Image Reconstruction using Superpixel Clustering and Tensor Completion

Maame G. Asante-Mensah*†, Anh Huy Phan†, *Member, IEEE*, Salman Ahmadi-Asl†, Zaher Al Aghbari‡
and Andrzej Cichocki†, *Fellow, IEEE*

Abstract

This paper presents a pixel selection method for compact image representation based on superpixel segmentation and tensor completion. Our method divides the image into several regions that capture important textures or semantics and selects a representative pixel from each region to store. We experiment with different criteria for choosing the representative pixel and find that the centroid pixel performs the best. We also propose two smooth tensor completion algorithms that can effectively reconstruct different types of images from the selected pixels. Our experiments show that our superpixel-based method achieves better results than uniform sampling for various missing ratios.

Index Terms

Superpixel, Tensor Completion, Uniform sampling, Nuclear norm minimization.

I. Introduction

Dealing with incomplete data tensors is inevitable in real-world applications due to sensor malfunctioning, inaccurate data acquisition, communication problems or inappropriate handling. However, data elements can also be manually removed to optimise space requirements or to remove unwanted outliers. Estimating the unknown elements of an incomplete data tensor is known as *tensor completion* and has played important roles in many machine learning problems, e.g. image/video completion [1] and recommender systems [2]. Due to the importance of tensor completion in the above-mentioned applications, several efficient algorithms have been developed during the past few decades to solve it. Indeed, tensor completion has its basis from matrix completion [3]. Similar to the matrix completion where the main assumption is low-rank property of the underlying data matrix, for the tensor case, it is also assumed that the data tensor has low-rank structure. This assumption plays a key role in the formulation of the optimization problem and also deriving theoretical results. It should be noted that the notion of rank for tensors is not unique as it is for matrices, different types of tensor ranks have been defined [4]. Minimizing the matrix rank is an NP hard problem and it has been proved that the nuclear norm is the convex envelope of the matrix rank which can be used to efficiently estimate the matrix rank [5]. Replacing the matrix rank function with the nuclear norm, converts a non-convex optimization problem into a convex one which is more favorable. Following the matrix case, the nuclear norm of the unfolding matrices has been used to efficiently recover different types of tensor ranks, see [1] for a comprehensive review on this topic.

†Skolkovo Institute of Science and Technology (SKOLTECH), CDISE, Moscow, *Corresponding author E-mail: gyamfua.asantemensah@skoltech.ru.
‡Department of Computer Science, University of Sharjah, Sharjah, 27272, United Arab Emirates.

This paper is inspired by the idea of using tensor completion technology to compress and transmit images [6]. The idea is to sample a subset of pixels from an image at the source and send them to the destination through a network. Then, at the destination, the tensor completion algorithms are applied to the incomplete image to recover it. This can save memory and speed up data transmission. Most existing papers use uniform sampling to select pixels, but we wonder if other heuristic approaches can perform better. To the best of our knowledge, this question has not been investigated before. We propose to cluster a given image into several partitions that have similar characteristics and then select pixels from each partition. As an efficient clustering method, superpixel algorithms have been used in several applications such as image segmentation [7], [8], [9], [10] and object detection [11]. Among several superpixel methods, Simple Linear Iterative Clustering (SLIC) algorithm [12] is known for its high performance and fast execution time. Due to these key issues, we use it in our work as a preprocessing stage to find partitions with similar texture/semantic. Then, we select pixels from each superpixel separately. Since each superpixel has homogeneous features and share similar pixel information, the preprocessing stage can hep us to avoid sampling many redundant and similar pixels. Of course, there are several ways to select pixels from each superpixel such as the pixel located in the center or those are on the border. It is also possible to apply the clustering algorithm repeatedly to each superpixel to further explore important pixels. We have extensively investigated the performance of such different sampling strategies. Our simulation results confirmed that the pixel selection based on superpixel preprocessing provides better results than the uniform sampling for a variety of images. Besides, the superpixel preprocessing method with center pixel selection achieved the best results compared with selecting other pixels in the superpixels.

For recovering the sampled/incomplete image, we also propose the Smooth Tensor nuclear norm (STNN) and Smooth Matrix Nuclear Norm (SMNN) algorithms for image completion. In particular, the smoothing processing enables the completion algorithms to provide better results and this is shown experimentally in our simulation results. The important pixel selection followed by applying efficient tensor completion algorithms totally improves the traditional uniform sampling approach.

We summarize our main contributions as follows:

- We sample and capture important pixels in images using the superpixel technique.
- We develop efficient tensor completion algorithms with smoothing and filtering methods to enhance their performance.
- We conduct extensive simulations on various images with missing patterns, including *random* and *structured* ones.

The rest of this paper is organized as follows: Section II introduces the notations and concepts that we use in our approach. Section III explains the superpixel technique and how it extracts important content or semantics from the images. Section IV studies the importance of pixel selection and how to sample important pixels. Section V develops tensor completion algorithms with an efficient smoothing technique to improve their performance. Section VI presents extensive experiments to demonstrate the applicability and feasibility of our methods. Section VII concludes the paper.

II. NOTATIONS AND DEFINITIONS

Basic notations used in this work are taken from [4]. We represent scalars by the lowercase letters. A vector is given by a boldface lower case letter, e.g. $\mathbf{a}$. A matrix is represented by boldface capital letter, e.g. $\mathbf{A}$ and a higher order tensors are also

denoted by bold underlined capital letter, e.g. $\underline{\mathbf{A}}$. For an Nth-order tensor $\underline{\mathbf{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ its $(i_1, i_2, \ldots, i_N)$th element is denoted by $x_{i_1,i_2,\ldots,i_N}$ and is represented as $\underline{\mathbf{A}}(i_1, i_2, \ldots, i_N)$.

The n-mode *matricization* of a tensor $\underline{\mathbf{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ which is also called mode-n unfolding of a tensor [13] is shown by $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times I_1 \cdots I_{n-1} I_{n+1} \cdots I_N}$. *Tensorization* or matrix folding is the process of converting a low-order tensor to a higher-order tensor. When we fix all indices except two of a tensor, a sub-tensor is generated and it is called a slice. For example, for a tensor $\underline{\mathbf{A}} \in \mathbb{R}^{I \times J \times K}$, the slices $\underline{\mathbf{A}}(:,:,k)$, $i = 1, 2, \ldots, I_3$, are called frontal slices and is denoted as $\mathbf{A}_{(k)}$, the tube in a tensor is denoted as $\underline{\mathbf{A}}(i, j, :)$. The inner product of two tensors $\underline{\mathbf{A}}$, $\underline{\mathbf{B}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined as $\langle \underline{\mathbf{A}}, \underline{\mathbf{B}} \rangle = \sum_{i_1} \sum_{i_2} \ldots \sum_{i_N} a_{i_1,\ldots,i_N} b_{i_1,\ldots,i_N}$ and the Frobenius norm of a tensor is given as $\|\underline{\mathbf{A}}\|_F = \sqrt{\langle \underline{\mathbf{A}}, \underline{\mathbf{A}} \rangle}$.

A. Low-Rank Tensor Completion

Let an incomplete data tensor $\underline{\mathbf{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, be given and assume the indices of its observed elements are arranged in the indexing binary tensor $\underline{\mathbf{\Omega}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. The complement of the indexing set $\underline{\mathbf{\Omega}}$ is represented as $\underline{\mathbf{\Omega}}^{\perp}(i_1, i_2, \ldots, i_N)$. The projection operator $\underline{\mathbf{\Omega}}$ over the data tensor $\underline{\mathbf{A}}$ is defined as follows:

$$\mathcal{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{A}}) = \begin{cases} a_{i_1,i_2,\ldots,i_N} & (i_1, i_2, \ldots, i_N) \in \underline{\mathbf{\Omega}}. \\ 0 & \text{for missing entries.} \end{cases}$$

The task of low rank tensor completion can be formulated as the following optimization problem

$$\arg\min_{\underline{\mathbf{B}}} \ \left\| \mathcal{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{B}}) - \mathcal{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{A}}) \right\|_F^2 , \tag{1}$$

where $\underline{\mathbf{B}}$ is the estimated data tensor. To make the problem (1) well-posed we need to impose constraints on the data tensor $\underline{\mathbf{B}}$, e.g. low-rank property. Depending on the tensor rank notion defined such as Tucker rank, Tensor Tran rank, Tubal rank etc, the minimization problem (1) is solved over the space of tensors with at most the predefined tensor rank. If estimating the tensor rank is difficult, then the minimization problem (1) can be converted to the tensor rank minimization problem formulated as follows

$$\arg\min_{\underline{\mathbf{B}}} \ \text{rank}(\underline{\mathbf{B}}) + \left\| \mathcal{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{B}}) - \mathcal{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{A}}) \right\|_F^2 , \tag{2}$$

where by solving problem (2), we can reconstruct the data tensor and also estimate the tensor rank. Unfortunately, the tensor rank minimization is an NP hard problem and a surrogate or a relaxation of it should be considered (to be discussed in Section V).

III. SUPERPIXEL CLUSTERING

A group of pixels that share similar features such as pixel intensity, texture and color are referred to as superpixels. Superpixels can be found in many applications of computer vision and machine learning tasks. As an efficient superpixel method, SLIC (Simple Linear Iterative Clustering) algorithm was introduced in [12] and due to its superior performance [14], [15] was extensively used in many applications such as image segmentation [7], [8], [9], [10], object detection [11], anomaly detection [16], [17] and image reconstruction [18], [19], [20]. To efficiently generate superpixels, the SLIC algorithm employs a k-means

clustering approach. It basically converts an RGB color space into the CIELAB color space as $[l, a, b, x, y]^T$ where $[l, a, b]^T$ is the pixel color vector in CIELAB color space and $[x, y]^T$ is the pixel position. The spatial distances are normalized in order to use the Euclidean distance in the 5D space. To cluster pixels in 5D space, a new distance measure that takes super-pixel size into account is introduced. This method takes as input a desired number of approximately equally sized superpixels K. At first, K superpixel cluster centers $\mathbf{c}_k$, $k = 1, 2, \ldots, K$ are chosen at regular grid intervals S. Since the spatial vastness of any superpixel is approximately S^2, we assume that pixels associated with this cluster center are located within a $2S \times 2S$ area on the xy plane surrounding the superpixel center. D_s is the normalized distance measure that will be used in 5D space and defined as follows

$$
\begin{aligned}
D_s &= d_{lab} + (m/S) * d_{xy}, \\
d_{lab} &= \sqrt{(l_k - l_i) + (a_k - a_i) + (b_k - b_i)}, \\
d_{xy} &= \sqrt{((x_k - x_i)^2 + (y_k - y_i)^2)},
\end{aligned}
\tag{3}
$$

where i represents the value to be clustered. The sum of the lab distance d_{lab} and the xy plane distance d_{xy} normalized by the grid interval S. Besides, the distance measure D_s includes a variable m that allows us to control the compactness of a superpixel. The greater the value of m, the cluster becomes more compact. This value can range between 1 and 20. The SLIC algorithm is summarized in the Appendix (Algorithm 5). See Figure 1 for a graphical illustration on the SLIC algorithm for $K = 50, 100, 200$ superpixels.

The use of superpixels for various computer vision and image processing tasks acts as a preprocessing step to reduce the complexity of subsequent processing. They are also used to capture redundancy in an image [15]. A superpixel clustering method was employed in [21] as an initial step to segmentation, texture learning and patch matching for image reconstruction. Indeed, the superpixel methods as clustering techniques allow for sampling important pixels used in the reconstruction of the images. The reconstructed images can then be used to perform other tasks achieving results comparable to the original image. The clustering techniques using superpixel have been implemented in many neural networks and autoencoders models. These methods have proven to be very effective with promising results. The unsupervised learning models are very useful for feature extraction. A Dual Graph Autoencoder (DGAE), proposed by Zhang et al. [22] constructs the superpixel-based similarity graph using entropy rate superpixel which captures the spatial information in the image and generate a band-based similarity graph that can be used to characterize the geometric structures of hyperspectral images. The dual graph convolution, allows more discriminative feature representations to be learned from the hidden layers that aids in the generation of a clustering map. Superpixels has also seen implementation in medical data segmentation [23], [24], [25], [26]. The implementation by Bechar et al [24] uses a semi-supervised model for optic cup and disc segmentation to calculate the cup to disc ratio value. Here, the SLIC superpixel was adopted for generating some labels used in the retina segmentation. However, the drawback to most of these sophisticated models is hyperparameter fine-tuning and also the computational load needed to achieve the desired result. Superpixel based image representation in [27] acquires mid-level information in order to improve the object recognition accuracy. Even though the main work focuses on the image recognition task, superpixel clustering has been performed specifically for the feature extraction step.

Fig. 1: Superpixel clustering using SLIC with $k = 50, 100$, and 200 superpixels.

Superpixel clustering

A centroid for this superpixel cluster

Fig. 2: Illustration of centroid superpixel sampling.

Original   70% Uniform sampling   70% Superpixel sampling

Fig. 3: Illustration of uniform sampling vs superpixel sampling on kodim20 image. Uniform sampling can not find the important pixels and treat them accordingly while the superpixel approach chooses the most promising pixels.

40.488   30.553   36.590   33.679

Original Image   Centroids   Boundary   Multi-stage sampling   Random

Fig. 4: PSNR comparison of kodim03 image using different superpixels sampling methods with approximately selecting 50% of the data. The experiment show that centroid pixel selection provides better results.

IV. PROPOSED PIXEL SAMPLING METHOD

Signal processing researchers have found that a data tensor with a low-rank structure can be efficiently recovered from a subset of its components [1]. This finding has led to many tensor completion algorithms for various types of data, such as

images and videos. The missing components can have either random or structured patterns. Random patterns remove pixels of images/videos randomly, while structured patterns remove sequential fibers, slices, or parts of the data. For example, [28] uses a structured pattern to remove slices of videos and recovers them using a Hankelization approach. Structured patterns are more challenging than random ones, because they lose information about a whole region of the data. However, the question of how to select the best subset of pixels for optimal image recovery performance has not been well studied. This paper aims to investigate this question. Most of the existing papers use uniform sampling to remove pixels of images or videos. For example, in [6], it is proposed to sample some pixels of a given image uniformly and remove the rest to reduce the memory requirement and to transfer it faster in the network. The sampled pixels are recovered in the destination through the tensor completion process. Contrary to the simple pixel selection using random sampling, we propose to first cluster the image to some partitions or so-called superpixels which share common characteristics. Then, we select the pixels from each of the clusters. Since the pixels in each cluster more likely have similar features, the clustering as a preprocessing stage can help to avoid selecting redundant pixels as is done in random sampling. In this sense, it is a kind of smart pixel sampling as we try to select pixels in a heuristic and an intelligent way.

It is worth pointing out that to select pixels in each cluster, there are several possibilities. For example, We can select a pixel located in the center of the cluster or on its boundary/border. Depending on a way used to select a pixel from each superpixel, we define the following categories:

- **Centriod superpixel approach** The centroid superpixel refers to the case when we select a center of each superpixel to be the selected pixel.
- **Boundary superpixel approach** This approach selects a pixel from a border/boundry of the superpixel and use it for the sampling procedure.
- **Multi-stage superpixel approach.** It is also possible to cluster each spuerpixel again to further explore important pixels and we call it multi-stage superpixel approach.

Figure 2, shows the superpixel clustering applied to a given image followed by selecting the center of each superpixel. Also, Figure 3, demonstrates what looks like an image after sampling 30% of pixels using uniform sampling and superpixel sampling with center selection. We extensively investigated the difference between the performance of different sampling methods for the superpixel approach and compared it with the uniform sampling. Our simulation results show that in most of our experiments on a variety of images, the center of superpixels provides quite promising results. For example, in Figure 4, the results of applying the superpixel clustering (with center, boundary and multi-stage selection) and the uniform sampling to an image with 50% pixel sampling have been reported. As we earlier mentioned, the superpixel with center pixel selection achieved the best performance. It is interesting to note that for 60% pixel sampling, the difference between uniform sampling and superpixel clustering with center selection was significant as seen in Figure 5. After sampling important pixels, we have a compact variant of the image which can be transmitted and it can be recovered in the destination using tensor completion algorithms (see Section V).

Fig. 5: The results show that centroid pixel selection provides better results than the uniform sampling. For Kodim23 image, we have used 237905 clusters which leads to sampling 60% of all pixels.

V. TENSOR COMPLETION WITH SMOOTHING/FILTERING

As discussed in Section IV, in the first stage of our methodology, we sample a part of images to have a compact representation of the image. This strategy was used in [6] to store only a part of pixels. In the second stage, the image with only observed pixels should be recovered based on the completion algorithms. Here, the performance of such completion algorithms is crucial for the better image recovery. In this section, we use the nuclear norm minimization (both matrix and tensor scenarios, see the Appendix) for the completion task. Nonetheless, we apply the smoothing strategy to enhance the performance of the algorithms. In the simulation section, we will show the importance of smoothing strategy for a better image recovery. The matrix completion and tensor completion formulations are presented in Subsections V-A and V-B, respectively.

A. Tensor completion based on matrix nuclear norm regularization

A significant advantage of the low-rank matrix approximation is that the vital information in a matrix, given in terms of degree of freedom, is substantially less than the total number of entries. As a result, even if the number of observed entries is small, there is still a decent chance of recovering the entire matrix [29]. This advantage has been shown in computer vision tasks [30], [31] for estimating missing values in images. In this section, we describe the completion task. The model of rank minimization based matrix completion is formulated as:

$$\min_{\mathbf{X}} \quad \mathrm{rank}\left(\mathbf{X}\right)$$
$$s.t. \quad \mathbf{X}_\Omega = \mathbf{Y}_\Omega$$

$$(4)$$

where $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$ and elements of $\mathbf{X}$ are determined such that the rank of the matrix $\mathbf{X}$ is as small as possible. However, finding the rank of a matrix is non-convex therefore the optimization problem in (4) becomes a non-convex problem. One common approach to solve this is to use the matrix nuclear norm $\|.\|_*$ to approximate the rank. The nuclear norm has the

advantage of being the tightest convex envelop for the rank of matrices [32], [33], [30]. Matrix nuclear norm has widely been used and has been successful in reconstruction and completion tasks [34]. Adopting nuclear norm the problem in (4) becomes

$$
\begin{aligned}
&\min_{\mathbf{X}} \quad \|\mathbf{X}\|_* \\
&s.t. \ \ \mathbf{X}_\Omega = \mathbf{Y}_\Omega,
\end{aligned}
\tag{5}
$$

with $\|.\|_*$ representing the nuclear norm of a matrix which is the sum of the singular values of the matrix.

To use the matrix completion formulation (5), when we are dealing with data tensors, we can use the unfolding of the underlying data tensor and replaced it in (6) ($\mathbf{X}_{(1)} \leftarrow \underline{\mathbf{X}}$, $\mathbf{Y}_{(1)} \leftarrow \underline{\mathbf{Y}}$ and $\mathbf{\Omega}_{(1)} \leftarrow \underline{\mathbf{\Omega}}$). Therefore we obtain the following optimization problem

$$
\min_{\mathbf{X}_{(1)}} \ \lambda \left\|\mathbf{X}_{(1)}\right\|_* + \left\|\mathcal{P}_{\mathbf{\Omega}_{(1)}} \left(\mathbf{X}_{(1)} - \mathbf{Y}_{(1)}\right)\right\|_F^2, \ s.t. \ \ \mathbf{X}_{\mathbf{\Omega}_{(1)}} = \mathbf{Y}_{\mathbf{\Omega}_{(1)}}
\tag{6}
$$

where λ denotes the trade-off parameter. A constrained optimization problem can be formulated by introducing an auxiliary matrix $\mathbf{Z}_{(1)}$ with the same size as the matrix $\mathbf{X}_{(1)}$

$$
\begin{aligned}
&\min_{\mathbf{X}_{(1)},\mathbf{Z}_{(1)}} \ \ \lambda \left\|\mathbf{X}_{(1)}\right\|_* + \left\|\mathcal{P}_{\mathbf{\Omega}_{(1)}} \left(\mathbf{Z}_{(1)} - \mathbf{Y}_{(1)}\right)\right\|_F^2 \\
&\quad s.t. \quad\ \ \mathbf{X}_{(1)} = \mathbf{Z}_{(1)}.
\end{aligned}
\tag{7}
$$

For the brevity of presentation, we use the notation $\mathbf{X}_{(1)} = \mathbf{X}$, $\mathbf{Z}_{(1)} = \mathbf{Z}$, $\mathbf{Y}_{(1)} = \mathbf{Y}$. We solve the minimization problem (7) via the Alternating Direction Method of Multipliers (ADMM) algorithm [35] which has been shown to have fast convergence and good performance. To do so, the augmented Lagrangian function for the constrained optimization problem (7) is first written as

$$
\mathcal{L}\left(\mathbf{X}, \mathbf{Z}, \mathbf{T}\right) = \lambda \|\mathbf{X}\|_* + \left\|\mathcal{P}_{\mathbf{\Omega}_{(1)}} \left(\mathbf{Z} - \mathbf{Y}\right)\right\|_F^2 + \langle \mathbf{T}, \mathbf{X} - \mathbf{Z}\rangle + \frac{\mu}{2} \|\mathbf{X} - \mathbf{Z}\|_F^2,
\tag{8}
$$

where $\mathbf{T}$ is a matrix representing the Lagrangian multipliers and μ is a penalty parameter. In our simulation results the ADMM method worked properly for μ between 0.1 to 1.1. According to the ADMM method, we update the matrices $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$, iteratively by fixing two of them and updated the other as presented below.

Update of X: By minimizing the augmented Lagrangian function (8) w.r.t. $\mathbf{X}$, we have

$$
\mathbf{X}_{k+1} = \min_{\mathbf{X}} \ \mathcal{L}\left(\mathbf{X}, \mathbf{Z}_k, \mathbf{T}_k, \mu_k\right),
\tag{9}
$$

which can be further simplified to

$$
\mathbf{X}_{k+1} = \min_{\mathbf{X}} \ \left(\lambda \|\mathbf{X}\|_* + \frac{\mu_k}{2} \left\|\mathbf{X} - \mathbf{Z}_k + \frac{\mathbf{T}_k}{\mu_k}\right\|_F^2\right).
\tag{10}
$$

According to the paper [33], the above problem has the closed form solution given by:

$$
\mathbf{X}_{k+1} = \mathbf{D}_\beta \left(\mathbf{Z_k} - \frac{\mathbf{T_k}}{\mu_\mathbf{k}}\right),
\tag{11}
$$

where $\beta > 0$ is constant and $\mathbf{D}_\beta(.)$ is the matrix singular value thresholding operation defined as follows

$$\mathbf{D}_\beta(\mathbf{X}) = \mathbf{U} * \mathbf{D}_\beta(\mathbf{S}) * \mathbf{V}^T, \tag{12}$$

where $\underline{\mathbf{D}}_\beta(\mathbf{S}^{(i)}) = \mathrm{diag}(\max\{\sigma_t - \beta, 0\}_{1 \leq t \leq R})$, $i = 1, \ldots, I_3$, $\beta > 0$ is a constant and R is the rank of $\mathbf{S}$. Here, the $\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ SVD of the matrix $\mathbf{X}$.

Update of Z: By minimizing the augmented Lagrangian function w.r.t. $\mathbf{Z}$, we have

$$\mathbf{Z}_{k+1} = \min_{\mathbf{Z}} \ \mathcal{L}\left(\mathbf{X}_{k+1}, \mathbf{Z}, \mathbf{T}_k, \mu_k\right), \tag{13}$$

and can be simplified as

$$\mathbf{Z}_{k+1} = \min_{\mathbf{Z}_k^{(n)}} \ \|\mathcal{P}_{\boldsymbol{\Omega}}\left(\mathbf{Z} - \mathbf{Y}\right)\|_F^2 + \frac{\mu_k}{2}\left\|\mathbf{X}_{k+1} - \mathbf{Z} + \frac{\mathbf{T}_k}{\mu_k}\right\|_F^2 \tag{14}$$

The closed form solution for problem 14 is also solved through

$$\mathbf{Z}_{k+1} = \mathcal{P}_{\boldsymbol{\Omega}}^{\perp}\left(\mathbf{X}_{k+1} + \frac{\mathbf{T}_k}{\mu_k}\right) + \mathbf{Y} \tag{15}$$

Update of T: The solution for the Lagrangian multiplier matrix $\mathbf{T}$ is similarly converted to simpler optimization problem as follows

$$\mathbf{T}_{k+1} = \min_{\mathbf{T}} \ \mathcal{L}\left(\mathbf{X}_{k+1}, \mathbf{Z}_{k+1}, \mathbf{T}, \mu_k\right). \tag{16}$$

which has the close solution

$$\mathbf{T}_{k+1} = \mathbf{T}_k + \mu_k\left(\mathbf{X}_{k+1} - \mathbf{Z}_{k+1}\right). \tag{17}$$

Besides, we update the parameter μ in the following way

$$\mu_{k+1} = \min \ \left(\alpha\mu_k, \mu_{max}\right). \tag{18}$$

where $\alpha > 1$ is a predetermined constant used to iteratively increase the penalty and μ_{max} represents the upper bound for the penalty. This procedure is summarized in Algorithm 1. As we will discuss in Subsection V-C, to improve the quality of the image reconstruction process, we smooth the auxiliary matrix $\mathbf{Z}$ in Line of Algorithm 1 after its computation. This smoothing approach totally improves the results as will be shown in the simulation part.

B. Tensor completion using tensor nuclear norm (TNN) regularization

In this section, we discuss the tensor formulation of the completion problem based on the tensor Singular Decomposition (t-SVD) model. The motivation for this new formulation is comparing the performance of the matrix and the tensor variants in reconstructing the images. We have seen better results of the tensor case than the matrix case as will be discussed in the simulation section. Kilmer et al. [36] proposed the t-SVD as a new tensor decomposition, for detailed description of this tensor model, see the Appendix. Inspired by the results achieved by the nuclear norm minimization of matrices for recovering

---

Algorithm 1: Algorithm for Smooth Matrix Nuclear Norm (SMNN)

Input : An observed data tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the observation index tensor $\underline{\mathbf{\Omega}}$, and regularization parameter $\lambda > 0$, $\mathbf{X}_0 = \mathbf{T}_0$, $\mathbf{Z}_0 = \mathbf{X}_0$, $\mathcal{P}_{\mathbf{\Omega}}(\mathbf{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathbf{Y})$.

Output: Completed data tensor $\underline{\mathbf{X}}$

1 **Perform** Super-pixel extraction to generate compressed data
2 **Perform** mode-n unfolding of tensor $\underline{\mathbf{X}}$
3 **while** *A stopping criterion is not satisfied* **do**
4   Update $\mathbf{X}_{k+1}$ using solution from equation 11
5   Update $\mathbf{Z}_{k+1}$ using solution from 15
6   Perform Smoothing operation
7   Update $\mathbf{T}_{k+1}$ using equation in 17
8   Update μ_{k+1} using solution from equation 18
9 **end**
10 **Reshape** $\mathbf{X}$ into tensor $\underline{\mathbf{X}}$
11 **Compute** $\underline{\mathbf{X}} = \mathcal{P}_{\mathbf{\Omega}}(\underline{\mathbf{X}}) + \mathcal{P}_{\mathbf{\Omega}^{\perp}}(\underline{\mathbf{X}})$
12 **Return** $\underline{\mathbf{X}}$

---

data matrices with missing values, Zhang *et al.*[37] proposed the tubal nuclear norm minimization approach based on t-SVD, defined as the sum of nuclear norms of all frontal slices in the Fourier domain and proved to be convex envelope to the tensor tubal rank (See the Appendix for the details). More precisely, the model of rank minimization based tensor completion is formulated as follows

$$\min_{\underline{\mathbf{X}}} \quad \text{tubal rank}(\underline{\mathbf{X}}) + \left\| \mathcal{P}_{\Omega}(\underline{\mathbf{X}} - \underline{\mathbf{Y}}) \right\|_F^2. \tag{19}$$

Similar to the matrix case, minimizing the tubal tensor rank is NP hard because it includes the matrix case as a special case. The matrix trace norm was generalized to the tensor case based on the t-product in [38], [39], [40], [37]. We use the one introduced in [40], [39] which has been shown to provide superior results compared to the others and to be faster because of using only the information of the first slice in the Fourier domain. So, we consider the following minimization problem

$$\min_{\underline{\mathbf{X}}} \quad \lambda \left\| \underline{\mathbf{X}} \right\|_* + \frac{1}{2} \left\| \mathcal{P}_{\Omega}(\underline{\mathbf{X}} - \underline{\mathbf{Y}}) \right\|_F^2, \tag{20}$$

where $\|.\|_*$ is the tubal nuclear norm and λ denotes the trade-off parameter. Note that the truncated tubal nuclear norm [40] can also be used in the formulation (20). Similar tensor completion formulation is used in [41] but here we have used unitary transform matrices instead of discrete Fourier transform matrix that is used in the traditional tensor SVD and has shown to provide better results [42]. We also proposed to improve the image recovery by smoothing the results at each iteration (Line 5 in Algorithm 2). To use the ADMM algorithm, we need to introduce an auxiliary tensor $\underline{\mathbf{Z}}$ with same size as the tensor $\underline{\mathbf{X}}$:

$$\begin{aligned} \min_{\underline{\mathbf{X}}, \underline{\mathbf{Z}}} \quad & \lambda \left\| \underline{\mathbf{X}} \right\|_* + \frac{1}{2} \left\| \mathcal{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Z}} - \underline{\mathbf{Y}}) \right\|_F^2 \\ s.t. \quad & \underline{\mathbf{X}} = \underline{\mathbf{Z}}, \end{aligned} \tag{21}$$

Here again, the augmented Lagrangian function corresponding to the constrained optimization problem (21), is written as

$$\mathcal{L}(\underline{\mathbf{X}}, \underline{\mathbf{Z}}, \underline{\mathbf{T}}) = \lambda \left\| \underline{\mathbf{X}} \right\|_* + \frac{1}{2} \left\| \mathcal{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Z}} - \underline{\mathbf{Y}}) \right\|_F^2 + \langle \underline{\mathbf{T}}, \underline{\mathbf{X}} - \underline{\mathbf{Z}} \rangle + \frac{\mu}{2} \left\| \underline{\mathbf{X}} - \underline{\mathbf{Z}} \right\|_F^2, \tag{22}$$

where $\underline{\mathbf{T}}$ is a tensor representing the Lagrangian multipliers and μ is a penalty parameter. Similarly as done for the matrix case, the Lagrangian function (22), is minimized with respect to the tensors $\underline{\mathbf{X}}$, $\underline{\mathbf{Y}}$, $\underline{\mathbf{T}}$, by fixing two of them and updating the other. Let us start with the tensor $\underline{\mathbf{X}}$ and by minimizing the augmented Lagrangian function (22) with respect to the tensor $\underline{\mathbf{X}}$, we have

$$\underline{\mathbf{X}}_{k+1} = \min_{\underline{\mathbf{X}}} \ \mathcal{L}\left(\underline{\mathbf{X}}, \underline{\mathbf{Z}}_k, \underline{\mathbf{T}}_k, \mu_k\right), \tag{23}$$

which can be simplified as

$$\underline{\mathbf{X}}_{k+1} = \min_{\underline{\mathbf{X}}} \ \left(\lambda \left\|\underline{\mathbf{X}}\right\|_* + \frac{\mu_k}{2} \left\|\underline{\mathbf{X}} - \underline{\mathbf{Z}}_k + \frac{\underline{\mathbf{T}}_k}{\mu_k}\right\|_F^2\right). \tag{24}$$

The minimization problem (23), has the close form solution

$$\underline{\mathbf{X}}_{k+1} = \underline{\mathbf{D}}_\beta \left(\underline{\mathbf{Z}}_k - \frac{\underline{\mathbf{T}}_k}{\mu_k}\right), \tag{25}$$

where $\underline{\mathbf{D}}_\beta(.)$ is the tensor singular value thresholding operation and defined similar to the matrix case (See the Appendix). To update $\underline{\mathbf{Z}}$, consider

$$\underline{\mathbf{Z}}_{k+1} = \min_{\underline{\mathbf{Z}}} \ \mathcal{L}\left(\underline{\mathbf{X}}_{k+1}, \underline{\mathbf{Z}}, \underline{\mathbf{T}}_k, \mu_k\right), \tag{26}$$

which be simplified as (26) for $\underline{\mathbf{Z}}$ can be solved through:

$$\underline{\mathbf{Z}}_{k+1} = \min_{\underline{\mathbf{Z}}_k} \ \frac{1}{2} \left\|\mathcal{P}_{\underline{\Omega}}\left(\underline{\mathbf{Z}} - \underline{\mathbf{Y}}\right)\right\|_F^2 + \frac{\mu_k}{2} \left\|\underline{\mathbf{X}}_{k+1} - \underline{\mathbf{Z}} + \frac{\underline{\mathbf{T}}_k}{\mu_k}\right\|_F^2. \tag{27}$$

and has the closed form solution defined as

$$\underline{\mathbf{Z}}_{k+1} = \mathcal{P}_{\underline{\Omega}}^\perp \left(\underline{\mathbf{X}}_{k+1} + \frac{\underline{\mathbf{T}}_k}{\mu_k}\right) + \underline{\mathbf{Y}}. \tag{28}$$

The solution for $\underline{\mathbf{T}}$ is also converted to a simpler optimization as

$$\underline{\mathbf{T}}_{k+1} = \min_{\underline{\mathbf{T}}} \ \mathcal{L}\left(\underline{\mathbf{X}}_{k+1}, \underline{\mathbf{Z}}_{k+1}, \underline{\mathbf{T}}, \mu_k\right). \tag{29}$$

which can be solved through:

$$\underline{\mathbf{T}}_{k+1} = \underline{\mathbf{T}}_k + \mu_k \left(\underline{\mathbf{X}}_{k+1} - \underline{\mathbf{Z}}_{k+1}\right), \tag{30}$$

Similar to the matrix case, we update the penalty parameter μ via

$$\mu_{k+1} = \min \ \left(\mathcal{J}\mu_k, \mu_{max}\right). \tag{31}$$

C. Smoothing techniques

Low rank modeling of data has achieved tremendous success in tensor completion, however, only a low rank prior is inadequate for a successful recovery of the underlying tensor [43], [44], [45]. The case is much difficult when the number of

---

Algorithm 2: Algorithm for Smooth Tensor Nuclear Norm (STNN)

Input : An observed data tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the observation index tensor $\underline{\mathbf{\Omega}}$, and regularization parameter $\lambda > 0$, $\underline{\mathbf{X}}_0 = \underline{\mathbf{T}}_0$, $\underline{\mathbf{Z}}_0 = \underline{\mathbf{X}}_0$, $\mathcal{P}_{\underline{\mathbf{\Omega}}}\left(\underline{\mathbf{X}}\right) = \mathcal{P}_{\underline{\mathbf{\Omega}}}\left(\underline{\mathbf{Y}}\right)$.

Output: Completed data tensor $\underline{\mathbf{X}}$

1 Perform Super-pixel extraction to generate compressed data

2 while *A stopping criterion is not satisfied* **do**

3 Update $\underline{\mathbf{X}}_{k+1}$ with equation 25

4 Update $\underline{\mathbf{Z}}_{k+1}$ with equation 28

5 Perform Smoothing operation

6 Update $\underline{\mathbf{T}}_{k+1}$ with equation 30

7 Update μ_{k+1} using equation 31

8 Check convergence conditions

9 $\left\|\underline{\mathbf{X}}_{k+1} - \underline{\mathbf{X}}_k\right\|_* \leq tol$

10 **Compute** $\underline{\mathbf{X}} = \mathcal{P}_{\underline{\mathbf{\Omega}}}\left(\widehat{\underline{\mathbf{X}}}_H\right) + \mathcal{P}_{\underline{\mathbf{\Omega}}^\perp}\left(\widehat{\underline{\mathbf{X}}}\right)$

11 end

Fig. 6: Pipeline of proposed method

missing pixels are high. Thankfully, many real world images and data exhibit some smoothness prior along both the spatial and the third modes especially in the case of RGB images, videos, and hyperspectral images [44], [46]. As such, it becomes a very useful property in modelling these types of data. The assumption of smoothness in data means that the differences between neighboring values are small in certain domains. For example, non-negative natural images are smooth in the spatial domain. Therefore, the smoothing constraint is a common assumption used to improve results when dealing with some datasets such as images and videos. More precisely, the smoothness constraint is imposed on the underlying factors along with the low-rank assumption of the tensor decomposition used in reconstruction model. Indeed, matrix/tensor factorization methods with smoothness constraints have a wide range of applications that require robustness in the presence of noisy signals, including image in-painting, denoising, brain signal analysis and hyperspectral imaging. As a result, many tensor completion algorithms have been proposed imposing smoothness on the recovered data [47], [44], [48], [43], [49], [49]. A very popular regularizer used to impose piece-wise smoothness is the Total Variation (TV) [50]. The TV is determined by the l_1-norm of the difference

between neighboring elements. Many methods in matrix and tensor completion have used the TV approach [43], [51], [52],[44], [53], [45], [46], [54]. The works by [43] and [55] incorporates smoothness into a PARAFAC model for partially observed tensors. Both models created two variants for the completion task. The former proposed the models based on total variation and quadratic variation regularizes. The approach by [55] uses the total variation (TV) regularizer to also formulate the tensor completion model, taking advantage of a piecewise prior and local smoothness constraint. The approaches adopts Canonical Polyadic(CP) and Tucker decomposition for a simultaneous decomposition and completion task. Furthermore, [44] adopts the tensor smoothness constraints using smooth matrix factorizations. The methods in [56], [54], [57], [44] exploit the spatial piecewise smoothness prior of the underlying tensor by increasing the piecewise smoothness of each row in the data. Other smoothness approaches such as methods used in [49], [58],[59], [60], [61], [57] have demonstrated significant improvement in results using various low rank and higher order tensor networks. In addition, different techniques and methodologies can be used for smoothing out the elements of data elements such as low-pass filtering, moving averaging, locally estimated scatterplot smoothing (LOESS) and gaussian filtering. We perform a simple Gaussian smoothing filtering with standard deviation which returns the filtered image. The "imgaussfilt" command in Matlab can be used for the mentioned smoothing technique.

VI. EXPERIMENTS

In this section, we present an evaluation of the proposed algorithm using real colored images. Experiments or simulations were performed on a laptop computer with 16GB memory and a 2.60 GHz Intel(R) Core(TM) i7-5500U processor. The Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index measure (SSIM) metrics were used to evaluate and compare the performance of different algorithms. We mainly consider three experiments. In the first simulation, we extensively compare the proposed superpixel sampling approach with the random sampling and show that in all cases the superpixel method provide better results. In the second simulation, we will use the superpixel sampling to remove the pixels as was shown in the first simulation to be better than the random one and compared the proposed completion algorithms with the baseline completion algorithms. Here, the effect of smooth filtering in the reconstruction performance is also illustrated. In the third simulation, we consider the more challenging cases where the pixels are removed in structural way and again compared the proposed completion algorithms with the baseline algorithms. We have used the RGB images "Kate", "House", "Lena","Plane", and "Peppers" and also some images from the Kodak dataset [62] as our benchmark images which are shown in Figure 7. "Lena", "House", "Peppers" and "Kate" images are of size $256 \times 256 \times 3$ whiles the Kodak images are of size $512 \times 768 \times 3$.

Fig. 7: Colored images used in numerical experiments.

Example 1. (Comparison of superpixel sampling methods and random sampling.) This experiment is devoted to illustrating the superiority of the proposed superpixel sampling approach compared to the random sampling method which was mostly used in literature. In Section V (Figures 4 and 5), we showed the better performance of the superpixel sampling compared to

random sampling for kodak images (kodim03, kodim23) and here we make new experiments for the kodim15, kodim22 and kodim02 images. Here, we make more experiments using images with 60%, 30% and 20% of pixel sampling. The pixels are sampled according to different superpixel sampling approaches described in Section V including Centroid, Boundary, Multi-stage sampling and also random sampling. Note that for the completion part (stage 2), we have used Algorithm 2 (STNN) with smoothing filtering. The results are reported in Figure 8. As can be seen, in all scenarios, the centriod superpixel sampling provided the best results. These extensive simulations on a variety of images and using different categories clearly convinced us that the centroid superpixel approach can select better pixels than other superpixel approaches and also the random sampling method.

Example 2. (Comparing the SMNN, the STNN with the baseline completion algorithms) Our main goal in this experiment is to show the better recovery performance of the proposed SMNN and STNN than the baseline completion algorithms: LRMC [63], LRTC_TNN [63], TRPCA [39], HaLTRC[30], SPC(TV) [43] and WSTNN [53] which to the best of our knowledge have provided the sate-of-the-art results. All the hyper-parameters were tuned as used in the methods for fair comparison. This is to ensure that all the methods performed as best as possible. Note that since the first experiment confirmed the better performance of the centroid superpixel approach provides better results than other sampling methods, throughout this experiment we use it to sample pixels. The results PSNR of the reconstructed images for 70%,30%,20%,10% and 5% pixel sampling are displayed in Figures 9-13. We see that our proposed algorithms perform better than the other algorithms in most cases and for various sampling ratios. Also, the results shows that the STNN performed better than the SMNN. Moreover, to highlight the effect of filtering/smoothing scheme, we performed a new experiment with 30% of available pixels and presents results for three images (kodim03, kodim23 and Kate). The reconstructed images using the STNN with/without filtering are displayed in Figure 14 and the PNSR and SSIM results show that the smoothing/filtering technique can improve the recovery results.

Owing to the fact that our method uses smoothness, we also performed experiments comparing our methods with other tensor completion algorithms that incorporate smoothness. Figures 15 and 16 show results comparing some smoothed tensor completion methods such as LRTC-TV-I [55], LRTC-TV-II [55], SPC(QV) [43], and LRTV-PDS [47]. The results show that our methods can provide comparable results to the mentioned smoothed tensor completion methods and even in some cases, they achieve better performance. We performed experiments on images with 50% and 30% of pixel sampled both using random and superpixels sampling methods. The obtained results are shown in Figures 15 and 16. We see that the proposed technique can achieve better results than the baseline techniques. We can also observe that reconstruction of incomplete image whose pixels were sampled by the proposed superpixel method have better quality. To further examine the proposed smoothed completion algorithm, we used the Washington DC mall hyperspectral data [1] which is of size $1208 \times 307 \times 191$. We used a sub-tensor of $256 \times 256 \times 30$ and sampled only 50% of its pixels. We applied the proposed superpixel sampling method with the centroid pixel selection to the first frontal slice to compute the mask operator and this mask was used for all frontal slices. Then our proposed smoothed tensor completion method, SMF-LRTC [44], LRTC-TV-II [55], and TR-ALS [64] algorithms were applied to it to reconstruct the hyperspectral data. Figure 17 represents the reconstructed images obtained by the algorithms for band 20. Clearly, we see that the proposed smoothed tensor completion method provided better recovery results.

[1]https://engineering.purdue.edu/ biehl/MultiSpec/hyperspectral.html

(a) Reconstructed images with 60% of pixels sampled from the kodim15 image using different sampling techniques.

(b) Reconstructed images with 30% of pixels sampled from the kodim23 image using different sampling techniques.

(c) Reconstructed images with 30% of pixels sampled from the kodim22 image using different sampling techniques.

(d) Reconstructed images with 20% of pixels sampled form the kodim02 image using different sampling techniques.

Fig. 8: PSNR comparison of Kodak images using different superpixels sampling methods for Example 1. The experiment show that centroid pixel selection provides better results.

Example 3. (Image recovery performance of the STNN for images with structured missing pixels) In this experiment, we evaluate the efficiency of the proposed STNN which achieved the best results among other completion algorithms for recovering images with structured missing pixels which is a more challenging case. To this end, we considered three types of structured missing patterns depicted in the first rows of Figures 18-20. Then, apply STNN algorithm to estimate the missing pixels. The reconstructed images using the STNN algorithm are shown in the third rows of Figures 18-20. Clearly, the results confirmed that the STNN algorithm is also applicable for recovering images with structured missing pixels.

Fig. 9: PSNR comparison of different low rank completion methods with sampling 70% of pixels for Example 2.

Fig. 10: PSNR comparison of different low rank completion methods with sampling 30% of pixels for Example 2.

VII. CONCLUSION AND FUTURE WORKS

In this work, we investigated the effects of superpixel clustering and pixel selection for the task of image completion. More precisely, we proposed to apply superpixel clustering method as an efficient segmentation/clustering approach to capture important textures underlying a given image in some partitions. Then we select pixels from each cluster based on different strategies, e.g., center, boundary, etc. The experiment results showed that the best results can be achieved by selecting the centroid pixel (pixel located in the center). We also formulated the tensor completion based on the tubal tensor nuclear norm and also matrix nuclear norm applied on the unfolding matrices. We equipped the algorithm with a smoothing technique

Fig. 11: PSNR comparison of different low rank completion methods with sampling 20% of pixels for Example 2.

Fig. 12: PSNR comparison of different low rank completion methods with sampling 10% of pixels for Example 2.

Fig. 13: PSNR comparison of different low rank completion methods with sampling 5% of pixels for Example 2.

Fig. 14: Reconstruction comparison for the TNN algorithm with/without smoothing. We have sampled 30% of pixels for Example 2.

to achieve better results. Extensive simulation results on a variety of images show the effectiveness and applicability of the proposed algorithm. In the future work, we will use the truncated tubal nuclear norm [40] in minimization problem (19). Also acceleration of the proposed algorithm using the randomization technique is an interesting topic needs to be investigated.

Fig. 15: Comparison of smoothing completion algorithms with sampling 50% of pixels for Example 2. a) Centroid sampling b) Random sampling.

Fig. 16: Comparison of smoothing completion algorithms with sampling 30% of pixels for Example 2. a) Centroid sampling b) Random sampling

Fig. 17: PSNR comparison of tensor completion algorithms with sampling 50% of pixels from the WDC hyper-spectral Data for Example 2. The superpixel with centriod sampling was used.

| PSNR: | 36.073 | 36.012 | 35.673 | 36.847 | 33.870 | 33.637 |
| SSIM: | 0.986 | 0.973 | 0.985 | 0.987 | 0.976 | 0.968 |

Fig. 18: The structured results of the STNN algorithm for structured missing pixels for Example 3.

| PSNR: | 38.298 | 37.082 | 38.992 | 38.165 | 37.458 | 36.692 |
| SSIM: | 0.982 | 0.976 | 0.992 | 0.967 | 0.984 | 0.981 |

Fig. 19: The structured results of the STNN algorithm for structured missing pixels for Example 3.

PSNR: 33.167	33.317	32.052	33.534	31.844	32.114
SSIM: 0.939	0.909	0.936	0.974	0.914	0.957

Fig. 20: The structured results of the STNN algorithm for structured missing pixels for Example 3.

REFERENCES

[1] Q. Song, H. Ge, J. Caverlee, and X. Hu, "Tensor completion algorithms in big data analytics," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 1, pp. 1–48, 2019.

[2] E. Frolov and I. Oseledets, "Tensor methods and recommender systems," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 7, no. 3, p. e1201, 2017.

[3] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[4] A. Cichocki, N. Lee, I. V. Oseledets, A.-H. Phan, Q. Zhao, and D. P. Mandic, "Tensor networks for dimensionality reduction and large-scale optimization. Part 1: Perspectives and Challenges," *Foundations and Trends in Machine Learning*, vol. 9(4-5), pp. 249–429, 2016.

[5] S. M. Fazel, "Matrix rank minimization with applications." 2003.

[6] N. Li and B. Li, "Tensor completion for on-board compression of hyperspectral images," in *2010 IEEE International Conference on Image Processing*. IEEE, 2010, pp. 517–520.

[7] A. Albayrak and G. Bilgin, "Automatic cell segmentation in histopathological images via two-staged superpixel-based algorithms," *Medical & biological engineering & computing*, vol. 57, no. 3, pp. 653–665, 2019.

[8] R. T. Meinhold, *Robust Path-based Image Segmentation Using Superpixel Denoising*. Rochester Institute of Technology, 2018.

[9] J. Chen, Z. Li, and B. Huang, "Linear spectral clustering superpixel," *IEEE Transactions on image processing*, vol. 26, no. 7, pp. 3317–3330, 2017.

[10] Z. Li and J. Chen, "Superpixel segmentation using linear spectral clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1356–1363.

[11] J. Yan, Y. Yu, X. Zhu, Z. Lei, and S. Z. Li, "Object detection by labeling superpixels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5107–5116.

[12] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels," Tech. Rep., 2010.

[13] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[14] T. Malisiewicz and A. A. Efros, "Improving spatial support for objects via multiple segmentations," 2007.

[15] P. Neubert and P. Protzel, "Superpixel benchmark and comparison," in *Proc. Forum Bildverarbeitung*, vol. 6, 2012, pp. 1–12.

[16] K. Zhou, J. Li, W. Luo, Z. Li, J. Yang, H. Fu, J. Cheng, J. Liu, and S. Gao, "Proxy-bridged image reconstruction network for anomaly detection in medical images," *IEEE Transactions on Medical Imaging*, 2021.

[17] K. Sakurada and T. Okatani, "Change detection from a street image pair using CNN features and superpixel segmentation." in *BMVC*, vol. 61, 2015, pp. 1–12.

[18] S. Cao, Y. He, H. Zhang, W. Lv, L. Lu, and W. Chen, "Dynamic PET image reconstruction incorporating multiscale superpixel clusters," *IEEE Access*, vol. 9, pp. 28 965–28 975, 2021.

[19] S. Kumar, Y. Dai, and H. Li, "Superpixel soup: Monocular dense 3d reconstruction of a complex dynamic scene," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 5, pp. 1705–1717, 2019.

[20] C. L. Zitnick and S. B. Kang, "Stereo for image-based rendering using image over-segmentation," *International Journal of Computer Vision*, vol. 75, no. 1, pp. 49–65, 2007.

[21] A. E. Bayá and M. G. Larese, "Pixel sampling by clustering," *Expert Systems with Applications*, vol. 159, p. 113576, 2020.

[22] Y. Zhang, Y. Wang, X. Chen, X. Jiang, and Y. Zhou, "Spectral–spatial feature extraction with dual graph autoencoder for hyperspectral image clustering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 12, pp. 8500–8511, 2022.

[23] Z. Tian, L. Liu, Z. Zhang, and B. Fei, "Superpixel-based segmentation for 3d prostate mr images," *IEEE transactions on medical imaging*, vol. 35, no. 3, pp. 791–801, 2015.

[24] M. E. A. Bechar, N. Settouti, V. Barra, and M. A. Chikh, "Semi-supervised superpixel classification for medical images segmentation: application to detection of glaucoma disease," *Multidimensional Systems and Signal Processing*, vol. 29, pp. 979–998, 2018.

[25] C. Ouyang, C. Biffi, C. Chen, T. Kart, H. Qiu, and D. Rueckert, "Self-supervision with superpixels: Training few-shot medical image segmentation without annotation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*. Springer, 2020, pp. 762–780.

[26] Q. Huang, Z. Miao, S. Zhou, C. Chang, and X. Li, "Dense prediction and local fusion of superpixels: A framework for breast anatomy segmentation in ultrasound image with scarce data," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–8, 2021.

[27] H. E. Tasli, R. Sicre, and T. Gevers, "Superpixel based mid-level image description for image recognition," *Journal of Visual Communication and Image Representation*, vol. 33, pp. 301–308, 2015.

[28] T. Yokota, B. Erem, S. Guler, S. K. Warfield, and H. Hontani, "Missing slice recovery for tensors using a low-rank model in embedded space," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8251–8259.

[29] L. T. Nguyen, J. Kim, and B. Shim, "Low-rank matrix completion: A contemporary survey," *IEEE Access*, vol. 7, pp. 94 215–94 237, 2019.

[30] 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*, vol. 35, no. 1, pp. 208–220, 2012.

[31] C. Lu, J. Feng, S. Yan, and Z. Lin, "A unified alternating direction method of multipliers by majorization minimization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 527–541, 2017.

[32] F. R. Bach, "Consistency of the group lasso and multiple kernel learning." *Journal of Machine Learning Research*, vol. 9, no. 6, 2008.

[33] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[34] L. Yuan, C. Li, D. Mandic, J. Cao, and Q. Zhao, "Tensor ring decomposition with rank minimization on latent space: An efficient approach for tensor completion," *arXiv preprint arXiv:1809.02288*, 2018.

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

[36] M. E. Kilmer, K. Braman, N. Hao, and R. C. Hoover, "Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging," *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 1, pp. 148–172, 2013.

[37] Z. Zhang and S. Aeron, "Exact tensor completion using t-svd," *IEEE Transactions on Signal Processing*, vol. 65, no. 6, pp. 1511–1526, 2016.

[38] C. Lu, X. Peng, and Y. Wei, "Low-rank tensor completion with a new tensor nuclear norm induced by invertible linear transforms," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5996–6004.

[39] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, "Tensor robust principal component analysis with a new tensor nuclear norm," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 4, pp. 925–938, 2019.

[40] S. Xue, W. Qiu, F. Liu, and X. Jin, "Low-rank tensor completion by truncated nuclear norm regularization," in *2018 24th International Conference on Pattern Recognition (ICPR)*.   IEEE, 2018, pp. 2600–2605.

[41] H. Xu, J. Zheng, X. Yao, Y. Feng, and S. Chen, "Fast tensor nuclear norm for structured low-rank visual inpainting," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 2, pp. 538–552, 2021.

[42] G. Song, M. K. Ng, and X. Zhang, "Robust tensor completion using transformed tensor singular value decomposition," *Numerical Linear Algebra with Applications*, vol. 27, no. 3, p. e2299, 2020.

[43] T. Yokota, Q. Zhao, and A. Cichocki, "Smooth Parafac decomposition for tensor completion," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5423–5436, 2016.

[44] Y.-B. Zheng, T.-Z. Huang, T.-Y. Ji, X.-L. Zhao, T.-X. Jiang, and T.-H. Ma, "Low-rank tensor completion via smooth matrix factorization," *Applied Mathematical Modelling*, vol. 70, pp. 677–695, 2019.

[45] W. He, N. Yokoya, L. Yuan, and Q. Zhao, "Remote sensing image reconstruction using tensor ring completion and total variation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 11, pp. 8998–9009, 2019.

[46] W. He, L. Yuan, and N. Yokoya, "Total-variation-regularized tensor ring completion for remote sensing image reconstruction," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.   IEEE, 2019, pp. 8603–8607.

[47] T. Yokota and H. Hontani, "Simultaneous visual data completion and denoising based on tensor rank and total variation minimization and its primal-dual splitting algorithm," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3732–3740.

[48] T. K. Sinha, J. Naram, and P. Kumar, "Nonnegative low-rank tensor completion via dual formulation with applications to image and video completion," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 3732–3740.

[49] S. Ahmadi-Asl, M. G. Asante-Mensah, A. Cichocki, A.-H. Phan, I. Oseledets, and J. Wang, "Cross tensor approximation for image and video completion," *arXiv preprint arXiv:2207.06072*, 2022.

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

[51] T. Yokota and A. Cichocki, "Tensor completion via functional smooth component deflation," in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 2514–2518.

[52] M. Ding, T.-Z. Huang, T.-Y. Ji, X.-L. Zhao, and J.-H. Yang, "Low-rank tensor completion using matrix factorization based on tensor train rank and total variation," *Journal of Scientific Computing*, vol. 81, pp. 941–964, 2019.

[53] Y.-B. Zheng, T.-Z. Huang, X.-L. Zhao, T.-X. Jiang, T.-Y. Ji, and T.-H. Ma, "Tensor n-tubal rank and its convex relaxation for low-rank tensor recovery," *Information Sciences*, vol. 532, pp. 170–189, 2020.

[54] T.-Y. Ji, T.-Z. Huang, X.-L. Zhao, T.-H. Ma, and G. Liu, "Tensor completion using total variation and low-rank matrix factorization," *Information Sciences*, vol. 326, pp. 243–257, 2016.

[55] X. Li, Y. Ye, and X. Xu, "Low-rank tensor completion with total variation for visual data inpainting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[56] J.-F. Cai, R. H. Chan, and Z. Shen, "A framelet-based image inpainting algorithm," *Applied and Computational Harmonic Analysis*, vol. 24, no. 2, pp. 131–149, 2008.

[57] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, T.-Y. Ji, and L.-J. Deng, "Matrix factorization for low-rank tensor completion using framelet prior," *Information Sciences*, vol. 436, pp. 403–417, 2018.

[58] C.-Y. Ko, K. Batselier, L. Daniel, W. Yu, and N. Wong, "Fast and accurate tensor completion with total variation regularized tensor trains," *IEEE Transactions on Image Processing*, vol. 29, pp. 6918–6931, 2020.

[59] J. Xue, Y. Zhao, S. Huang, W. Liao, J. C.-W. Chan, and S. G. Kong, "Multilayer sparsity-based tensor decomposition for low-rank tensor completion," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 11, pp. 6916–6930, 2021.

[60] J. Xue, Y. Zhao, W. Liao, J. C.-W. Chan, and S. G. Kong, "Enhanced sparsity prior model for low-rank tensor completion," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 4567–4581, 2019.

[61] X.-T. Li, X.-L. Zhao, T.-X. Jiang, Y.-B. Zheng, T.-Y. Ji, and T.-Z. Huang, "Low-rank tensor completion via combined non-local self-similarity and low-rank regularization," *Neurocomputing*, vol. 367, pp. 1–12, 2019.

[62] R. Franzen. Kodak lossless true color image suite. [Online]. Available: http://r0k.us/graphics/kodak/

[63] C. Lu, J. Feng, S. Yan, and Z. Lin, "A unified alternating direction method of multipliers by majorization minimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 527—-541, 2018.

[64] W. Wang, V. Aggarwal, and S. Aeron, "Efficient low rank tensor ring completion," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5697–5705.

[65] M. E. Kilmer and C. D. Martin, "Factorization strategies for third-order tensors," *Linear Algebra and its Applications*, vol. 435, no. 3, pp. 641–658, 2011.

CONFLICT OF INTEREST STATEMENT

The authors have no conflicts of interest to declare.

APPENDIX

We provide definitions to some concepts used in our paper. We also provide some algorithms we adopted for our paper.

Definition 1. (t-product) The t-product of two tensors $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and $\underline{\mathbf{Y}} \in \mathbb{R}^{I_2 \times I_4 \times I_3}$, is given by $\underline{\mathbf{C}} \in \mathbb{R}^{I_1 \times I_4 \times I_3}$, this is defined as

$$\underline{\mathbf{C}} = \underline{\mathbf{X}} * \underline{\mathbf{Y}} = \text{fold}\left(\text{circ}\left(\underline{\mathbf{X}}\right) . \text{unfold}\left(\underline{\mathbf{Y}}\right)\right), \tag{32}$$

where

$$\text{circ}\left(\underline{\mathbf{X}}\right) = \begin{bmatrix} \underline{\mathbf{X}}(:,:,1) & \underline{\mathbf{X}}(:,:,I_3) & \cdots & \underline{\mathbf{X}}(:,:,2) \\ \underline{\mathbf{X}}(:,:,2) & \underline{\mathbf{X}}(:,:,1) & \cdots & \underline{\mathbf{X}}(:,:,3) \\ \vdots & \vdots & \ddots & \vdots \\ \underline{\mathbf{X}}(:,:,I_3) & \underline{\mathbf{X}}(:,:,I_3-1) & \cdots & \underline{\mathbf{X}}(:,:,1) \end{bmatrix},$$

and

$$\text{unfold}(\underline{\mathbf{Y}}) = \begin{bmatrix} \underline{\mathbf{Y}}(:,:,1) \\ \underline{\mathbf{Y}}(:,:,2) \\ \vdots \\ \underline{\mathbf{Y}}(:,:,I_3) \end{bmatrix}, \quad \underline{\mathbf{Y}} = \text{fold}\left(\text{unfold}\left(\underline{\mathbf{Y}}\right)\right).$$

It can be seen that the t-product operation (32) is equivalent to the circular convolution operator, and can therefore be easily computed through the Fast Fourier Transform (FFT). To be precise, all tubes from the two tensors $\underline{\mathbf{X}}$, $\underline{\mathbf{Y}}$ are transformed into the frequency domain, then the frontal slices of the spectral tensors are multiplied. we then apply the Inverse Fast Fourier Transform (IFFT) to all the tubes in the last tensor. The t-product can also be written in the Fourier domain as follows:

$$\underline{\mathbf{C}} = \underline{\mathbf{X}} * \underline{\mathbf{Y}} \Longleftrightarrow \widehat{\mathbf{C}} = \widehat{\mathbf{X}}\widehat{\mathbf{Y}}, \tag{33}$$

where $\widehat{\mathbf{X}}, \widehat{\mathbf{Y}}$ and $\widehat{\mathbf{C}}$ are block diagonal matrices defined as follows:

$$\widehat{\mathbf{X}} = bdiag(\widehat{\underline{\mathbf{X}}}) = \begin{bmatrix} \widehat{\mathbf{X}}^{(1)} & & & \\ & \widehat{\mathbf{X}}^{(2)} & & \\ & & \ddots & \\ & & & \widehat{\mathbf{X}}^{(I_3)} \end{bmatrix}$$

where $\widehat{\mathbf{X}}^{(1)}$ is the a matrix computed by applying the fast Fourier transform. The operator $bdiag(.)$ maps the tensor $\widehat{\underline{\mathbf{X}}}$ to the block diagonal matrix $\widehat{\mathbf{X}}$. The procedure for t-product in the Fourier domain is summarized in Algorithm 3.

Definition 2. (Transpose) The transpose of a tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is denoted by $\underline{\mathbf{X}}^T \in \mathbb{R}^{I_2 \times I_1 \times I_3}$. It is obtained by applying transpose to all the frontal slices and then reversing the order of the transposed frontal slices from the second through to the last frontal slice.

Definition 3. (Identity tensor) An identity tensor $\underline{\mathbf{I}} \in \mathbb{R}^{I_1 \times I_1 \times I_3}$ is a tensor with first frontal slice being an identity matrix of

---

Algorithm 3: The t-product tensor in the Fourier domain [65]

Input : Two data tensors $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, $\underline{\mathbf{Y}} \in \mathbb{R}^{I_2 \times I_4 \times I_3}$
Output: t-product $\underline{\mathbf{C}} = \underline{\mathbf{X}} * \underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_4 \times I_3}$
1 $\widehat{\underline{\mathbf{X}}} = \text{fft}\left(\underline{\mathbf{X}}, [], 3\right)$
2 $\widehat{\underline{\mathbf{Y}}} = \text{fft}\left(\underline{\mathbf{Y}}, [], 3\right)$
3 for $i = 1, 2, \ldots, I_3$ **do**
4 $\quad \widehat{\underline{\mathbf{C}}}\left(:,:,i\right) = \widehat{\underline{\mathbf{X}}}\left(:,:,i\right) \widehat{\underline{\mathbf{Y}}}\left(:,:,i\right)$
5 end
6 $\underline{\mathbf{C}} = \text{ifft}\left(\widehat{\underline{\mathbf{C}}}, [], 3\right)$

---

size $I_1 \times I_1$, and all other frontal slices being equal to zero.

Definition 4. (Orthogonal tensor) A tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_1 \times I_3}$ is orthogonal if $\underline{\mathbf{X}}^T * \underline{\mathbf{X}} = \underline{\mathbf{X}} * \underline{\mathbf{X}}^T = \underline{\mathbf{I}}$ is satisfied.

Definition 5. (f-diagonal tensor) An f-diagonal tensor is a tensor with all of its frontal slices being diagonal.

Definition 6. (t-SVD) A tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, can be decomposed as

$$\underline{\mathbf{X}} = \underline{\mathbf{U}} * \underline{\mathbf{S}} * \underline{\mathbf{V}}^T,$$

where $\underline{\mathbf{U}} \in \mathbb{R}^{I_1 \times I_1 \times I_3}$, $\underline{\mathbf{V}} \in \mathbb{R}^{I_2 \times I_2 \times I_3}$ are orthogonal tensors, and tensor $\underline{\mathbf{S}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is f-diagonal.

Algorithm 4: Truncated t-SVD [36]

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and target tubal rank R
Output: $\underline{\mathbf{U}}_R \in \mathbb{R}^{I_1 \times R \times I_3}$, $\underline{\mathbf{S}}_R \in \mathbb{R}^{R \times R \times I_3}$, $\underline{\mathbf{V}}_R \in \mathbb{R}^{I_2 \times R \times I_3}$
1 $\widehat{\underline{\mathbf{X}}} = \text{fft}\left(\underline{\mathbf{X}}, [], 3\right)$
2 for $i = 1, 2, \ldots, I_3$ **do**
3 $\quad [\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{truncated\_svd}\left(\bar{\mathbf{X}}(:,:,i), R\right)$
4 $\quad \widehat{\underline{\mathbf{U}}}\left(:,:,i\right) = \mathbf{U}$
5 $\quad \widehat{\underline{\mathbf{S}}}\left(:,:,i\right) = \mathbf{S}$
6 $\quad \widehat{\underline{\mathbf{V}}}\left(:,:,i\right) = \mathbf{V}$
7 end
8 $\underline{\mathbf{U}} = \text{ifft}\left(\widehat{\underline{\mathbf{U}}}, [], 3\right), \ \underline{\mathbf{S}} = \text{ifft}\left(\widehat{\underline{\mathbf{S}}}, [], 3\right), \ \underline{\mathbf{V}} = \text{ifft}\left(\widehat{\underline{\mathbf{V}}}, [], 3\right)$

The t-SVD can be obtained using the SVD of frontal slices of the original data tensor in the Fourier domain. The algorithm for computing the t-SVD for tensors is outlined in Algorithm 4. So, for the t-SVD of $\underline{\mathbf{X}}$, we have:

$$\widehat{\mathbf{X}}^{(i)} = \widehat{\mathbf{U}}^{(i)} * \widehat{\mathbf{S}}^{(i)} * (\widehat{\mathbf{V}}^{(i)})^T, \ \ i = 1, 2, \ldots, I_3, \tag{34}$$

with $\widehat{\mathbf{X}}^{(i)}$ being the i-th frontal slice in the Fourier domain, i.e., $\widehat{\mathbf{X}}^{(i)} = \widehat{\underline{\mathbf{X}}}(:,:,i)$.

Now, we introduce the tensor nuclear norm based on the t-product.

Definition 7. (tensor tubal rank and nuclear norm) [40] The tensor tubal rank of $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is defined as the maximum rank among all frontal slices of an f-diagonal tensor $\underline{\mathbf{S}}$, i.e., $\max \ rank(\mathbf{S}^{(i)})$. Additionally, the tensor nuclear norm $\|\underline{\mathbf{X}}\|_*$ is

defined as the sum of the singular values in all frontal slices of $\underline{\mathbf{S}}$, i.e.,

$$\|\underline{\mathbf{X}}\|_* = \text{tr}(\underline{\mathbf{S}}) = \sum_{i=1}^{I_3} \text{tr}(\mathbf{S}^{(i)}), \tag{35}$$

where $\underline{\mathbf{S}}^{(i)}$ were defined in (34).

It is shown in [40] that the trace of tensor product $(\underline{\mathbf{X}} * \underline{\mathbf{Y}})$ equals to the trace of the product of $\widehat{\mathbf{X}}^{(1)}$ and $\widehat{\mathbf{Y}}^{(1)}$, that is

$$\text{tr}(\underline{\mathbf{X}} * \underline{\mathbf{Y}}) = \text{tr}(\widehat{\mathbf{X}}^{(1)}\widehat{\mathbf{Y}}^{(1)}). \tag{36}$$

Then, it is proved [40] that the tensor nuclear norm defined in (35) can be simplified as

$$\|\underline{\mathbf{X}}\|_* = \text{tr}(\underline{\mathbf{S}}) = \text{tr}(\widehat{\mathbf{S}}^{(1)}) = \left\|(\widehat{\mathbf{X}}^{(1)})\right\|_*. \tag{37}$$

Definition 8. (Tensor singular value thresholding) The singular value thresholding (SVT) [40] operator $\underline{\mathbf{D}}_\beta(.)$ is performed on each frontal slice of the f-diagonal tensor $\widehat{\underline{\mathbf{S}}}$. That is,

$$\underline{\mathbf{D}}_\beta(\underline{\mathbf{X}}) = \underline{\mathbf{U}} * \underline{\mathbf{D}}_\beta(\underline{\mathbf{S}}) * \underline{\mathbf{V}}^T \tag{38}$$

where $\underline{\mathbf{D}}_\beta(\underline{\mathbf{S}})$ is the inverse FFT of $\underline{\mathbf{D}}_\beta(\widehat{\underline{\mathbf{S}}})$. and $\underline{\mathbf{D}}_\beta(\widehat{\underline{\mathbf{S}}}^{(i)}) = \text{diag}(\max\{\sigma_t - \beta, 0\}_{1 \le t \le R})$, $i = 1, \dots, I_3$, $\beta > 0$ is a constant and R is the tubal rank.

Algorithm 5: The SLIC method [12]

1 **Set** clusters centers $c_k = [l_k, a_k, b_k, x_k, y_k]^T$ by taking regular grid steps S to sample pixels
2 **shift** Cluster centers in an $n \times n$ neighborhood to the lowest gradient location
3 **while** $\epsilon \ge$ *threshold* **do**
4   **for** *each cluster center c_k* **do**
5   **Assign** the best matching pixels from a $2S \times 2S$ square neighborhood around the cluster center c_k using the distance measure for D_s in Equation (3)
6   **end**
7   **Compute** cluster centers
8   **Calculate** the residual error ϵ using the L_1 distance between recomputed centers and previous centers.
9 **end**
