Generalized Tensor Total Variation Minimization for Visual Data Recovery*

Xiaojie Guo[1] and Yi Ma[2]
[1]State Key Laboratory of Information Security, IIE, CAS, Beijing, 100093, China
[2]School of Information Science and Technology, ShanghaiTech University, Shanghai, 200031, China
xj.max.guo@gmail.com | mayi@shanghaitech.edu.cn

Abstract

In this paper, we propose a definition of Generalized Tensor Total Variation norm (GTV) that considers both the inhomogeneity and the multi-directionality of responses to derivative-like filters. More specifically, the inhomogeneity simultaneously preserves high-frequency signals and suppresses noises, while the multi-directionality ensures that, for an entry in a tensor, more information from its neighbors is taken into account. To effectively and efficiently seek the solution of the GTV minimization problem, we design a novel Augmented Lagrange Multiplier based algorithm, the convergence of which is theoretically guaranteed. Experiments are conducted to demonstrate the superior performance of our method over the state of the art alternatives on classic visual data recovery applications including completion and denoising.

1. Introduction

In real data analysis applications, we often have to face handling dirty observations, say incomplete or noisy data. Recovering the missing or noise-free data from such observations thus becomes crucial to provide us more precise information to refer to. Besides, compared to 1-D vectors and 2-D matrices, the data encountered in real world applications is more likely to be high order, for instance a multi-spectral image is a 3-D tensor and a color video is a 4-D tensor. This work concentrates on the problem of visual data recovery, *i.e.* restoring tensors of visual data from polluted observations. However, without additional priors, the problem is intractable as it usually has infinitely many solutions and thus, it is apparently impossible to identify which of these candidate solutions is indeed the "correct" one.

Mathematically, under some assumptions, one would want to recover a n-order tensor $\mathcal{T} \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_n}$ from an observation $\mathcal{O} \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_n}$ in presence of noise $\mathcal{N} \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_n}$ by solving the following problem:

$$\min_{\mathcal{T},\mathcal{N}} \Phi(\mathcal{T}) + \lambda \Psi(\mathcal{N}) \ \text{s.t.} \ \mathcal{P}_\Omega(\mathcal{O}) = \mathcal{P}_\Omega(\mathcal{T} + \mathcal{N}), \quad (1)$$

where $\mathcal{P}_\Omega(\cdot)$ is the orthogonal projection operator on the support $\Omega \in \{0,1\}^{D_1 \times D_2 \times \cdots \times D_n}$ that indicates which elements are observed. $\Psi(\mathcal{N})$ is a penalty with respect to noise, which usually adopts ℓ^1 norm that is optimal for Laplacian noise, or Frobenius norm for Gaussian noise. The non-negative parameter λ provides a trade-off between the noise sensitivity and closeness to the observed signal. And $\Phi(\mathcal{T})$ stands for the assumption to make the ill-posed tensor recovery problem well-defined. In literature, there are mainly two lines about the assumption, *i.e.* the low rankness and the low total variation.

The low-rank nature of visual data, say $\Phi(\mathcal{T}) := \text{rank}(\mathcal{T})$, has been focus of considerable research in past years, the effectiveness of which has been witnessed by numerous applications, such as image colorization [15], rectification [18], denoising [5], depth enhancement [10] and subspace learning [13]. As for the tensor recovery task, specifically, Liu *et al.* [9] first introduce the trace norm for tensors to achieve the goal. More recently, Wang *et al.* [14] simultaneously impose the low rank prior and the spatial-temporal consistency for completing videos. Zhang *et al.* [17] design a hybrid singular value thresholding strategy for enhancing the low rankness of the underlying tensors. Although these methods produce very promising results on the visual data with strong global structure, the performance of which would degrade on general tensors of visual data. To mitigate the over-strict constraint, [3] uses factor priors to simultaneously decompose and complete tensors, which follows the tensor factorization framework proposed in [11]. Actually, [3] instead enforces the low rank constraint on the low dimensional representation (submanifold) of the data.

Alternatively, the visual tensor to be recovered should be piece-wise smooth, that is $\Phi(\mathcal{T}) := \|\mathcal{T}\|_{TV}$. In last decades, the advances in matrix Total Variation (TV) minimization have demonstrated its significance as a theoretic

*This work was supported in part by the National Natural Science Foundation of China (No. 61402467) and in part by the Excellent Young Talent Programme through the Institute of Information Engineering, CAS.

foundation for this problem. The TV model was first introduced in [12] as a regularizer to remove noises and handle proper edges in images. A variety of applications [2, 6] have proven its great benefit. Although the TV norm has been extensively studied for matrices, there is not much work on tensors. Yang *et al.* [16] simply extend the TV norm for matrices to higher-order tensors and design an efficient algorithm to solve the optimization problem. But, it only takes care of the variations along fibers (see the definition below), and breaks the original problem into a set of 1-D vector problems [4] to accelerate the procedure, which limits its flexibility (multi-directionality).

1.1. Notations and Tensor Basics

We first define the notations used in this paper. Lowercase letters $(a, b, ...)$ mean scalars and bold lowercase letters $(\boldsymbol{a}, \boldsymbol{b}, ...)$ vectors. Bold uppercase letters $(\boldsymbol{A}, \boldsymbol{B}, ...)$ stand for matrices. The vectorization operation of a matrix $\mathrm{vec}(\boldsymbol{A})$ is to convert a matrix into a vector. For brevity, $\boldsymbol{A}_j$ stands for the j^{th} column of $\boldsymbol{A}$. While bold calligraphic uppercase letters $(\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}, ...)$ represent high order tensors. $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_n}$ denotes an n-order tensor, whose elements are represented by $a_{d_1,d_2,...,d_n} \in \mathbb{R}$, where $1 \leq d_k \leq D_k$ and $1 \leq k \leq n$. $\boldsymbol{a}_{d_1,...,d_{k-1},:,d_{k+1},...,d_n} \in \mathcal{R}^{D_k}$ means the mode-k fiber of $\boldsymbol{\mathcal{A}}$ at $\{d_1, ..., d_{k-1}, d_{k+1}, ..., d_n\}$, which is the higher order analogue of matrix rows and columns.

The Frobenius, ℓ^1 and ℓ^0 norms of $\boldsymbol{\mathcal{A}}$ are respectively defined as $\|\boldsymbol{\mathcal{A}}\|_F := \sqrt{\sum_{d_1,d_2,\cdots,d_n} a^2_{d_1,d_2,...,d_n}}$, $\|\boldsymbol{\mathcal{A}}\|_1 := \sum_{d_1,d_2,\cdots,d_n} |a_{d_1,d_2,...,d_n}|$ and $\|\boldsymbol{\mathcal{A}}\|_0 := \sum_{d_1,d_2,\cdots,d_n} a_{d_1,d_2,...,d_n} \neq 0$. The inner product of two tensors with identical size is computed as $\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle := \sum_{d_1,d_2,\cdots,d_n} a_{d_1,d_2,...,d_n} \cdot b_{d_1,d_2,...,d_n}$. $\boldsymbol{\mathcal{A}} \odot \boldsymbol{\mathcal{B}}$ means the element-wise product of two tensors with same size. The mode-k unfolding of $\boldsymbol{\mathcal{A}}$ is to convert a tensor $\boldsymbol{\mathcal{A}}$ into a matrix, *i.e.* $\mathrm{unfold}(\boldsymbol{\mathcal{A}}, k) = \boldsymbol{A}_{[k]} \in \mathbb{R}^{D_k \times \prod_{i \neq k} D_i}$, while the mode-$k$ folding reshapes $\boldsymbol{A}_{[k]}$ back to $\boldsymbol{\mathcal{A}}$, say $\mathrm{fold}(\boldsymbol{A}_{[k]}, k) = \boldsymbol{\mathcal{A}}$. It is clear that, for any k, $\|\boldsymbol{\mathcal{A}}\|_F = \|\boldsymbol{A}_{[k]}\|_F$, $\|\boldsymbol{\mathcal{A}}\|_1 = \|\boldsymbol{A}_{[k]}\|_1$, $\|\boldsymbol{\mathcal{A}}\|_0 = \|\boldsymbol{A}_{[k]}\|_0$, and $\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle = \langle \boldsymbol{A}_{[k]}, \boldsymbol{B}_{[k]} \rangle$. $\mathcal{S}_\omega[\boldsymbol{\mathcal{A}}]$ represents the uniform shrinkage operator on tensors, the definition of which is that, for each element in $\boldsymbol{\mathcal{A}}$, $\mathcal{S}_\omega[a_{d_1,d_2,...,d_n}] := \mathrm{sgn}(a_{d_1,d_2,...,d_n}) \cdot \max(|a_{d_1,d_2,...,d_n}| - \omega, 0)$. More generally, the non-uniform shrinkage $\mathcal{S}_{\boldsymbol{\mathcal{W}}}[\boldsymbol{\mathcal{A}}]$ extends the uniform one by performing the shrinkage on the elements of $\boldsymbol{\mathcal{A}}$ with thresholds given by corresponding entries of $\boldsymbol{\mathcal{W}}$.

1.2. Motivations of This Work

A natural extension of the TV norm for matrices to higher-order tensors has the following shape [16]:

$$\|\boldsymbol{\mathcal{T}}\|_{TV} := \|f_{\frac{\pi}{2}} * [\mathrm{vec}(\boldsymbol{T}_{[1]}) | \mathrm{vec}(\boldsymbol{T}_{[2]}) | ... | \mathrm{vec}(\boldsymbol{T}_{[n]})]\|_p, \tag{2}$$

where $\|\cdot\|_p$ can be either ℓ^1 norm corresponding to the anisotropic tensor TV norm, or $\ell^{2,1}$ the isotropic one. $f_{\frac{\pi}{2}}$ is the derivative filter in the vertical ($\theta = \frac{\pi}{2}$) direction and $*$ is the operator of convolution. It is obvious that the TV norms for vectors and matrices are two examples of this definition.

However, in practical scenarios, this definition (2) is inadequate mainly due to the following limitations:

- The homogeneous penalty leads to the over-smoothness on high-frequency signal, such as corners and edges in visual data, and creates heavy staircase artifacts, which would significantly disturb the perception of visual data.
- It only considers the variations along fibers, which might miss important information in other directions. Although the responses to multi-directional derivative-like filters can be approximately represented by the gradients, the gap between them remains.

The above two limitations motivate us to propose a more general tensor TV norm for boosting the performance on the visual data recovery problem.

1.3. Contributions of This Work

The contributions of this work are summarized as:

- We propose a Generalized Tensor Total Variation norm (GTV) concerning both the inhomogeneity and the multi-directionality.
- We design an Augmented Lagrange Multiplier based algorithm to efficiently and effectively seek the solution of the GTV minimization problem, the convergence of which is theoretically guaranteed.
- To demonstrate the efficacy and the superior performance of the proposed algorithm in comparison with state-of-the-art alternatives, extensive experiments on several visual data recovery tasks are conducted.

2. Methodology

2.1. Definition and Formulation

Suppose we have a matrix $\boldsymbol{A} \in \mathbb{R}^{D_1 \times D_2}$, the response of $\boldsymbol{A}$ to a directional derivative-like filter f_{θ_*} can be computed by $f_{\theta_*} * \boldsymbol{A}$. The traditional TV norm takes into account only the responses to derivative filters along fibers. In other words, it potentially ignores important details from other directions. One may wonder if the multi-directional response can be represented by the gradients. Indeed, for differentiable functions, the directional derivatives along some directions have an equivalent relationship with the gradients. However, for tensors of visual data, this relationship no longer holds as the differentiability is violated. Based on

this fact, we here propose a definition of the responses of a matrix to m-directional derivative-like filters as follows:

Definition 1. *(RMDF: Response to Multi-directional Derivative-like Filters.) The response of a matrix to m derivative-like filters in θ_m directions with weights β_m is defined as $\mathfrak{R}(\boldsymbol{A}, \boldsymbol{\beta}) \in \mathbb{R}^{D_1 D_2 \times m} :=$*

$$\left[\beta_1 \operatorname{vec}(f_{\theta_1} * \boldsymbol{A}) \middle| \beta_2 \operatorname{vec}(f_{\theta_2} * \boldsymbol{A}) \middle| \cdots \middle| \beta_m \operatorname{vec}(f_{\theta_m} * \boldsymbol{A})\right],$$

where $\boldsymbol{\beta} = [\beta_1, \beta_2, \cdots, \beta_m]$ with $\forall j \in [1, ..., m]$ $\beta_j \geq 0$ and $\sum_{j=1}^{m} \beta_j = 1$.

Another drawback of the traditional TV is its homogeneity to all elements in tensors, which favors piecewise constant solutions. This property would result in oversmoothing high-frequency signals and introducing staircase effects. Intuitively, in visual data, the high-frequency signals should be preserved to maintain the perceptual details, while the low-frequency ones could be smoothed to suppress noises. This intuition inspires us to differently treat the variations. Considering both the multi-directionality and the inhomogeneity gives the definition of generalized tensor TV as:

Definition 2. *(GTV: Generalized Tensor Total Variation Norm.) The GTV norm of an n-order tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{D_1 \times \cdots \times D_n}$ is:*

$$\|\boldsymbol{\mathcal{A}}\|_{GTV} := \sum_{k=1}^{n} \alpha^k \|\boldsymbol{W}^k \odot \mathfrak{R}(\boldsymbol{A}_{[k]}, \boldsymbol{\beta}^k)\|_p,$$

where $\boldsymbol{\alpha} = [\alpha^1, ..., \alpha^k, ..., \alpha^n]$ is the non-negative coefficient balancing the importance of k-mode matrices and satisfying $\sum_{k=1}^{n} \alpha^k = 1$, and p could be either 1 or $2, 1$ corresponding to the anisotropic total variation (ℓ^1) and the isotropic one ($\ell^{2,1}$), respectively. In addition, $\boldsymbol{W}^k \in \mathbb{R}^{\prod_{i=1}^{n} D_i \times m}$ acts as the non-negative weight matrix, the elements of which correspond to those of $\mathfrak{R}(\boldsymbol{A}_{[k]}, \boldsymbol{\beta}^k)$.

It is apparent that GTV satisfies the properties that a norm should do, and the traditional TV norm is a specific case of GTV. With the definition of GTV, the visual data recovery problem can be naturally formulated as:

$$\begin{aligned} \underset{\boldsymbol{\mathcal{T}}, \boldsymbol{\mathcal{N}}}{\operatorname{argmin}} & \sum_{k=1}^{n} \alpha^k \|\boldsymbol{W}^k \odot \mathfrak{R}(\boldsymbol{T}_{[k]}, \boldsymbol{\beta}^k)\|_p + \lambda \Psi(\boldsymbol{\mathcal{N}}) \\ & \text{s.t. } \mathcal{P}_\Omega(\boldsymbol{\mathcal{O}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{T}} + \boldsymbol{\mathcal{N}}). \end{aligned} \tag{3}$$

In the next sub-section, we will introduce a novel algorithm to efficiently and effectively solve this problem.

2.2. Optimization

In Eq. (3), $\Psi(\boldsymbol{\mathcal{N}})$ can adopt various forms. In this work, we consider that the noise is either dense Gaussian noise (ℓ^2) or sparse (Laplacian) noise[1] (ℓ^1). To be more general for the ℓ^1 case, we further employ $\|\boldsymbol{\mathcal{W}}_{\mathcal{N}} \odot \boldsymbol{\mathcal{N}}\|_1$. By setting all entries in $\boldsymbol{\mathcal{W}}_{\mathcal{N}}$ to 1, it reduces to $\|\boldsymbol{\mathcal{N}}\|_1$. Although $\Psi(\boldsymbol{\mathcal{N}})$ and p have different options, we do not distinguish them until necessarily. In addition, for the weights $\boldsymbol{W}^k$s, if they are set wisely, the recovery results would be significantly improved, the goal of which is to encourage sharp structured signals by small weights while to discourage smooth regions via large weights. But it is impossible to construct the precise weights without knowing the intrinsic tensors themselves. The situation of $\boldsymbol{\mathcal{W}}_{\mathcal{N}}$ is analogue. Thus, we need to iteratively refine the weights. In this part, we focus on the problem with the weights fixed (inner loop). The reweighting strategy (outer loop) will be discussed in the next section.

As can be seen from Eq. (3), it is difficult to directly optimize because the GTV regularizer breaks the linear structure of $\boldsymbol{\mathcal{T}}$. To efficiently and effectively solve the problem, we introduce auxiliary variables for making the problem separable, which gives the following optimization problem:

$$\begin{aligned} \underset{\boldsymbol{\mathcal{T}}, \boldsymbol{\mathcal{N}}}{\operatorname{argmin}} & \sum_{k=1}^{n} \alpha^k \|\boldsymbol{W}^k \odot \boldsymbol{Q}^k\|_p + \lambda \Psi(\boldsymbol{\mathcal{N}}) \\ \text{s.t. } & \mathcal{P}_\Omega(\boldsymbol{\mathcal{O}}) = \mathcal{P}_\Omega(\boldsymbol{\mathcal{T}} + \boldsymbol{\mathcal{N}}); \forall k \ \ \boldsymbol{Q}^k = \mathfrak{R}(\boldsymbol{T}_{[k]}, \boldsymbol{\beta}^k). \end{aligned} \tag{4}$$

The Augmented Lagrange Multiplier (ALM) with Alternating Direction Minimizing (ADM) strategy can be employed for solving the above problem [8, 7]. The augmented Lagrangian function of (4) is:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\mathcal{T}}, \boldsymbol{\mathcal{N}}, \boldsymbol{Q}^k) := & \sum_{k=1}^{n} \alpha^k \|\boldsymbol{W}^k \odot \boldsymbol{Q}^k\|_p + \lambda \Psi(\mathcal{P}_\Omega(\boldsymbol{\mathcal{N}})) + \\ & \Phi(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{O}} - \boldsymbol{\mathcal{T}} - \boldsymbol{\mathcal{N}}) + \sum_{k=1}^{n} \Phi(\boldsymbol{Y}^k, \boldsymbol{Q}^k - \mathfrak{R}(\boldsymbol{T}_{[k]}, \boldsymbol{\beta}^k)), \end{aligned} \tag{5}$$

with the definition $\Phi(\boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{C}}) := \frac{\mu}{2}\|\boldsymbol{\mathcal{C}}\|_F^2 + \langle \boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{C}} \rangle$, where μ is a positive penalty scalar. $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{Y}^k$ are the Lagrangian multipliers. Besides the Lagrangian multipliers, there are $\boldsymbol{\mathcal{T}}, \boldsymbol{\mathcal{N}}$ and $\boldsymbol{Q}^k$ to solve. The solver iteratively updates one variable at a time by fixing the others. Fortunately, each step has a simple closed-form solution, and hence can be computed efficiently. Below, the solutions of the subproblems are given in the following:

$\boldsymbol{\mathcal{T}}$-subproblem: With other terms fixed, we have:

$$\begin{aligned} \underset{\boldsymbol{\mathcal{T}}}{\operatorname{argmin}} & \Phi(\boldsymbol{\mathcal{X}}^{(t)}, \boldsymbol{\mathcal{O}} - \boldsymbol{\mathcal{T}} - \boldsymbol{\mathcal{N}}^{(t)}) + \\ & \sum_{k=1}^{n} \Phi(\boldsymbol{Y}^{k(t)}, \boldsymbol{Q}^{k(t)} - \mathfrak{R}(\boldsymbol{T}_{[k]}^{(t)}, \boldsymbol{\beta}^k)). \end{aligned} \tag{6}$$

[1] The sparse noise is usually modeled as $\|\boldsymbol{\mathcal{N}}\|_0$, which is non-convex and difficult to approximate (NP-hard), the widely used convex relaxation is to replace ℓ^0 norm with ℓ^1 that is the tightest convex proxy of ℓ^0. With the replacement, the sparse and Laplacian noises have the same form.

For computing $\mathcal{T}^{(t+1)}$, we take derivative of (6) with respect to $\mathcal{T}$ and set it to zero. By applying n-D FFT techniques on this problem, we can efficiently obtain:

$$\mathcal{T}^{(t+1)} = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\mathcal{G}^{(t)})}{\mathbf{1} + \sum_{k=1}^{n}\sum_{j=1}^{m}|\mathcal{F}(\boldsymbol{F}_j^k)|^2}\right), \quad (7)$$

where, for brevity, we denote $\mathcal{G}^{(t)} := \mathcal{O} - \mathcal{N}^{(t)} + \frac{\mathcal{X}^{(t)}}{\mu} + \sum_{k=1}^{n}\text{fold}\left(\sum_{j=1}^{m}(\boldsymbol{F}_j^k)^T(\boldsymbol{Q}_j^{k(t)} + \frac{\boldsymbol{Y}_j^{k(t)}}{\mu}), k\right)$, and $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ stand for the n-D FFT and the inverse n-D FFT operators, respectively. The division in (7) is element-wise. Please notice that $\boldsymbol{F}_j^k$ is the functional matrix corresponding to the directional derivative-like filter $\beta_j^k f_{\theta_j^k}$.

$\mathcal{N}$-subproblem. By dropping unrelated terms, we can update $\mathcal{N}^{(t+1)}$ by minimizing the following problem:

$$\underset{\mathcal{N}}{\text{argmin}}\ \lambda\Psi(\mathcal{P}_\Omega(\mathcal{N})) + \Phi(\mathcal{X}^{(t)}, \mathcal{O} - \mathcal{T}^{(t+1)} - \mathcal{N}). \quad (8)$$

1) **Case Dense Noise** $\Psi(\mathcal{P}_\Omega(\mathcal{N})) := \|\mathcal{P}_\Omega(\mathcal{N})\|_F^2$: Taking derivative of (8) with respect to $\mathcal{N}$ provides:

$$\mathcal{N}^{(t+1)} = \frac{\mathcal{X}^{(t)} + \mu(\mathcal{O} - \mathcal{T}^{(t+1)})}{2\lambda\boldsymbol{\Omega} + \mu}, \quad (9)$$

where the division operates element-wisely.

2) **Case Sparse Noise** $\Psi(\mathcal{P}_\Omega(\mathcal{N})) := \|\mathcal{P}_\Omega(\mathcal{W}_\mathcal{N} \odot \mathcal{N})\|_1$: With the help of the non-uniform shrinkage operator, the closed form solution for this case is:

$$\mathcal{N}^{(t+1)} = \mathcal{S}_{\frac{\mathcal{P}_\Omega(\lambda\mathcal{W}_\mathcal{N})}{\mu}}\left[\mathcal{O} - \mathcal{T}^{(t+1)} + \frac{\mathcal{X}^{(t)}}{\mu}\right]. \quad (10)$$

$\boldsymbol{Q}^k$-subproblem: For each $\boldsymbol{Q}^k$, the update can be done via minimizing the following problem:

$$\underset{\boldsymbol{Q}^k}{\text{argmin}}\ \alpha^k\|\boldsymbol{W}^k \odot \boldsymbol{Q}^k\|_p + \Phi(\boldsymbol{Y}^{k(t)}, \boldsymbol{Q}^k - \mathfrak{R}(\boldsymbol{T}_{[k]}^{(t+1)}, \boldsymbol{\beta}^k)). \quad (11)$$

1) **Case Anisotropic** $p := 1$: The solution of this case can be efficiently computed via:

$$\boldsymbol{Q}^{k(t+1)} = \mathcal{S}_{\frac{\alpha^k\boldsymbol{W}^k}{\mu}}\left[\mathfrak{R}(\boldsymbol{T}_{[k]}^{(t+1)}, \boldsymbol{\beta}^k) - \frac{\boldsymbol{Y}^{k(t)}}{\mu}\right]. \quad (12)$$

2) **Case Isotropic** $p := 2, 1$: Alternatively, the update for this case can be done through:

$$\boldsymbol{Q}^{k(t+1)} = \mathcal{S}_{\frac{\alpha^k\boldsymbol{W}^k}{\mu\boldsymbol{Z}^{k(t)}}}[\mathbf{1}] \odot \left(\mathfrak{R}(\boldsymbol{T}_{[k]}^{(t+1)}, \boldsymbol{\beta}^k) - \frac{\boldsymbol{Y}^{k(t)}}{\mu}\right), \quad (13)$$

where $\mathbf{1}$ is the matrix with the same size as $\boldsymbol{W}^k$, as well all its elements are 1, and the division on the weight performs component-wisely. The m columns of $\boldsymbol{Z}^{k(t)}$ are all $\sqrt{\sum_{j=1}^{m}\left|\left(\mathfrak{R}(\boldsymbol{T}_{[k]}^{(t+1)}, \boldsymbol{\beta}^k) - \frac{\boldsymbol{Y}^{k(t)}}{\mu}\right)_j\right|^2}$, where the operations are also component-wise.

Algorithm 1: GTV Minimization

Input: The observed tensor $\mathcal{O} \in \mathbb{R}^{D_1\times\cdots\times D_n}$ and its support $\boldsymbol{\Omega} \in \{0,1\}^{D_1\times\cdots\times D_n}$; $\lambda \geq 0$; $\forall k \in [1,..,n]$ $\alpha^k \leq 0$ and $\sum_{k=1}^n \alpha^k = 1$; $\boldsymbol{\beta}^k \succeq 0$ and $\sum_{j=1}^m \beta_j^k = 1$.

Initi.: $h = 0$; $\mathcal{W}_\mathcal{N}{}^{(h)} = \mathbf{1} \in \mathbb{R}^{D_1\times\cdots\times D_n}$; $\forall k \in [1,..,n]$ $\boldsymbol{W}^{k(h)} = \mathbf{1} \in \mathbb{R}^{\prod_{i=1}^n D_i\times m}$;

while *not converged* **do**

  $t = 0$; set $\mathcal{T}^{(t)}$, $\mathcal{N}^{(t)}$ and $\mathcal{X}^{(t)}$ to $\mathbf{0} \in \mathbb{R}^{D_1\times\cdots\times D_n}$, $\forall k \in [1,...,n]$ $\boldsymbol{Q}^{k(t)}$ to $\mathbf{0} \in \mathbb{R}^{\prod_{i=1}^n D_i\times m}$, $\mu > 0$;

  while *not converged* **do**

    Update $\mathcal{T}^{(t+1)}$ via Eq. (7);

    Update $\mathcal{N}^{(t+1)}$ via either Eq. (9) for cases with dense Gaussian noises or Eq. (10) sparse noises;

    Update $\boldsymbol{Q}^{k(t+1)}$ via either Eq. (12) for the anisotropic GTV or Eq. (13) the isotropic one;

    Update multipliers via Eq. (14);

    $t = t + 1$;

  end

  Update $\boldsymbol{W}^{k(h+1)}$**s and** $\mathcal{W}_\mathcal{N}{}^{(h+1)}$ through the way discussed in Proposition 2;

  $h = h + 1$;

end

Output: $\mathcal{T}^* = \mathcal{T}^{(t-1)}, \mathcal{N}^* = \mathcal{N}^{(t-1)}$

Multipliers and μ: Besides, there are the multipliers and μ need to be updated, which can be simply accomplished by:

$$\begin{aligned}\mathcal{X}^{(t+1)} &= \mathcal{X}^{(t)} + \mu(\mathcal{O} - \mathcal{T}^{(t+1)} - \mathcal{N}^{(t+1)});\\ \boldsymbol{Y}^{k(t+1)} &= \boldsymbol{Y}^{k(t)} + \mu(\boldsymbol{Q}^{k(t+1)} - \mathfrak{R}(\boldsymbol{T}_{[k]}^{(t+1)}, \boldsymbol{\beta}^k).\end{aligned} \quad (14)$$

For clarity, the procedure of solving the problem (3) is summarized in Algorithm 1. The outer loop terminates when the change of the recovered results between neighboring iterations is sufficiently small or the maximal number of outer iterations is reached. The inner loop is stopped when $\|\mathcal{O} - \mathcal{T}^{(t+1)} - \mathcal{N}^{(t+1)}\|_F \leq \delta\|\mathcal{O}\|_F$ with $\delta = 10^{-6}$ or the maximal number of inner iterations is reached. It is worth nothing that any type of derivative-like filters, such as (directional) first/second derivative, Gaussian and Laplacian, can be readily applied to the proposed general framework.

3. Theoretical Analysis

In this section, we give two key propositions about the convergence and the weight updating strategy to show the

theoretical guarantee of the proposed Algorithm 1.

Proposition 1. *The inner loop of Algorithm 1 for solving the problem (4) converges at a linear rate.*

Proof. Recall the standard form of minimizing a separable convex function subject to linear equality constraints:

$$\min \sum_{i=1}^{q} g_i(\boldsymbol{x}_i) \quad \text{s.t.} \quad \sum_{i=1}^{q} \boldsymbol{E}_i \boldsymbol{x}_i = \boldsymbol{b}. \tag{15}$$

It has proven to be with a linear convergence rate using an ALM-ADM based algorithm, when $g_i(\cdot)$s are convex functions, $\boldsymbol{E}_i$s are the operation matrices, and $\boldsymbol{b}$ is constant [7]. To establish the convergence of the inner loop of Algorithm 1, we transform the problem (4) into the standard form (15). It is easy to connect $g(\boldsymbol{\mathcal{N}}) := \Psi(\boldsymbol{\mathcal{N}})$ and $g_k(\boldsymbol{Q}^k) := \alpha^k \|\boldsymbol{W}^k \odot \boldsymbol{Q}^k\|_p$. Before transforming the constraints, we here denote that $\boldsymbol{o} := \text{vec}(\boldsymbol{O}_{[1]})$, $\boldsymbol{t} := \text{vec}(\boldsymbol{T}_{[1]})$ and $\boldsymbol{n} := \text{vec}(\boldsymbol{N}_{[1]})$, respectively. Now the following constraint is equivalent to those of (4):

$$\begin{bmatrix} \boldsymbol{\Omega} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{o} = \begin{bmatrix} \boldsymbol{\Omega} \\ \boldsymbol{F}_1^1 \\ \vdots \\ \boldsymbol{F}_m^n \end{bmatrix} \boldsymbol{t} + \begin{bmatrix} \boldsymbol{\Omega} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{n} - \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{Q}_1^1 \\ \vdots \\ \boldsymbol{Q}_m^n \end{bmatrix}, \tag{16}$$

where $\boldsymbol{\Omega}$ performs the same with $\mathcal{P}_\Omega(\cdot)$ and $\boldsymbol{0}$s are zero matrices with proper sizes. We can see that the problem (4) is a specific case of (15), and thus this proposition holds. □

Proposition 2. *With the weights, $\boldsymbol{W}^k$s and $\boldsymbol{\mathcal{W}}_{\boldsymbol{\mathcal{N}}}$, updated via concave functions, the result can be iteratively improved, and the (local) optimality of Algorithm 1 is guaranteed.*

Proof. Although our algorithm might involves two kinds of weight, *i.e.* $\boldsymbol{\mathcal{W}}_{\boldsymbol{\mathcal{N}}}$ for the sparse noise term and $\boldsymbol{W}^k$ for the GTV related term, they are separable. That means we only need to analyze the case $\|\boldsymbol{W} \odot \boldsymbol{A}\|_p$ here, where p can be either 1 or $2, 1$. Similar to [1], our thought of iteratively reweighted scheme falls into the general class of Majorization-Minimization framework, *i.e.*:

$$\underset{\boldsymbol{v}}{\text{argmin}}\, g(\boldsymbol{v}) \quad \text{s.t.} \quad \boldsymbol{v} \in \mathfrak{C} \tag{17}$$

where $\mathfrak{C}$ is a convex set. As aforementioned, we expect to preserve the sharp structure information meanwhile suppress the noise effect. To this end, the function $g(\cdot)$ should be concave, which yields the local linearization to achieve the minimizing goal. Consequently, we have:

$$\begin{aligned} \boldsymbol{v}^{(t+1)} &= \underset{\boldsymbol{v}\in\mathfrak{C}}{\text{argmin}}\, g(\boldsymbol{v}^{(t)}) + \langle \nabla g(\boldsymbol{v}^{(t)}), \boldsymbol{v} - \boldsymbol{v}^{(t)} \rangle \\ &= \underset{\boldsymbol{v}\in\mathfrak{C}}{\text{argmin}} \langle \nabla g(\boldsymbol{v}^{(t)}), \boldsymbol{v} \rangle. \end{aligned} \tag{18}$$

Figure 1: Benchmark images used in the experiments.

This clearly indicates $\nabla g(\boldsymbol{v}^{(t)})$ can perform as the updated weight (several special cases will be introduced in experiments), and thus recognizes our reweighting strategy. We can see that, as the outer loop iterates, the algorithm will be converged at least at a local optimum. □

4. Experiments

In this section, we evaluate the efficacy of our method on several classic visual data recovery applications, *i.e.* image/video completion and denoising. Two well-known metrics, PSNR and SSIM, are employed to quantitatively measure the quality of recovery, while the time to reflect the computational cost. All the experiments are conducted on a PC running Windows 7 64bit operating system with Intel Core i7 3.4 GHz CPU and 8.0 GB RAM. All of the involved algorithms are implemented in Matlab, which assures the fairness of the time cost comparison. Please notice that the efficiency of Algorithm 1 can be further improved by gradually increasing μ after each iteration with a relatively small initialization, *e.g.* $\mu^{(0)} = 1.25$, $\mu^{(t+1)} = \rho\mu^{(t)}, \rho > 1$, instead of using a constant μ. For the experiments, we use this strategy to accelerate the procedure.

The first experiment is carried out to reveal the superior performance of our GTV model over the state-of-the-art methods, including STDC [3] and HaLRTC[2] [9], on the task of color image completion. Figure 1 displays the benchmark images: one image with global structure *Facade*, one poor textural *Peppers*, and other eight of more complex and richer textures. The incomplete images and their corresponding supports $\boldsymbol{\Omega}$ are generated by randomly throwing away a fraction $f \in \{0.3, 0.5, 0.7\}$ of pixels. As our model in Eq. (4) has two options for both $\Psi(\boldsymbol{\mathcal{N}})$ and p, and various ways to update weights, instead of evaluating every possible combination, we only test a specific case of it in this experiment. The competitors employ the ℓ^2 noise penalty, to be fair, our method adopts the same. As for the type of GTV, the isotropic $p := 2, 1$ is selected for our method. The updating of $\boldsymbol{W}^{k(h+1)}$ for the $(h+1)^{th}$ out-

[2]Both the codes of STDC and HaLRTC are downloaded from the authors' websites, the parameters of which are all set as suggested by the authors to obtain their best possible results.

Table 1: Performance comparison in terms of PSNR(dB)/SSIM/Time(s).

Method	*Monarch*$_{(768\text{x}512\text{x}3)}$	*Mandrill*$_{(512\text{x}512\text{x}3)}$	*Frymire*$_{(1118\text{x}1105\text{x}3)}$	*Lena*$_{(512\text{x}512\text{x}3)}$	*Barbara*$_{(256\text{x}256\text{x}3)}$
STDC$_{0.3}$	30.49/.9776/92.02	25.00/.8996/51.32	20.02/.7431/427.6	32.49/.9863/52.12	30.54/.9368/10.11
HaLRTC$_{0.3}$	32.60/.9873/332.9	26.06/**.9194**/142.8	20.87/.7856/1083.	34.22/.9906/177.0	31.51/.9465/29.94
ℓ^2I-GTV$_{0.3}$	**34.95/.9932/51.77**	**26.34**/.9169/**37.12**	**22.66/.8911/216.2**	**35.17/.9916/38.02**	**32.09/.9508/8.85**
STDC$_{0.5}$	28.82/.9706/94.77	22.50/.8259/52.82	17.37/.6832/428.1	30.92/.9800/54.41	28.49/.8997/9.03
HaLRTC$_{0.5}$	27.86/.9664/351.3	22.82/.8293/155.0	17.81/.7031/1418.	30.08/.9773/169.0	27.55/.8853/31.64
ℓ^2I-GTV$_{0.5}$	**31.41/.9867/56.48**	**23.70/.8465/34.81**	**19.56/.8418/223.7**	**32.55/.9854/33.33**	**29.38/.9198/7.48**
STDC$_{0.7}$	27.21/.9610/97.41	19.99/.7238/52.80	15.03/.6127/438.1	29.51/.9726/52.06	26.46/.8550/9.87
HaLRTC$_{0.7}$	23.49/.9238/356.3	20.30/.7012/190.6	15.21/.5950/1401.	25.94/.9494/193.5	23.55/.7791/37.01
ℓ^2I-GTV$_{0.7}$	**27.61/.9719/53.71**	**21.63/.7463/34.09**	**16.96/.7578/219.7**	**29.68/.9745/33.39**	**26.54/.8704/7.36**

Method	*Peppers*$_{(512\text{x}512\text{x}3)}$	*Sail*$_{(768\text{x}512\text{x}3)}$	*Facade*$_{(256\text{x}256\text{x}3)}$	*Serrano*$_{(629\text{x}794\text{x}3)}$	*Tulips*$_{(768\text{x}512\text{x}3)}$
STDC$_{0.3}$	34.41/.9904/55.46	28.30/.9310/95.44	32.86/.9552/10.40	26.83/.9675/128.1	30.86/.9268/96.38
HaLRTC$_{0.3}$	**37.31/.9950**/180.9	29.96/.9513/348.3	**34.88/.9711**/27.83	27.36/.9723/343.1	34.36/.9656/339.6
ℓ^2I-GTV$_{0.3}$	35.10/.9942/**35.58**	**31.29/.9620/55.07**	29.64/.9211/**8.58**	**30.06/.9873/117.0**	**35.70/.9720/53.70**
STDC$_{0.5}$	**33.07**/.9874/53.08	26.31/.8932/98.87	30.59/.9252/10.10	24.90/.9533/122.2	29.40/.9102/97.52
HaLRTC$_{0.5}$	32.35/.9849/198.9	25.91/.8846/356.8	**31.28/.9364**/31.36	23.89/.9437/362.6	29.12/.9109/342.5
ℓ^2I-GTV$_{0.5}$	31.36/**.9882/33.43**	**28.22/.9269/56.21**	26.48/.8472/**6.97**	**26.78/.9763/113.9**	**32.27/.9532/55.25**
STDC$_{0.7}$	**31.88/.9840**/53.94	24.18/.8382/98.12	27.83/.8731/10.47	22.90/.9322/127.6	28.12/.8939/99.77
HaLRTC$_{0.7}$	27.31/.9581/220.4	22.51/.7772/363.9	**27.90/.8740**/34.97	20.51/.8886/390.1	24.15/.8047/360.6
ℓ^2I-GTV$_{0.7}$	27.99/.9761/**36.45**	**25.42/.8675/56.76**	23.42/.7172/**8.05**	**23.61/.9551/112.8**	**28.61/.9179/56.44**

er iteration is done via $\boldsymbol{W}^{k(h+1)} := c\sqrt{\exp\left(-|\boldsymbol{Q}^{k(h)}|\right)}$, where c is a positive constant (here we simply use $c = 1$). This reweighting strategy satisfies the conditions mentioned in Proposition 2. Besides, we fix the parameters as $\boldsymbol{\alpha} = [1; 0; 0]$ (3-order, only consider the mode-1 unfolding), $\boldsymbol{\beta}^k = [0.25; 0.25; 0.25; 0.25]^T$ (4-direction, say 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, $\frac{3\pi}{4}$), and $\lambda = 20$ for all cases. In addition, to save the computational load, the maximal outer iteration number is set to 2 (latter we will see that the weights can be updated sufficiently well through 2 iterations). The model (4) with this setting is denoted as ℓ^2I-GTV.

Table 1 reports the average results of STDC, HaLRTC and ℓ^2I-GTV over 10 independent trials. From the time comparison, it is easy to see that our method is much more efficient than the others. Specifically, STDC and HaLRTC cost about 1.7 times and 5 times as much as ℓ^2I-GTV does, respectively. In terms of PSNR and SSIM, our method significantly outperforms the others except for the images *Peppers* and *Facade*, please see the Frymire case shown in Fig. 2. The reason for the exceptions may be that these two images are of either regular (*Facade*) or poor texture (*Peppers*), which fit the low-rank prior well. Even though, the quality of our recovery is still competitive or even better, please see the Peppers case shown in Fig. 3 for example.

As our model might involve two kinds of weight, $\boldsymbol{W}^k$s and $\boldsymbol{\mathcal{W}}_{\boldsymbol{\mathcal{N}}}$, to validate the benefit of the reweighting strategy (or sparsity enhancing), we thus employ the one with $\Psi(\boldsymbol{\mathcal{N}}) := \|\boldsymbol{\mathcal{W}}_{\boldsymbol{\mathcal{N}}} \odot \boldsymbol{\mathcal{N}}\|_1$ and anisotropic GTV. The task

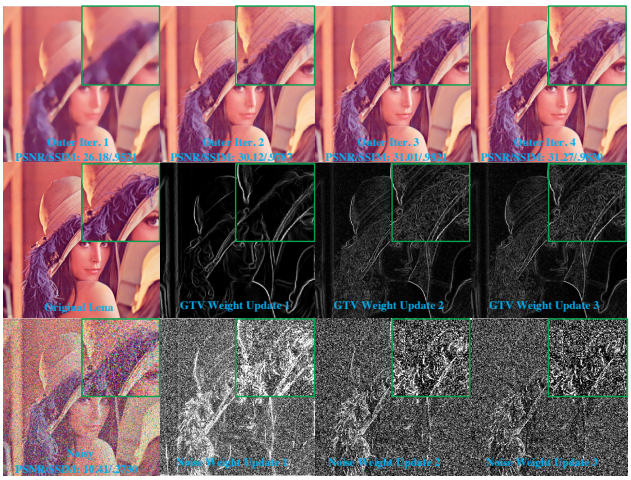

Figure 4: The benefit of the reweighting strategy. The noisy image is synthesized by introducing 35% Salt&Pepper noise into the original. Lighter pixels in weight maps indicate higher probabilities of being high-frequency signals in the second row or being noises in the third row (lower weights), while darker ones mean lower probabilities (higher weights).

is to restore images from noisy observations (polluted by Salt & Pepper noise), which is similar to the completion. But the difference-the unknown support of clean entries-makes it more difficult. The way to update $\boldsymbol{\mathcal{W}}_{\boldsymbol{\mathcal{N}}}{}^{(h+1)}$ u-

Figure 2: Visual comparison on Frymire. **Top row** shows the results with 30% information missed. **Middle** and **Bottom** correspond to those with 50% and 70% elements missed, respectively. **From Left to Right:** input frames, recoverd results by STDC, Ha LRTC and GTV, respectively. Details can be better observed in zoomed-in patches.

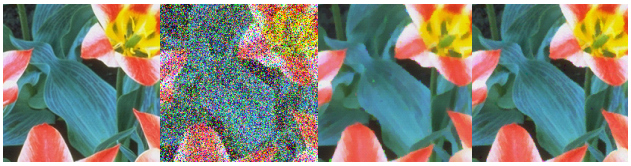

Figure 5: An example of image denoisng. **Left:** Original image. **Mid-Left:** Polluted image by 40% Salt & Pepper noise (PSNR/SSIM: 8.71dB/0.1488). **Rest:** Recovered results by traditional TV (27.86dB/0.9144) and ℓ^1A-GTV (29.08dB/0.9234), respectively.

tilizes $\frac{1}{|\mathcal{N}^{(h)}|+\epsilon}$, where the division is element-wise and ϵ is a positive constant to avoid zero-valued denominators and provide stability. The updating of $\boldsymbol{W}^{k(h+1)}$ and, the settings of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}^k$ follow the previous experiment. Due to the importance of λ to the restoration, we design an adaptive updating scheme for the sake of robustness. That is, at the 1^{st} outer iteration, a relative small λ (0.2 for the rest experiments) is first set, then iteratively refine it to be inversely

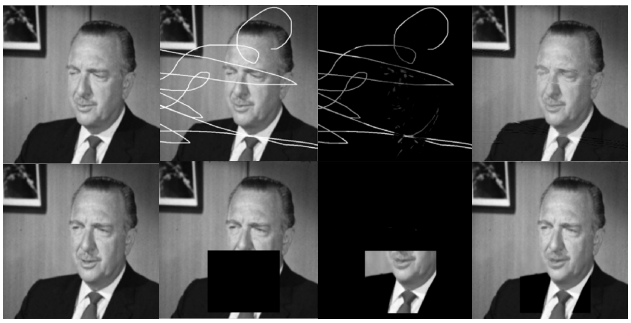

Figure 6: Two examples of video inpainting. **From Left to Right:** Original frames, corrupted frames, detected corruptions and recovered results by ℓ^1A-GTV, respectively.

proportional to $\frac{\|\mathcal{N}^{(h)}\|_1}{\prod_i D_i}$. We denote this setting of our model as ℓ^1A-GTV. It can be seen from the results shown in Fig. 4 that, as the outer loop iterates, both the visual quality of recovery and the accuracy of weights increase. Please note that, the 2^{nd} outer iteration significantly improves the 1^{st}

Figure 3: Visual comparison on Peppers. **Top row** shows the results with 30% information missed. **Middle** and **Bottom** correspond to those with 50% and 70% elements missed, respectively. **From Left to Right:** input frames, recoverd results by STDC, Ha LRTC and GTV, respectively. Details can be better observed in zoomed-in patches.

with very promising result. In other words, the outer loop converges rapidly. In addition, the inner loop of all the experiments can be converged within 40-50 iterations.

Figure 5 shows an example on image denoising to demonstrate the superior performance of GTV over the traditional TV [16, 2]. The middle-right picture in Fig. 5 is the best possible result of the traditional TV by tuning $\lambda \in \{0.1, 0.2, ..., 1.0\}$, while the right is automatically obtained by ℓ^1A-GTV. The recovered details are the clear and convincing evidences on the advance of GTV.

To further show the ability of ℓ^1A-GTV, we apply it to video inpainting. Different to the previous setting of ℓ^1A-GTV, the parameter $\boldsymbol{\alpha}$ for this task adopts $[0; 1; 1]$ to reveal the power of temporal information. Two examples are shown in Fig. 6, as can be viewed, our method can successfully detect and repair the corruption in the video sequence. For the upper case in Fig. 6, the original frame is scratched arbitrarily, the PSNR/SSIM of the corrupted frame is 18.66dB/.8442. Our inpainting for this case gives 34.91dB/.9168 high-quality recovery. While a larger area of the frame in the lower row is perturbed (13.99dB/.7076), the repaired result of which is with 29.36dB/.9026.

5. Conclusion

Visual data recovery is an important, yet highly ill-posed problem. The piece-wise smooth nature of visual data makes the problem well-posed. This paper has proposed a novel generalized tensor total variation norm (GTV) definition to exploit the underlying structure of visual data. We have formulated a class of GTV minimization problems in a unified optimization framework, and designed an effective algorithm to seek the optimal solution with the theoretical guarantee. The experimental results on visual data completion, denoising and inpainting have demonstrated the clear advantages of our method over the state-of-the-art alternatives. It is positive that our proposed GTV can be widely applied to many other visual data restoration tasks, such as deblurring, colorization and super resolution.

References

[1] E. Candès, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008. 5

[2] S. Chan, R. Khoshabeh, K. Gibson, P. Gill, and T. Nguyen. An augmented lagrangian method for total variation video restoration. *IEEE Transactions on Image Processing*, 20(11):3097–3111, 2011. 2, 8

[3] Y. Chen, C. Hsu, and H. Liao. Simultaneous tensor decomposition and completion using factor priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):577–591, 2014. 1, 5

[4] L. Condat. A direct algorithm for 1-d total variation denoising. *IEEE Signal Processing Letters*, 20(11):1054–1057, 2013. 2

[5] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with applications to image denoising. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2862–2869, 2014. 1

[6] X. Guo, X. Cao, and Y. Ma. Robust sepration of reflection from multiple images. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2187–2194, 2014. 2

[7] M. Hong and Z. Luo. On the linear convergence of the alternating direction method of multipliers. *arXiv preprint arXiv:1208.3922*, 2013. 3, 5

[8] Z. Lin, R. Liu, and Z. Su. Linearized alternating direction method with adaptive penalty for low rank representation. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 612–620, 2011. 3

[9] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2013. 1, 5

[10] S. Lu, X. Ren, and F. Liu. Depth enhancement via low-rank matrix completion. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3390–3397, 2014. 1

[11] A. Narita, K. Hayashi, R. Tomioka, and H. Kashima. Tensor factorization using auxiliary information. In *Proceedings of Machine Learning and Knowlegde Discovery in Databases*, pages 501–516, 2011. 1

[12] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phisica D: Nonlinear Phenomena*, 60(1):259–268, 1992. 2

[13] X. Shu, F. Porikli, and N. Ahuja. Robust orthonormal subspace learning: Efficient recovery of corrupted low-rank matrices. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3874–3881, 2014. 1

[14] H. Wang, F. Nie, and H. Huang. Low-rank tensor completion with spatio-temporal consistency. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 2846–2852, 2014. 1

[15] S. Wang and Z. Zhang. Colorization by matrix completion. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 1169–1174, 2012. 1

[16] S. Yang, J. Wang, W. Fan, X. Zhang, P. Wonka, and J. Ye. An efficient admm algorithm for multimensional anisotropic toal variation regularization problems. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 641–649, 2013. 2, 8

[17] X. Zhang, Z. Zhou, D. Wang, and Y. Ma. Hybrid singular value thresholding for tensor completion. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 1362–1368, 2014. 1

[18] Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. TILT: Transform invariant low-rank textures. *Internation Journal of Computer Vision*, 99(1):1–24, 2012. 1