\gamma\iota_{\mathcal{C}}}$ should be in the set $\mathcal{C}$ and minimize $\|\mathbf{x}-\mathbf{y}\|_2^2$. Thus, for any index $\gamma > 0$, this proximity operator is equivalent to the metric projection onto $\mathcal{C}$, i.e., $\mathcal{P}_{\mathcal{C}}(\mathbf{x}) = \mathrm{prox}_{\gamma\iota_{\mathcal{C}}}(\mathbf{x})$.

Let $\mathbf{l}$ and $\mathbf{u} \in \mathbb{R}^N$ be the lower and upper bounds, respectively. The box constraint forces each element of $\mathbf{x}$ into the dynamic range $[l_i, u_i]$ for $i = 1, \ldots, N$, and its closed convex set is defined as

$$\mathcal{C}_{[\mathbf{l},\mathbf{u}]} := \Big\{ \mathbf{x} \in \mathbb{R}^N \;\Big|\; l_i \le x_i \le u_i \;\; (i = 1, \ldots, N) \Big\}. \tag{10}$$

The computation of the metric projection onto $\mathcal{C}_{[\mathbf{l},\mathbf{u}]}$ for $i = 1, \ldots, N$ is given by

$$\left[\mathcal{P}_{\mathcal{C}_{[\mathbf{l},\mathbf{u}]}}(\mathbf{x})\right]_i = \begin{cases} l_i, & \text{if } x_i < l_i, \\ u_i, & \text{if } x_i > u_i, \\ x_i, & \text{if } l_i \le x_i \le u_i. \end{cases} \tag{11}$$

The ℓ_2 ball constraint forces the Euclidean distance between a vector $\mathbf{x}$ and a centered vector $\mathbf{v}$ to be less than a radius ϵ, and its closed convex set is defined as

$$\mathcal{B}^2_{\mathbf{v},\epsilon} := \Big\{ \mathbf{x} \in \mathbb{R}^N \;\Big|\; \|\mathbf{x}-\mathbf{v}\|_2 \le \epsilon \Big\}. \tag{12}$$

The computation of the metric projection onto $\mathcal{B}^2_{\mathbf{v},\epsilon}$ is given by

$$\mathcal{P}_{\mathcal{B}^2_{\mathbf{v},\epsilon}}(\mathbf{x}) = \begin{cases} \mathbf{x}, & \text{if } \|\mathbf{x}-\mathbf{v}\|_2 \le \epsilon, \\ \mathbf{v} + \epsilon\frac{\mathbf{x}-\mathbf{v}}{\|\mathbf{x}-\mathbf{v}\|_2}, & \text{otherwise .} \end{cases} \tag{13}$$

2.4. Total Variation

The total variation (TV) is defined as the total magnitude of the vertical and horizontal discrete gradients of an image [16]. When we utilize the TV as a regularization on minimization problems for images, it promotes the local smoothness of images to be estimated.

Let $\mathbf{x} \in \mathbb{R}^N$ be a vectorized grayscale image, where N is the total number of pixels. Also, let $\mathbf{D}_v$ and $\mathbf{D}_h \in \mathbb{R}^{N\times N}$ be the vertical and horizontal first-order differential operators with a Neumann boundary, respectively. Then, the differential operator with respect to $\mathbf{x}$ is defined by $\mathbf{D} := [\mathbf{D}_v^\top \; \mathbf{D}_h^\top]^\top \left(\in \mathbb{R}^{2N\times N}\right)$, and thus the TV is defined as [16,33,34]

$$\|\mathbf{x}\|_{\mathrm{TV}} := \|\mathbf{D}\mathbf{x}\|_{1,2} = \sum_{i=1}^{N} \sqrt{(\mathbf{D}_v\mathbf{x})_i^2 + (\mathbf{D}_h\mathbf{x})_i^2}, \tag{14}$$

where $(\mathbf{D}_v\mathbf{x})_i$ and $(\mathbf{D}_h\mathbf{x})_i$ are the i-th element of $\mathbf{D}_v\mathbf{x}$ and $\mathbf{D}_h\mathbf{x}$, respectively.

The minimization problem with TV regularization, which is often used in PnP-ADMM as a denoiser, is defined as

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}\in\mathbb{R}^N} \lambda\|\mathbf{x}\|_{\mathrm{TV}} + \frac{1}{2}\|\mathbf{x}-\mathbf{x}_{\mathrm{in}}\|_2^2, \tag{15}$$

where $\mathbf{x}_{\text{in}}$ is a vectorized input image and $\lambda > 0$ is a balancing weight of two terms. We can find the optimal solution of Equation (15) by using the ADMM algorithm.

By introducing auxiliary variables $\mathbf{z} \in \mathbb{R}^{2N}$, we rewrite Equation (15) into the following equivalent expression:

$$\min_{\mathbf{x},\mathbf{z}} \lambda \|\mathbf{z}\|_{\text{TV}} + \frac{1}{2}\|\mathbf{x} - \mathbf{x}_{\text{in}}\|_2^2 \quad \text{s.t. } \mathbf{z} = \mathbf{x}. \tag{16}$$

The algorithm for solving Equation (16) with $\rho > 0$ is summarized in Algorithm 1. The update of $\mathbf{x}$ can be achieved by solving a simple quadratic minimization problem. The solution of the sub-problem with respect to $\mathbf{z}$ can be obtained for each sub-vector $\mathbf{z}^{\mathcal{G}_1}, \ldots, \mathbf{z}^{\mathcal{G}_N}$, by

$$\mathbf{z}^{\mathcal{G}_i} = \text{prox}_{\lambda/\rho\|\cdot\|_{\text{TV}}}\left(\mathbf{y}^{\mathcal{G}_i}\right) = \max\left\{1 - \frac{\lambda}{\rho}\left(y_i^2 + y_{i+N}^2\right)^{-\frac{1}{2}},\ 0\right\}\mathbf{y}^{\mathcal{G}_i}, \tag{17}$$

where $\mathbf{z}^{\mathcal{G}_i} =: \{z_i, z_{i+N}\}$ and $\mathbf{y}^{\mathcal{G}_i} =: \{y_i, y_{i+N}\}$ are the i-th sub-vector of $\mathbf{z}$ and $\mathbf{y}$, respectively.

Algorithm 1 Solved algorithm for Equation (16)

1: **Input** : $\mathbf{x}_{\text{in}}$, $\mathbf{z}^{(0)}$, $\mathbf{b}^{(0)}$, ρ, λ
2: **Output** : $\mathbf{x}^{(t)}$
3: **while** A stopping criterion is not satisfied **do**
4: $\quad \mathbf{x}^{(t+1)} \leftarrow \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \mathbf{x}_{\text{in}}\|_2^2 + \frac{\rho}{2}\|\mathbf{z}^{(t)} - \mathbf{D}\mathbf{x} - \mathbf{b}^{(t)}\|_2^2$;
5: $\quad \mathbf{z}^{(t+1)} \leftarrow \text{prox}_{\lambda/\rho\|\cdot\|_{\text{TV}}}\left(\mathbf{D}\mathbf{x}^{(t+1)} + \mathbf{b}^{(t)}\right)$;
6: $\quad \mathbf{b}^{(t+1)} \leftarrow \mathbf{b}^{(t)} + \mathbf{D}\mathbf{x}^{(t+1)} - \mathbf{z}^{(t+1)}$;
7: $\quad t \leftarrow t + 1$;
8: **end while**

3. Proposed Methods

3.1. Data Model

We consider the following observation model

$$\mathbf{y}_m = \mathbf{x}_m + \mathbf{n}_m, \tag{18}$$

where $\mathbf{x}_m \in \mathbb{R}^N (m = 1, \ldots, M+1)$ denotes a vectorized latent video frame, N is the number of pixels, $M+1$ is the number of frames, $\mathbf{n}_m \in \mathbb{R}^N$ is an AWGN vector, and $\mathbf{y}_m \in \mathbb{R}^N (m = 1, \ldots, M+1)$ is a vectorized observed video frame. Furthermore, we defined the matrix form of $m = 1, \ldots, M$ frames of the observed and decomposed video by using the above $\mathbf{y}_m$ and the DMD algorithm described in Section 2.1 as

$$\mathbf{Y} := [\mathbf{y}_1\ \mathbf{y}_2\ \ldots\ \mathbf{y}_M] \approx \widehat{\mathbf{\Phi}}\mathbf{B}\mathbf{\Sigma}, \tag{19}$$

where $\widehat{\mathbf{\Phi}} \in \mathbb{C}^{N \times r}$ is the matrix consisting of noisy DMD modes arranged to the row direction. We assumed that DMD modes are degraded, while its amplitudes $\mathbf{B}$ and the temporal evolution $\mathbf{\Sigma}$ are scarcely affected by noise.

3.2. Minimization Problem

Our aim is to find a noiseless DMD mode matrix $\mathbf{\Phi}^\star$ from a noisy observed video $\mathbf{Y} \approx \widehat{\mathbf{\Phi}}\mathbf{B}\mathbf{\Sigma}$. To estimate $\mathbf{\Phi}^\star$, we formulate the following minimization problem:

$$\min_{\mathbf{\Phi}} \alpha\mathcal{R}_r(\mathbf{\Phi}\mathbf{B}\mathbf{\Sigma}) + (1-\alpha)\mathcal{R}_m(\mathbf{\Phi}) \quad \text{s.t. } \|\mathbf{Y} - \mathbf{\Phi}\mathbf{B}\mathbf{\Sigma}\|_F \le \epsilon,\ \mathbf{\Phi}\mathbf{B}\mathbf{\Sigma} \in \mathbb{R}^{N \times M}, \tag{20}$$

where $\mathcal{R}_r$ and $\mathcal{R}_m$ are regularization terms for a reconstructed video $\mathbf{\Phi}\mathbf{B}\mathbf{\Sigma}$ and a DMD mode matrix $\mathbf{\Phi}$, respectively, and $\alpha \in [0, 1]$ is the balancing weight of these terms. The observed video matrix consists of real numbers, but the reconstructed video may contain complex numbers due to the nature of the matrices obtained by the DMD algorithm, which

are complex. Therefore, we introduce a real-valued constraint on the reconstructed video in the minimization problem.

To find a solution of Equation (20), we employ the PnP-ADMM algorithm described in Section 2.2.

3.3. Optimization

The minimization problem Equation (20) is not directly applicable to PnP-ADMM. We reformulate it in a form that can be applied to PnP-ADMM. First, we define the convex set $\mathcal{B}^F_{\mathbf{Y},\epsilon}$ as

$$\mathcal{B}^F_{\mathbf{Y},\epsilon} := \left\{ \mathbf{X} \in \mathbb{R}^{N\times M} \mid \|\mathbf{X}-\mathbf{Y}\|_F \leq \epsilon \right\}. \tag{21}$$

Then, we reformulate Equation (20) into the following unconstrained problem:

$$\min_{\mathbf{\Phi}} \ \alpha\mathcal{R}_r(\mathbf{\Phi B\Sigma}) + (1-\alpha)\mathcal{R}_m(\mathbf{\Phi}) + \iota_{\mathcal{B}^F_{\mathbf{Y},\epsilon}}(\mathbf{\Phi B\Sigma}) + \iota_{\mathbb{R}^{N\times M}}(\mathbf{\Phi B\Sigma}), \tag{22}$$

where $\iota_{\mathcal{B}^F_{\mathbf{Y},\epsilon}}(\cdot)$ is the indicator function of $\mathcal{B}^F_{\mathbf{Y},\epsilon}$. This function guarantees that the Frobenius norm of $\mathbf{Y}-\mathbf{\Phi B\Sigma}$ is less than or equal to ϵ. Similarly, $\iota_{\mathbb{R}^{N\times M}}(\cdot)$ is the indicator function of $\mathbb{R}^{N\times M}$. This function guarantees that the reconstructed video is composed of real numbers. Thus, the role of the third and fourth terms of Equation (22) correspond to the constraints of the minimization problem Equation (20). Furthermore, by introducing auxiliary variables $\mathbf{Z}_1 \in \mathbb{C}^{N\times M}, \mathbf{Z}_2 \in \mathbb{C}^{N\times r}, \mathbf{Z}_3 \in \mathbb{C}^{N\times M}$, and $\mathbf{Z}_4 \in \mathbb{C}^{N\times M}$, we rewrite the minimization problem Equation (22) into the following equivalent expression:

$$\min_{\mathbf{\Phi},\mathbf{Z}_i(i=1,2,3,4)} \alpha\mathcal{R}_r(\mathbf{Z}_1) + (1-\alpha)\mathcal{R}_m(\mathbf{Z}_2) + \iota_{\mathcal{B}^F_{\mathbf{Y},\epsilon}}(\mathbf{Z}_3) + \iota_{\mathbb{R}^{N\times M}}(\mathbf{Z}_4), \quad \text{s.t.} \quad \mathbf{Z}_1 = \mathbf{\Phi B\Sigma}, \mathbf{Z}_2 = \mathbf{\Phi}, \mathbf{Z}_3 = \mathbf{\Phi B\Sigma}, \mathbf{Z}_4 = \mathbf{\Phi B\Sigma}. \tag{23}$$

The minimization problem Equation (23) can be applied to PnP-ADMM. The process of PnP-ADMM for solving Equation (23) with ρ_i $(i = 1,2,3,4)$ is summarized in Algorithm 2.

Algorithm 2 Proposed algorithm for Equation (23)

1: **Input**: $\mathbf{Y}$, $\mathbf{Z}_i^{(0)}$, $\mathbf{\Theta}_i^{(0)}$, ρ_i $(i = 1,2,3,4)$, α, ϵ
2: **Output** : $\mathbf{\Phi}^{(t)}$
3: **while** A stopping criterion is not satisfied **do**
4: $\mathbf{\Phi}^{(t+1)} \leftarrow \arg\min_{\mathbf{\Phi}} \frac{\rho_1}{2}\|\mathbf{Z}_1^{(t)} - \mathbf{\Phi B\Sigma} - \mathbf{\Theta}_1^{(t)}\|_F^2 + \frac{\rho_2}{2}\|\mathbf{Z}_2^{(t)} - \mathbf{\Phi} - \mathbf{\Theta}_2^{(t)}\|_F^2 + \frac{\rho_3}{2}\|\mathbf{Z}_3^{(t)} - \mathbf{\Phi B\Sigma} - \mathbf{\Theta}_3^{(t)}\|_F^2 + \frac{\rho_4}{2}\|\mathbf{Z}_4^{(t)} - \mathbf{\Phi B\Sigma} - \mathbf{\Theta}_4^{(t)}\|_F^2$;
5: $\mathbf{Z}_1^{(t+1)} \leftarrow \mathcal{D}_{\mathcal{R}_{r,\alpha/\rho_1}}\left(\mathbf{\Phi}^{(t+1)}\mathbf{B\Sigma} + \mathbf{\Theta}_1^{(t)}\right)$;
6: $\mathbf{Z}_2^{(t+1)} \leftarrow \mathcal{D}_{\mathcal{R}_{m,(1-\alpha)/\rho_2}}\left(\mathbf{\Phi}^{(t+1)} + \mathbf{\Theta}_2^{(t)}\right)$;
7: $\mathbf{Z}_3^{(t+1)} \leftarrow \mathcal{P}_{\mathcal{B}^F_{\mathbf{Y},\epsilon}}\left(\mathbf{\Phi}^{(t+1)}\mathbf{B\Sigma} + \mathbf{\Theta}_3^{(t)}\right)$;
8: $\mathbf{Z}_4^{(t+1)} \leftarrow \mathcal{P}_{\mathbb{R}^{N\times M}}\left(\mathbf{\Phi}^{(t+1)}\mathbf{B\Sigma} + \mathbf{\Theta}_4^{(t)}\right)$;
9: $\mathbf{\Theta}_1^{(t+1)} \leftarrow \mathbf{\Theta}_1^{(t)} + \mathbf{\Phi}^{(t+1)}\mathbf{B\Sigma} - \mathbf{Z}_1^{(t+1)}$;
10: $\mathbf{\Theta}_2^{(t+1)} \leftarrow \mathbf{\Theta}_2^{(t)} + \mathbf{\Phi}^{(t+1)} - \mathbf{Z}_2^{(t+1)}$;
11: $\mathbf{\Theta}_3^{(t+1)} \leftarrow \mathbf{\Theta}_3^{(t)} + \mathbf{\Phi}^{(t+1)}\mathbf{B\Sigma} - \mathbf{Z}_3^{(t+1)}$;
12: $\mathbf{\Theta}_4^{(t+1)} \leftarrow \mathbf{\Theta}_4^{(t)} + \mathbf{\Phi}^{(t+1)}\mathbf{B\Sigma} - \mathbf{Z}_4^{(t+1)}$;
13: $t \leftarrow t+1$;
14: **end while**

The update of $\mathbf{\Phi}$ in step 4 of Algorithm 2 is achieved by solving the quadratic minimization problem. The optimal solution satisfies the condition that the partial derivative of the following quadratic cost function with respect to $\mathbf{\Phi}$ is zero (hereafter, the superscript (t) is omitted for simplicity):

$$E(\mathbf{\Phi}) = \frac{\rho_1}{2}\|\mathbf{Z}_1 - \mathbf{\Phi B\Sigma} - \mathbf{\Theta}_1\|_F^2 + \frac{\rho_2}{2}\|\mathbf{Z}_2 - \mathbf{\Phi} - \mathbf{\Theta}_2\|_F^2 + \frac{\rho_3}{2}\|\mathbf{Z}_3 - \mathbf{\Phi B\Sigma} - \mathbf{\Theta}_3\|_F^2 + \frac{\rho_4}{2}\|\mathbf{Z}_4 - \mathbf{\Phi B\Sigma} - \mathbf{\Theta}_4\|_F^2. \tag{24}$$

By setting the first-order derivative to zero, the optimal solution is determined by solving the system of linear equations:

$$\begin{aligned} &\boldsymbol{\Phi}\boldsymbol{\Psi} = \boldsymbol{\Xi}, \\ &\begin{cases} \boldsymbol{\Psi} = \rho_1 \mathbf{B}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^*\mathbf{B}^* + \rho_2 \mathbf{I} + \rho_3 \mathbf{B}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^*\mathbf{B}^* + \rho_4 \mathbf{B}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^*\mathbf{B}^*, \\ \boldsymbol{\Xi} = \rho_2(\mathbf{Z}_1 - \boldsymbol{\Theta}_1)\boldsymbol{\Sigma}^*\mathbf{B}^* + \rho_2(\mathbf{Z}_2 - \boldsymbol{\Theta}_2) + \rho_3(\mathbf{Z}_3 - \boldsymbol{\Theta}_3)\boldsymbol{\Sigma}^*\mathbf{B}^* + \rho_4(\mathbf{Z}_4 - \boldsymbol{\Theta}_4)\boldsymbol{\Sigma}^*\mathbf{B}^*, \end{cases} \end{aligned} \tag{25}$$

where $\mathbf{I} \in \mathbb{R}^{M\times M}$ is the identity matrix. The optimal solution is obtained by the inverse problem $\boldsymbol{\Phi}^\star = \boldsymbol{\Xi}\boldsymbol{\Psi}^{-1}$.

The updates of $\mathbf{Z}_1$ and $\mathbf{Z}_2$ in steps 5 and 6 of Algorithm 2 can be accomplished by employing the Gaussian denoiser $\mathcal{D}_{\mathcal{R}_{r,\alpha/\rho_1}}$ and $\mathcal{D}_{\mathcal{R}_{m,(1-\alpha)/\rho_2}}$ as regularization terms $\mathcal{R}_r$ and $\mathcal{R}_m$, respectively. In our experiments, we utilized the TV algorithm discussed in Section 2.4 or BM3D [22] for both denoisers of $\mathcal{D}_{\mathcal{R}_{r,\alpha/\rho_1}}$ and $\mathcal{D}_{\mathcal{R}_{m,(1-\alpha)/\rho_2}}$. These Gaussian denoisers can restore smooth reconstructed frames and DMD modes while effectively removing noise.

The updates of $\mathbf{Z}_3$ and $\mathbf{Z}_4$ in steps 7 and 8 of Algorithm 2 require the computation of the proximity operators for the indicator functions of $\iota_{\mathcal{B}^F_{\mathbf{Y},\epsilon}}(\cdot)$ and $\iota_{\mathbb{R}^{N\times M}}(\cdot)$, which are equivalent to the metric projections onto them. Similar to Equation (13), the metric projection onto $\mathcal{B}^F_{\mathbf{Y},\epsilon}$ is given by

$$\mathcal{P}_{\mathcal{B}^F_{\mathbf{Y},\epsilon}}(\mathbf{X}) = \begin{cases} \mathbf{X}, & \text{if } \mathbf{X} \in \mathcal{B}^F_{\mathbf{Y},\epsilon}, \\ \mathbf{Y} + \epsilon \frac{(\mathbf{X}-\mathbf{Y})}{\|\mathbf{X}-\mathbf{Y}\|_F}, & \text{otherwise.} \end{cases} \tag{26}$$

Then, the metric projection onto $\mathbb{R}^{N\times M}$ is given by $\mathcal{P}_{\mathbb{R}^{N\times M}}(\mathbf{X}) = \text{real}(\mathbf{X})$, where $\text{real}(\mathbf{X})$ is the real part of $\mathbf{X}$.

4. Experiments

To demonstrate the effectiveness of the proposed method, we applied it to several noisy videos and compared it with naive noise removal methods in which the TV and BM3D denoisers were applied directly to the video frames. We refer to these methods as "naive TV" and "naive BM3D". These results were obtained by setting α to 1, and these methods did not consider any regularization for the DMD mode. By comparing with such naive noise removal approaches, we can confirm the effectiveness of our proposed method that considers regularization for the DMD mode, namely the effectiveness of explicitly applying denoisers to the DMD mode. We refer to our method with the TV denoiser and the BM3D denoiser as "Ours with TV" and "Ours with BM3D".

Figure 1 shows the original video scenes used for experiments. Scene 1 and Scene 2 videos were captured by the authors, Scene 3 and Scene 4 videos were selected from the SBMnet dataset [35], and Scene 5 video was selected from the DAVIS dataset [36]. We extracted $M = 10, 20, 30$ [frames] from the first frame of each video. For the sake of simplicity, a color video was converted to grayscale. The details of each scene are briefly summarized as follows:

Scene 1 Scene 2 Scene 3 Scene 4 Scene 5

Figure 1. Input video scenes. Scene 1: A person walks in front of a simple stationary background. Scene 2: A bicycle passes in front of a complex stationary background. Scene 3: Some people walk in front of a simple stationary background. Scene 4: A car passes in front of a dynamic background. Scene 5: Background and foreground move simultaneously. The camera is not fixed.

We independently added AWGN with three intensities, i.e., $15/255, 25/255$, and $35/255$, to input video frames. The visually best results with the proposed method were obtained by setting $\epsilon = 0.95\sqrt{NM\sigma^2}$ and adjusting the value of α from 0 to 1 in steps of 0.1. For the quality metrics, we used the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM) [37]. If the structure similarity between the input and reference images is high, the SSIM value is closer to 1 (for details, see [37]).

Tables 1 and 2 show the average PSNR and SSIM of all frames obtained by the proposed and naive methods. In the cases of "Ours with TV" and "Ours with BM3D", the values of α that yielded the best results are shown in brackets. One can see from Table 1 that "Ours with BM3D" has the highest average PSNR values compared to naive TV, naive BM3D, and "Ours with TV". Then, one observes from Table 2 that "Ours with BM3D" has higher average SSIM values than the other methods in most cases. However, in Scene 2, "Ours with TV" tends to have higher values than the other methods. This is because BM3D cannot preserve the complex texture of the background concrete wall, and it is lost due to over-smoothing. Thus, the SSIM values of BM3D denoiser-based methods tend to be lower than those of TV denoiser-based methods. It has been observed that "Ours with TV" has higher PSNR and SSIM values than naive TV, regardless of noise intensity in the cases of $M = 10$ and 20. However, in the case of $M = 30$ and $\sigma = 35/255$, the values of "Ours with TV" are lower than those of naive TV. As frames increase, the DMD algorithm yields more DMD modes, including high-frequency modes representing fine vibrational components. The TV denoiser is suitable for improving spatial smoothness but does not preserve textures like repeating patterns. The estimated DMD modes by "Ours with TV" have fewer high-frequency components, so PSNR and SSIM values deteriorate.

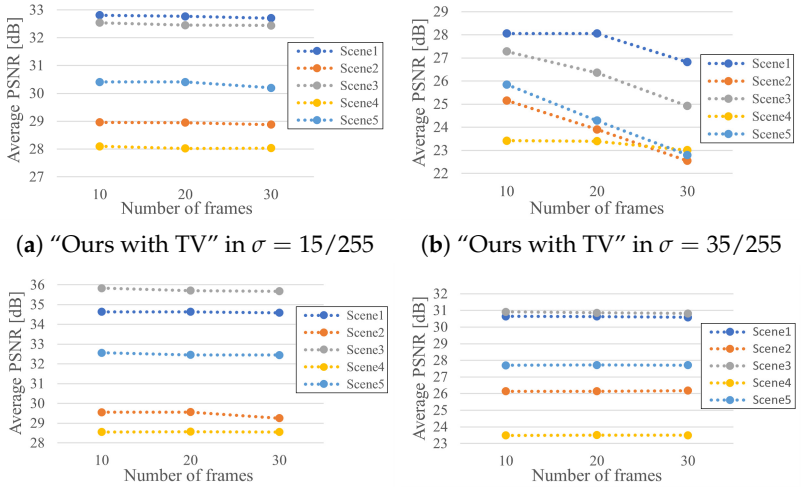
Table 1. Average PSNR comparison.

Scene	σ	$M = 10$ [Frame]					$M = 20$ [Frame]					$M = 30$ [Frame]				
		Noisy	Naive TV	Naive BM3D	Ours with TV (α)	Ours with BM3D (α)	Noisy	Naive TV	Naive BM3D	Ours with TV (α)	Ours with BM3D (α)	Noisy	Naive TV	Naive BM3D	Ours with TV (α)	Ours with BM3D (α)
1	15/255	24.62	32.29	34.14	32.81 (0.4)	**34.64** (0.1)	24.62	32.27	33.06	32.77 (0.5)	**34.64** (0.1)	24.62	32.27	33.05	32.70 (0.6)	**34.59** (0.1)
	25/255	20.23	30.21	31.81	30.31 (0.8)	**32.24** (0.3)	20.23	30.28	31.65	30.25 (0.9)	**32.19** (0.3)	20.23	29.95	31.65	30.25 (0.9)	**32.18** (0.3)
	35/255	17.45	28.53	30.50	28.06 (0.9)	30.64 (0.5)	17.45	28.53	30.49	28.06 (0.9)	**30.63** (0.5)	17.45	27.56	30.36	26.82 (0.9)	**30.59** (0.6)
2	15/255	24.66	28.37	29.46	28.96 (0.5)	**29.55** (0.4)	24.66	28.36	29.50	28.95 (0.6)	**29.56** (0.6)	24.66	28.37	28.78	28.88 (0.8)	**29.25** (0.1)
	25/255	20.32	26.34	27.47	26.55 (0.7)	**27.48** (0.8)	20.32	26.52	27.00	26.54 (0.9)	**27.28** (0.4)	20.32	26.29	27.04	26.02 (0.9)	**27.32** (0.4)
	35/255	17.55	25.18	26.09	25.16 (0.9)	**26.14** (0.5)	17.55	25.02	26.00	23.91 (0.9)	**26.14** (0.5)	17.54	24.34	26.03	22.55 (0.9)	**26.18** (0.5)
3	15/255	24.76	31.85	34.80	32.54 (0.4)	**35.83** (0.1)	24.76	31.85	34.77	32.45 (0.6)	**35.71** (0.2)	24.76	32.23	34.73	32.44 (0.6)	**35.68** (0.1)
	25/255	20.44	29.25	32.74	29.63 (0.8)	**33.08** (0.3)	20.44	29.57	32.74	29.61 (0.9)	**33.07** (0.3)	20.45	29.41	32.70	29.09 (0.9)	**33.03** (0.3)
	35/255	17.64	27.57	30.80	27.28 (0.9)	**30.91** (0.7)	17.65	26.92	30.73	26.36 (0.9)	**30.86** (0.6)	17.65	25.60	30.72	24.92 (0.9)	**30.82** (0.6)
4	15/255	24.75	26.84	28.49	28.10 (0.3)	**28.55** (0.4)	24.74	26.90	28.51	28.02 (0.5)	**28.57** (0.6)	24.74	26.91	28.51	28.03 (0.6)	**28.55** (0.7)
	25/255	20.43	24.31	25.66	25.17 (0.5)	**25.67** (0.7)	20.43	24.93	**25.27**	25.16 (0.8)	**25.27** (0.2)	20.43	24.93	25.28	25.16 (0.8)	25.40 (0.2)
	35/255	17.66	23.16	23.30	23.42 (0.8)	**23.49** (0.3)	17.66	23.42	23.25	23.40 (0.9)	**23.51** (0.3)	17.66	23.24	23.25	23.01 (0.9)	**23.50** (0.3)
5	15/255	24.71	30.00	31.22	30.41 (0.6)	**32.56** (0.2)	24.70	29.99	31.12	30.41 (0.7)	**32.46** (0.2)	24.70	30.34	31.12	30.20 (0.9)	**32.45** (0.2)
	25/255	20.41	27.43	29.26	27.68 (0.8)	**29.67** (0.4)	20.39	27.67	29.18	27.57 (0.9)	**29.66** (0.3)	20.38	27.09	29.17	26.70 (0.9)	**29.65** (0.4)
	35/255	17.64	25.92	27.55	25.85 (0.9)	**27.70** (0.7)	17.61	24.81	27.67	24.29 (0.9)	**27.72** (0.9)	17.61	23.33	27.66	22.80 (0.9)	**27.71** (0.6)

Table 2. Average SSIM comparison.

Scene	σ	$M = 10$ [Frame]					$M = 20$ [Frame]					$M = 30$ [Frame]				
		Noisy	Naive TV	Naive BM3D	Ours with TV (α)	Ours with BM3D (α)	Noisy	Naive TV	Naive BM3D	Ours with TV (α)	Ours with BM3D (α)	Noisy	Naive TV	Naive BM3D	Ours with TV (α)	Ours with BM3D (α)
1	15/255	0.4092	0.8536	0.8857	0.8618 (0.5)	**0.8953** (0.1)	0.4094	0.8536	0.8620	0.8609 (0.6)	**0.8955** (0.1)	0.4095	0.8534	0.8617	0.8590 (0.7)	**0.8933** (0.1)
	25/255	0.2383	0.8042	0.8374	0.8044 (0.9)	**0.8502** (0.3)	0.2381	0.8017	0.8323	0.7909 (0.9)	**0.8484** (0.2)	0.2381	0.7613	0.8324	0.7904 (0.9)	**0.8480** (0.2)
	35/255	0.1591	0.7215	0.8077	0.6779 (0.9)	**0.8135** (0.5)	0.1591	0.7222	0.8075	0.6783 (0.9)	**0.8133** (0.5)	0.1592	0.6367	0.7929	0.5783 (0.9)	**0.8107** (0.7)
2	15/255	0.6173	0.7465	0.7792	**0.7865** (0.5)	0.7822 (0.4)	0.6197	0.7501	0.7833	**0.7912** (0.6)	0.7854 (0.6)	0.6202	0.7514	0.7370	**0.7887** (0.8)	0.7659 (0.1)
	25/255	0.4191	0.6513	0.6842	0.6800 (0.7)	**0.6852** (0.8)	0.4226	0.6760	0.6446	**0.6824** (0.9)	0.6687 (0.2)	0.4233	**0.6757**	0.6487	0.6641 (0.9)	0.6725 (0.2)
	35/255	0.3022	0.6031	0.5991	**0.6084** (0.9)	0.6045 (0.5)	0.3062	0.6097	0.5939	0.5577 (0.9)	**0.6114** (0.5)	0.3066	0.5795	0.5982	0.4961 (0.9)	**0.6159** (0.5)
3	15/255	0.4541	0.8882	0.9123	0.8918 (0.6)	**0.9241** (0.1)	0.4541	0.8879	0.9118	0.8903 (0.8)	**0.9216** (0.1)	0.4556	0.8868	0.9115	0.8900 (0.8)	**0.9218** (0.1)
	25/255	0.2913	0.8449	0.8907	0.8445 (0.9)	**0.8966** (0.3)	0.2913	0.8394	0.8897	0.8263 (0.9)	**0.8950** (0.5)	0.2925	0.7969	0.8891	0.7633 (0.9)	**0.8940** (0.3)
	35/255	0.2082	0.7530	0.8658	0.7108 (0.9)	**0.8679** (0.5)	0.2081	0.6728	0.8639	0.6200 (0.9)	**0.8661** (0.5)	0.2088	0.5611	0.8554	0.5139 (0.9)	**0.8651** (0.5)
4	15/255	0.7696	0.8540	0.8963	0.8876 (0.4)	**0.8965** (0.8)	0.7676	0.8545	0.8966	0.8821 (0.5)	**0.8969** (0.6)	0.7678	0.8545	0.8966	0.8863 (0.6)	**0.8972** (0.1)
	25/255	0.6069	0.7631	0.8174	0.8062 (0.5)	**0.8178** (0.1)	0.6045	0.7955	0.7971	0.8050 (0.8)	**0.8175** (0.1)	0.6043	0.7952	0.7971	0.8047 (0.8)	**0.8076** (0.2)
	35/255	0.4835	0.7177	0.7058	0.7365 (0.8)	**0.7530** (0.2)	0.4813	0.7350	0.7032	0.7347 (0.9)	**0.7526** (0.2)	0.4810	0.7255	0.7031	0.7138 (0.9)	**0.7523** (0.2)
5	15/255	0.5465	0.8786	0.8990	0.8810 (0.7)	**0.9167** (0.1)	0.5442	0.8766	0.8962	0.8774 (0.9)	**0.9141** (0.1)	0.5457	0.8511	0.8956	0.8350 (0.9)	**0.9140** (0.1)
	25/255	0.3851	0.8186	0.8663	0.8142 (0.9)	**0.8761** (0.4)	0.3821	0.7947	0.8628	0.7719 (0.9)	**0.8734** (0.3)	0.3819	0.7175	0.8612	0.6849 (0.9)	**0.8725** (0.3)
	35/255	0.2914	0.7450	0.8319	0.7178 (0.9)	**0.8341** (0.7)	0.2883	0.6049	0.8228	0.5676 (0.9)	**0.8310** (0.6)	0.2874	0.5102	0.8205	0.4828 (0.9)	**0.8292** (0.5)

Figure 2a,b illustrate the PSNR trends for "Ours with TV" under AWGN with $\sigma = 15/255$ and $35/255$, respectively. They indicate that as the frame count rises, particularly under high noise levels, the performance of "Ours with TV" declines due to the increasing complexity of the DMD mode. The TV denoiser struggles to preserve minor DMD mode changes. Figure 2c,d demonstrate that "Ours with BM3D" maintains consistently high PSNR values, even with more frames and elevated noise levels. This stability is attributed to BM3D's ability to exploit similar patches in the high-frequency DMD mode effectively. Similar trends were observed in the average SSIM values.



(**a**) "Ours with TV" in $\sigma = 15/255$ (**b**) "Ours with TV" in $\sigma = 35/255$

(**c**) "Ours with BM3D" in $\sigma = 15/255$ (**d**) "Ours with BM3D" in $\sigma = 35/255$

Figure 2. Relationship between number of frames and PSNR of the proposed methods.

Figure 3 shows some close-ups of Scenes 1 and 4 degraded by AWGN with the standard deviation $\sigma = 25/255$. In Scene 1, "Ours with BM3D" and naive BM3D can remove noise, preserving the human silhouette, while "Ours with BM3D" shows superior preservation of wood fine texture. However, "Our with TV" and naive TV only partially restore sharp edges and textures. In Scene 4, "Ours with BM3D" and naive BM3D successfully remove noise while maintaining the car and wood silhouette, whereas "Ours with TV" and naive TV cannot restore sharp edges. Notably, "Ours with BM3D" outperforms naive BM3D in restoring tree edges and textures. Its direct noise removal in the DMD mode effectively preserves high-frequency modes even in areas with motion and complex textures.

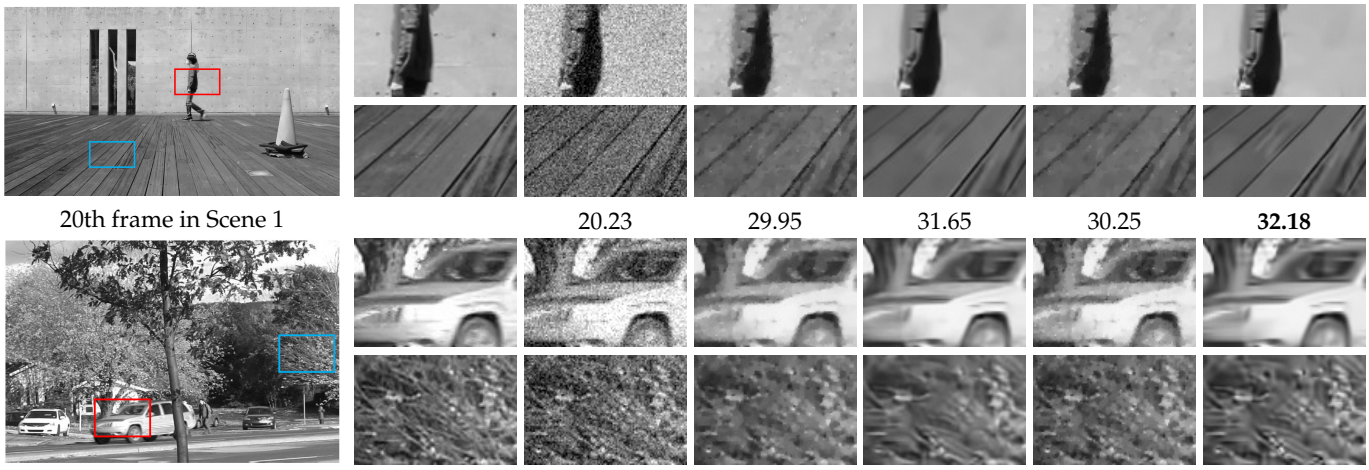
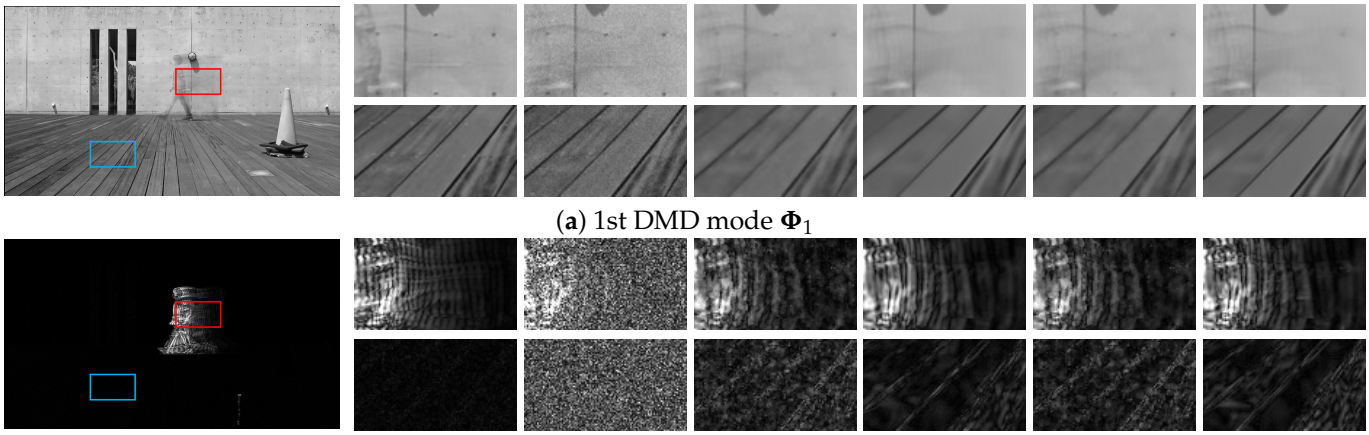


Figure 3. Experimental results of (**top**) the 20th frame in Scene 1 and (**bottom**) the 10th frame in Scene 4 and their PSNR values [dB]. The close-up images indicated by the red and blue boxes are shown as follows: (from left to right) reference frame, input frame, naive TV, naive BM3D, Ours with TV using the best α, and Ours with BM3D using the best α.

Figure 4 shows some close-ups of the DMD mode of Scene 1 in the case of AWGN with $\sigma = 25/255$. This figure shows that "Ours with BM3D" can remove noise while preserving the edges and rich textures of the DMD modes $\mathbf{\Phi}_1$ and $\mathbf{\Phi}_{20}$. The restoration results of naive BM3D can effectively remove noise similar to "Ours with BM3D". However, due to the implicit restoration of the DMD mode, its restoration accuracy seems inferior to "Ours with BM3D". In contrast, "Ours with BM3D" can restore the textures of the DMD mode more clearly than naive BM3D. This is because noise removal can be applied directly to the DMD mode, resulting in preserved edges and textures. Although "Ours with TV" can reduce noise better than naive TV, it is less effective at removing noise than both methods based on the BM3D denoiser, especially in the high-frequency mode $\mathbf{\Phi}_{20}$.



(**a**) 1st DMD mode $\mathbf{\Phi}_1$

(**b**) 20th DMD mode $\mathbf{\Phi}_{20}$

Figure 4. Results of the estimated DMD modes in Scene 1 (**a**) $\mathbf{\Phi}_1$ and (**b**) $\mathbf{\Phi}_{20}$. The close-up images indicated by the red and blue boxes are shown as follows: (from left to right) reference frame, input frame, naive TV, naive BM3D, "Ours with TV" using the best α, and "Ours with BM3D" using the best α.

Next, we applied "Ours with BM3D" and "Ours with TV" to real noisy video captured with a high ISO setting in low light conditions and compared the results with those of naive TV and naive BM3D to show their effectiveness on real video. Figure 5 shows some close-ups of the resulting images with the gamma correction set to $\gamma = 1.3$ for better visibility. This figure shows that "Ours with BM3D" effectively preserves the edge of the fence and has a better noise removal effect than naive TV, naive BM3D, and "Ours with TV". Conversely, "Ours with TV" is better at preserving complex details, such as concrete wall patterns that are difficult to recover with the BM3D denoiser.

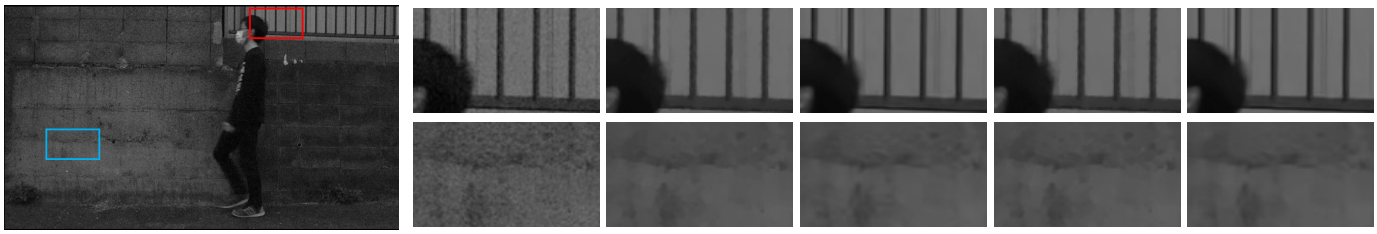


Figure 5. Results of the 20th frame in a real scene captured with high ISO setting. The close-up images indicated by the red and blue boxes are shown as follows: (from left to right) input frame, naive TV, naive BM3D, "Ours with TV" using the best α, and "Ours with BM3D" using the best α.

Finally, we discuss the computational cost of the proposed method. All experiments were conducted using MATLAB R2021a on a system equipped with an AMD EPYC 7402P 2.80 GHz processor and 128 GB RAM. Our method uses the iterative algorithm based on the PnP-ADMM framework, where each iteration requires a denoiser computation. The

TV denoiser requires iterative computation using the ADMM algorithm (see Section 2.4 for details). Fortunately, as shown in the reference [21], it is possible to perform fast computations using the fast Fourier transform to solve the quadratic minimization problem with respect to $\mathbf{x}$ in Equation (16). Therefore, this denoiser can be executed in a relatively short computation time. The BM3D denoiser requires iterative computation using a non-local mean algorithm. Specifically, it is necessary to repeatedly search for many patches similar to the target patch from a wide range around it, which requires high computational cost and relatively long execution time. Figure 6 shows the average computational time when naive TV, naive BM3D, "Ours with the TV", and "Ours with BM3D" on a video with an image size of 256×256. The execution time when applying each method to a video with a frame size of 10 was as follows:

- Naive TV was performed in less than 10 s.
- Naive BM3D was performed in about 25 s.
- "Ours with TV" was performed in about 20 s and shorter execution time than naive BM3D.
- "Ours with BM3D" was performed in about 75 s and about three times slower than naive BM3D.

"Ours with BM3D" achieves the highest noise removal accuracy among the compared methods. However, it also requires a longer computational time. Despite taking roughly three times longer than naive BM3D, we still consider it practical. Furthermore, the figure illustrates that the computation time for each method increases proportionally with the number of frames.

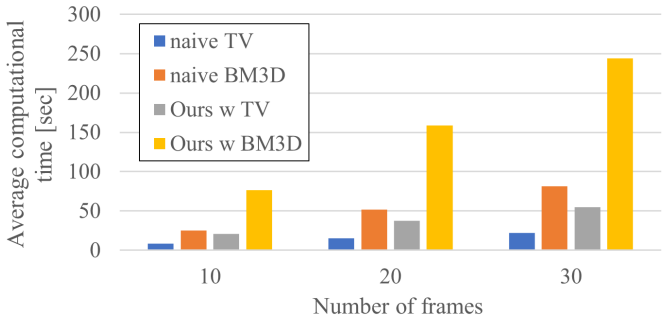


Figure 6. Average computational time measurement results.

5. Conclusions

In this paper, we introduced a novel noise removal method for the DMD mode of a noisy video. Specifically, the minimization problem that simultaneously reduces the noise of the DMD mode and the reconstructed video was defined. Then, we solved the proposed problem using the PnP-ADMM algorithm. The experiments confirmed that the proposed method can effectively remove noise in the DMD mode and the reconstructed video. These results suggest the potential to provide more reliable results in image recognition and object detection, especially in video surveillance and object tracking applications where foreground and background separation is essential. The proposed method consistently demonstrated effectiveness over naive noise removal methods throughout the experiments.

In future work, we will employ stochastic gradient descent algorithms to improve the computational efficiency of the proposed PnP-ADMM algorithm. We will also apply the proposed method to other high-dimensional volume data noise removal problems, e.g., hyperspectral and CT/MRI imaging.

Author Contributions: Methodology, H.Y., S.A. and R.M.; software, H.Y., S.A. and R.M.; validation, H.Y., S.A. and R.M.; formal analysis, R.M.; investigation, R.M.; data curation, R.M.; writing—original draft preparation, H.Y., S.A. and R.M.; writing—review and editing, R.M.; project administration, R.M.; funding acquisition, R.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by JSPS KAKENHI Grant Number 21K17767 and MEXT Promotion of Distinctive Joint Research Center Program Grant Number #JPMXP 0621467946. The experiments in this paper were performed using DeeplearningBOX/Alpha Workstation at The University of Kitakyushu.

Data Availability Statement: Data will be shared with interested third parties on reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Grosek, J.; Kutz, J.N. Dynamic mode decomposition for real-time background/foreground separation in video. *arXiv* **2014**, arXiv:1404.7592.
2. Kutz, J.N.; Fu, X.; Brunton, S.L.; Erichson, N.B. Multi-resolution Dynamic Mode Decomposition for Foreground/Background Separation and Object Tracking. In Proceedings of the IEEE International Conference on Computer Vision Workshop (ICCVW), Santiago, Chile, 7–13 December 2015; pp. 921–929. [CrossRef]
3. Kutz, J.N.; Grosek, J.; Brunton, S.L. Dynamic mode decomposition for robust pca with applications to foreground/background subtraction in video streams and multi-resolution analysis. In *CRC Handbook on Robust Low-Rank and Sparse Matrix Decomposition: Applications in Image and Video Processing*; CRC Press: Boca Raton, FL, USA, 2016.
4. Erichson, N.B.; Donovan, C. Randomized low-rank dynamic mode decomposition for motion detection. *Comput. Vis. Image Underst.* **2016**, *146*, 40–50. [CrossRef]
5. Dicle, C.; Mansour, H.; Tian, D.; Benosman, M.; Vetro, A. Robust low-rank dynamic mode decomposition for compressed domain crowd and traffic flow analysis. In Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), Seattle, WA, USA, 11–15 July 2016; pp. 1–6. [CrossRef]
6. Pendergrass, S.; Brunton, S.L.; Kutz, J.N.; Erichson, N.B.; Askham, T. Dynamic Mode Decomposition for Background Modeling. In Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 22–29 October 2017; pp. 1862–1870. [CrossRef]
7. Takeishi, N.; Kawahara, Y.; Yairi, T. Sparse nonnegative dynamic mode decomposition. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 2682–2686. [CrossRef]
8. Bi, C.; Yuan, Y.; Zhang, J.; Shi, Y.; Xiang, Y.; Wang, Y.; Zhang, R. Dynamic Mode Decomposition Based Video Shot Detection. *IEEE Access* **2018**, *6*, 21397–21407. [CrossRef]
9. Erichson, N.B.; Brunton, S.L.; Kutz, J.N. Compressed dynamic mode decomposition for background modeling. *J. Real-Time Image Process.* **2019**, *16*, 1479–1492. [CrossRef]
10. Schmid, P.J. Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.* **2010**, *656*, 5–28. [CrossRef]
11. Mezić, I. Analysis of Fluid Flows via Spectral Properties of the Koopman Operator. *Annu. Rev. Fluid Mech.* **2013**, *45*, 357–378. [CrossRef]
12. Kutz, J.N.; Brunton, S.L.; Brunton, B.W.; Proctor, J.L. *Dynamic Mode Decomposition*; Socoety for Industiral and Applied Mathemathics: Philadelphia, PA, USA, 2016. [CrossRef]
13. Dawson, S.T.; Hemati, M.S.; Williams, M.O.; Rowley, C.W. Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition. *Exp. Fluids* **2016**, *57*, 42. [CrossRef]
14. Hemati, M.S.; Rowley, C.W.; Deem, E.A.; Cattafesta, L.N. De-biasing the dynamic mode decomposition for applied Koopman spectral analysis of noisy datasets. *Theor. Comput. Fluid Dyn.* **2017**, *31*, 349–368. [CrossRef]
15. Rudin, L.I.; Osher, S.; Fatemi, E. Nonlinear Total Variation Based Noise Removal Algorithms. *Phys. D* **1992**, *60*, 259–268. [CrossRef]
16. Bresson, X.; Chan, T.F. Fast Dual Minimization of the Vectorial Total Variation Norm and Applications to Color Image Processing. *Inverse Probl. Imag.* **2008**, *2*, 455–484. [CrossRef]
17. Chambolle, A. An Algorithm for Total Variation Minimization and Applications. *J. Math. Imag. Vis.* **2004**, *20*, 89–97. [CrossRef]
18. Blomgren, P.; Chan, T.F. Color TV: Total variation methods for restoration of vector-valued images. *IEEE Trans. Image Process.* **1998**, *7*, 304–309. [CrossRef] [PubMed]
19. Chan, S.H.; Khoshabeh, R.; Gibson, K.B.; Gill, P.E.; Nguyen, T.Q. An augmented Lagrangian method for total variation video restoration. *IEEE Trans. Image Process.* **2011**, *20*, 3097–3111. [CrossRef] [PubMed]
20. Matsuoka, R.; Ono, S.; Okuda, M. Transformed-Domain Robust Multiple-Exposure Blending With Huber Loss. *IEEE Access* **2019**, *7*, 162282–162296. [CrossRef]
21. Matsuoka, R.; Okuda, M. Beyond Staircasing Effect: Robust Image Smoothing via ℓ_0 Gradient Minimization and Novel Gradient Constraints. *Signals* **2023**, *4*, 669–686. [CrossRef]
22. Dabov, K.; Foi, A.; Egiazarian, K. Video denoising by sparse 3D transform-domain collaborative filtering. In Proceedings of the 2007 15th European Signal Processing Conference, Poznan, Poland, 3–7 September 2007; pp. 145–149.
23. Dabov, K.; Foi, A.; Katkovnik, V.; Egiazarian, K. Color image denoising via sparse 3D collaborative filtering with grouping constraint in luminance-chrominance space. In Proceedings of the 2007 IEEE International Conference on Image Processing, San Antonio, TX, USA, 16 September–19 October 2007; Volume 1, pp. 1–313.

24. Maggioni, M.; Boracchi, G.; Foi, A.; Egiazarian, K. Video denoising, deblocking, and enhancement through separable 4-D nonlocal spatiotemporal transforms. *IEEE Trans. Image Process.* **2012**, *21*, 3952–3966. [CrossRef] [PubMed]
25. Mäkinen, Y.; Azzari, L.; Foi, A. Collaborative filtering of correlated noise: Exact transform-domain variance for improved shrinkage and patch matching. *IEEE Trans. Image Process.* **2020**, *29*, 8339–8354. [CrossRef] [PubMed]
26. Anami, S.; Matsuoka, R. Noise Removal for Dynamic Mode Decomposition Based on Plug-and-Play ADMM. In Proceedings of the 2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Tokyo, Japan, 14–17 December 2021; pp. 1405–1409.
27. Tu, J.H. Dynamic Mode Decomposition: Theory and Applications. Ph.D. Thesis, Princeton University, Princeton, NJ, USA, 2013.
28. Brunton, S.L.; Proctor, J.L.; Tu, J.H.; Kutz, J.N. Compressed sensing and dynamic mode decomposition. *J. Comput. Dyn.* **2015**, *2*, 165. [CrossRef]
29. Gabay, D.; Mercier, B. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Comput. Math. Appl.* **1976**, *2*, 17–40. [CrossRef]
30. Venkatakrishnan, S.V.; Bouman, C.A.; Wohlberg, B. Plug-and-play priors for model-based reconstruction. In Proceedings of the IEEE Global Conference on Signal and Information Processing, Austin, TX, USA, 3–5 December 2013; pp. 945–948.
31. Chan, S.H.; Wang, X.; Elgendy, O.A. Plug-and-play ADMM for image restoration: Fixed-point convergence and applications. *IEEE Trans. Comput. Imag.* **2016**, *3*, 84–98. [CrossRef]
32. Moreau, J.J. Fonctions convexes duales et points proximaux dans un espace hilbertien. *C. R. Acad. Sci.* **1962**, *255*, 2897–2899.
33. Combettes, P.L.; Pesquet, J.C. A proximal decomposition method for solving convex variational inverse problems. *Inverse Probl.* **2008**, *24*, 065014. [CrossRef]
34. Duval, V.; Aujol, J.F.; Vese, L.A. Mathematical Modeling of Textures: Application to Color Image Decomposition with a Projected Gradient Algorithm. *J. Math. Imag. Vis.* **2010**, *37*, 232–248. [CrossRef]
35. Jodoin, P.M.; Maddalena, L.; Petrosino, A.; Wang, Y. Extensive Benchmark and Survey of Modeling Methods for Scene Background Initialization. *IEEE Trans. Image Process.* **2017**, *26*, 5244–5256. [CrossRef] [PubMed]
36. Pont-Tuset, J.; Perazzi, F.; Caelles, S.; Arbelaez, P.; Sorkine-Hornung, A.; Gool, L.V. The 2017 DAVIS Challenge on Video Object Segmentation. *arXiv* **2017**, arXiv:1704.00675.
37. Wang, Z.; Bovik, A.; Sheikh, H.; Simoncelli, E. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]