Non-local Patch-based Low-rank Tensor Ring Completion for Visual Data

Yicong He and George K. Atia, *Senior Member, IEEE*

**Abstract**—Tensor completion is the problem of estimating the missing entries of a partially observed tensor with a certain low-rank structure. It improves on matrix completion for image and video data by capturing additional structural information intrinsic to such data. Traditional completion algorithms treat the entire visual data as a tensor, which may not always work well especially when camera or object motion exists. In this paper, we develop a novel non-local patch-based tensor ring completion algorithm. In the proposed approach, similar patches are extracted for each reference patch along both the spatial and temporal domains of the visual data. The collected patches are then formed into a high-order tensor and a tensor ring completion algorithm is proposed to recover the completed tensor. A novel interval sampling-based block matching (ISBM) strategy and a hybrid completion strategy are also proposed to improve efficiency and accuracy. Further, we develop an online patch-based completion algorithm to deal with streaming video data. An efficient online tensor ring completion algorithm is proposed to reduce the time cost. Extensive experimental results demonstrate the superior performance of the proposed algorithms compared with state-of-the-art methods.

I. INTRODUCTION

Multi-way data analysis uses techniques that represent data as multi-dimensional arrays known as tensors. It has drawn increased attention in recent years given its ability to reveal patterns intrinsic to high-order data undetected by other methods by capturing correlations across its different dimensions. Such techniques have found numerous applications in machine learning [1], [2], [3], [4], signal processing [5], [6], [7] and computer vision [8], [9], [10], [11].

Tensor completion, an extension to the matrix completion problem, aims to fill in the missing entries of a partially observed tensor by leveraging its low-rank structure, which stems from the redundancy of the information it represents [12], [13]. For example, a natural multi-channel image or a video sequence can be represented as tensors that exhibit high correlations across the spatial and temporal dimensions. Based on different tensor rank models, several completion algorithms have been proposed, such as CP-based algorithms [14], [15], tucker-based algorithms [16], [17], [18], tubal-based algorithms [19], [20], [21], tensor train-based algorithms [9] and tensor ring-based algorithms [22], [23], [24]. Extensive experimental results of these tensor completion algorithms have shown their superior completion performance compared to matrix completion.

Recently, a non-local patch-based method has been applied to tensor completion of visual data. The non-local patch-based method, originally introduced for image and video denoising [25], [26], [27], finds similar image patches across the spatial or temporal domain using block matching. Taking advantage of the high correlations that exist among these patches, the stacked patches along the temporal dimension can be well approximated by a low-rank tensor (See Fig.1), whose missing pixels can be recovered using tensor completion. Based on that, [28] proposed a non-local patch-based t-SVD completion algorithm for image completion. The missing pixels of the images are filled using triangle interpolation, then block matching is used to locate similar patches in the interpolated images. However, the patch matching performance is directly impacted by the completion or interpolation performance. When the number of observed pixels is relatively small, the interpolation performance deteriorates, resulting in poor patch matching performance. Further, t-SVD-based completion can only complete third order tensors, which limits its ability to adequately handle higher order tensors.

Rather than finding similar 2-D patches, [29] proposed a non-local tensor train completion algorithm by searching for similar 3-D tubes across the spatial domain. The matched tubes are used to form a high-order tensor whose missing entries are recovered using a tensor train completion algorithm. Since

![Figure 1](image-url)
more pixels are observed in tubes than in patches, the matching accuracy can be improved, thereby reducing the reliance on interpolation. However, tube matching requires more storage and higher computational complexity than block matching, which limits its applicability to long video sequences. At the same time, the high-order tensor formed from tubes significantly increases the time complexity of nuclear norm-based tensor train completion algorithms.

In this paper, we propose a new non-local patch-based approach for visual data completion. To improve accuracy while maintaining efficiency, we develop novel strategies for both block matching and completion. For block matching, rather than the interpolation used in [23], we improve the matching performance by propagating the observed pixels to unobserved neighboring pixels, and perform the matching procedure on the dilated data. To improve the efficiency of the search, we propose an interval sampling method to down-sample the partially observed data. Block matching is then performed on the down-sampled data thereby reducing the time complexity. For tensor completion, we develop a hybrid completion strategy that combines both nuclear norm and decomposition-based completion algorithms. For each tensor, the completion method is selected adaptively based on the observability ratio and the texture variance to guarantee both high accuracy and efficiency.

In addition, we develop a non-local patch-based online tensor completion framework for streaming video sequences. A first-in-first-out (FIFO) buffer is used to store the last few frames and a core tensor pool is constructed to store the entire core tensor required to recover the frames in the buffer. The tensor is constructed sequentially by tracking similar patches as a new frame arrives. Unlike previous nuclear norm-based methods [28], [29] which require the full tensor at once, we develop a new online tensor ring completion algorithm which can adaptively estimate the tensor as the tensor size increases. With a warm start based on the previous core tensors estimation, the proposed online completion algorithm converges fast and greatly saves the time cost.

The main contributions of the paper can be summarized as follows.

1. We develop a novel tensor completion framework incorporating a non-local patch-based method and a tensor ring rank model. New interval sampling and block matching method are proposed to improve the efficiency. To improve the completion performance, a hybrid completion strategy combining nuclear norm and tensor decomposition methods is developed, which can adapt to different texture and observability.

2. We develop a new online patch-based tensor completion algorithm for streaming data. A patch tracking strategy that adds new patches to tensors as new frames arrive is developed. Also, an online tensor ring algorithm is proposed. By updating the core tensor using the previous estimate, the time cost can be further reduced.

3. We conduct extensive experiments that demonstrate the superior performance of the proposed algorithms over state-of-the-art tensor completion algorithms for both images and video sequences.

The paper is organized as follows. In Section II we introduce our notation and provide some preliminary background on the tensor ring and its properties. In Section III we propose the new non-local patch-based tensor ring completion algorithm. In Section IV we develop the non-local patch-based online tensor completion framework and propose the corresponding online completion algorithm. In Section V we present experimental results to demonstrate the reconstruction performance. Finally, conclusion is given in Section VI.

II. PRELIMINARIES

Uppercase script letters are used to denote tensors (e.g., \( \mathcal{X} \)), boldface uppercase letters to denote matrices (e.g., \( \mathbf{X} \)) and boldface lowercase letters to denote vectors (e.g., \( \mathbf{x} \)). An \( N \)-order tensor is defined as \( \mathcal{X} \in \mathbb{C}^{I_1 \times \cdots \times I_N} \), where \( I_i, i \in [N] := \{1, \ldots, N\} \) is the dimension of the \( i \)-th way of the tensor, and \( \mathcal{X}_{i_1 \ldots i_N} \) denotes the \((i_1, i_2, \ldots, i_N)\)-th entry of tensor \( \mathcal{X} \). The vector \( \mathbf{x} \in \mathbb{R}^{I_1 \times \cdots \times I_N} \) denotes the vectorization of tensor \( \mathcal{X} \). For a 3-way order tensor (i.e., \( N = 3 \)), the notation \( \mathcal{X}(\cdot, i, \cdot), \mathcal{X}(i, \cdot, \cdot), \mathcal{X}(\cdot, \cdot, \cdot) \) denotes the frontal, lateral, horizontal slices of \( \mathcal{X} \), respectively. The Frobenius norm of a tensor \( \mathcal{X} \) is defined as \( \|\mathcal{X}\|_F = \sqrt{\sum_{i_1, \ldots, i_N} |\mathcal{X}_{i_1 \ldots i_N}|^2} \). The product \( \mathcal{A} \circ \mathcal{B} \) denotes the Hadamard (element-wise) product of two tensors \( \mathcal{A} \) and \( \mathcal{B} \). Next, we introduce some definitions and theorems for tensor ring used throughout the paper.

Definition 1 (TR Decomposition [30]). In TR decomposition, a high-order tensor \( \mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N} \) can be represented using a sequence of circularly contracted 3-order core tensors \( \mathcal{Z}_k \) for \( k = 1, \ldots, N, r_{N+1} = r_1 \). Specifically, the element-wise relation of tensor \( \mathcal{X} \) and its TR core tensors \( \{\mathcal{Z}_k\}_{k=1}^N \) is defined as

\[
\mathcal{X}_{i_1 \ldots i_N} = \text{Tr} \left( \prod_{k=1}^N \mathcal{Z}_k(:, i_k, :) \right),
\]

where \( \text{Tr}() \) is the matrix trace operator. The relation of tensor \( \mathcal{X} \) and its TR decomposition core tensors \( \{\mathcal{Z}_k\}_{k=1}^N \) is written as \( \mathcal{X} = \mathcal{R}(\mathcal{Z}_1, \mathcal{Z}_2, \ldots, \mathcal{Z}_N) \), where \( \mathcal{R} \) is the function defined through (1) and \([r_1, \ldots, r_N]\) is called TR rank.

Definition 2 (Tensor Circular Unfolding [31], [24]). Let \( \mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \) be a \( N \)-order tensor. The tensor circular unfolding of \( \mathcal{X} \) is a matrix denoted by \( \mathbf{X}_{(k,d)} \) of size \( \prod_{k=1}^{N-1} I_k \times \prod_{j=k+1}^{N} I_j \), whose elements are defined by \( X_{(k,d)}(i_k \cdots i_{k+d-1}, i_{k+d} \cdots i_{k+N-1}) = X(i_1, i_2, \ldots, i_d) \) with \( I_{k+N} = I_k, i_{k+N} = i_k \) for \( k = 1, \ldots, N \) and \( \sum_{c=a}^{b-1} (i_c - 1) \prod_{j=a}^{c-1} I_j = I_{k+d} \). With this notation, the mode-\( k \) unfolding matrix of tensor \( \mathcal{X} \) is denoted by \( \mathbf{X}_{[k]} \) of size \( n_k \times \prod_{j \neq k} n_j \) with its elements defined by

\[
X_{[k]}(i_k, i_{k+1}, \ldots, i_{N+1}, \ldots, i_k) = X(i_1, i_2, \ldots, i_N).
\]
Another classical mode-k unfolding matrix of \( \mathcal{X} \) is denoted by 
\[
X_{(k)} = (i_k, i_1, \ldots, i_{k-1}i_{k+1}, \ldots, i_N) = \mathcal{X} (i_1, i_2, \ldots, i_N).
\]

**Theorem 1** (30). Given a TR decomposition \( \mathcal{X} = \mathbb{R}(\mathcal{Z}_1, \ldots, \mathcal{Z}_N) \), its mode-k unfolding matrix \( X_{[k]} \) can be written as 
\[
X_{[k]} = Z_{[k]} \left( Z_{[2]}^{\neq k} \right)^T
\]
where \( Z_{[k]} \in \mathbb{R}^{r_{k+1} \times \Pi_N^{k-1}} \) is a subchain obtained by merging all cores except \( Z_k \), whose slice matrices are defined by 
\[
Z_{\neq k} (i_{k+1} \cdots i_N) = \prod_{j=k+1}^{N} Z_j (i_{j+1} i_j) \prod_{j=1}^{k-1} Z_j (i_j i_j)
\]

**Theorem 2.** Assume \( \mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_k} \) is a N-order tensor with TR rank \( [r_1, r_2, \ldots, r_N] \), then for each unfolding matrix \( X_{<k,d>} \), 
\[
\text{rank} (X_{<k,d>}) \leq r_k r_{k+d},
\]
where \( r_{k+N} = r_k \) for \( k = 1, \ldots, N \).

**III. Non-local patch-based tensor ring completion**

In this section, we propose our non-local patch-based approach for visual data completion, which is illustrated in Fig. 1. Assume we have a partially observed video sequence \( \{\mathbf{Y}^t\}_{t=1}^{T} \), \( \mathbf{Y}^t \in \mathbb{R}^{I_1 \times I_2 \times n} \) with corresponding observed pixel index sets \( \{\Omega_i\}_{t=1}^{T} \). For grey scale and color frames, the value \( n \) is 1 and 3, respectively. Before processing, all the frames are padded by mirroring \( b \) pixels at all boundaries and corners, resulting in a padded video sequence \( \{\tilde{\mathbf{Y}}^t\}_{t=1}^{T} \) with frame size \((I_1 + 2b) \times (I_2 + 2b) \times n \) and corresponding observation index sets \( \{\tilde{\Omega}^t\}_{t=1}^{T} \). For a given (reference) video frame \( \tilde{\mathbf{Y}}^t \), we first divide the frame into overlapping patches of size \( m \times m \times n \) with a number of overlap pixels \( o \). Then, for each patch, we find the similar patches among adjacent frames \( \{\tilde{\mathbf{Y}}^{t-d} \}_{d=1}^{d} \) within a searching window size \( l \times l \times (2d+1) \) using block matching, where \( l \) and \( d \) controls the search region along the spatial and temporal dimensions, respectively. Then, the nearest \( K \) patches (i.e., with the smallest Euclidean distances) are selected and stacked in a tensor of size \( m \times m \times n \times K \). Subsequently, the tensor ring completion algorithm is applied to recover the full tensor. Finally, the recovered frame \( \tilde{\mathbf{Y}}^t \) for \( \mathbf{Y}^t \) is obtained by aggregating all the recovered patches belonging to \( \mathbf{Y}^t \) and removing padded border pixels.

The approach in Fig. 2 bears some resemblance to the video denoising framework in [27], however, the problem at hand is substantially more challenging due to the existence of missing pixels. In the sequel, we will discuss the main ensuing challenges and present our proposed solutions.

**A. Block matching with missing data**

Similar to [27], [28], we perform block matching to find similar patches. However, in sharp contrast to [27], we only observe a portion of the pixels in any given patch. In turn, block matching becomes challenging, especially when the number of the observed pixels is relatively small. For example, for a patch of size \( 32 \times 32 \) and observation ratio 10%, the number of common observed pixels between two patches will be about 10 in average. In this case, the matching accuracy degrades due to lack of common information. One possibility to address this problem, would be to use interpolation as in [28] to first fill the missing entries then perform block matching on the interpolated images. However, directly filling the missing entries could introduce error, which may misinform the matching process. In this paper, we address this problem by using image dilation, that is, we propagate the observed pixels to the unobserved neighboring pixels as shown in Fig. 3. The \((i,j)\)-th entry of the dilated frame \( \tilde{\mathbf{Y}} \) is given by

\[
\tilde{Y}_{i,j} = \begin{cases} 
Y_{i,j}, & (i,j) \in \tilde{\Omega} \\
\max_{(p,q) \in \mathcal{N}_{i,j} \cap \tilde{\Omega}} Y_{p,q} - Y_{i,j} \in \tilde{\Omega}, \mathcal{N}_{i,j} \cap \tilde{\Omega} \neq \emptyset 
\end{cases}
\]

where \( \mathcal{N}_{i,j} \) is the index set formed by 8 neighbors of index \((i,j)\), and NaN denotes that the corresponding entry is missing.

Following dilation, the observation percentage is greatly improved. We remark that using a dilated image for matching may yield patches that are within a small offset from the true matching positions. However, since it is a small offset, it does not affect the completion performance significantly, which will be verified in the experiments.

**B. Interval sampling for block matching**

Another bottleneck for the block matching-based method is the computational efficiency. Searching similar patches could take a long time if the size of the search window is large. On the other hand, a small search window size may decrease the matching performance. To improve the efficiency of block matching, one could make use of a hierarchical or multi-resolution search method [33, 34], which searches for similar patches on different coarse to fine scales. However, due to the existence of missing entries, the hierarchical search method cannot be directly applied to our setting. To improve the matching efficiency while maintaining accuracy, we propose a new interval sampling-based block matching (ISBM) method, illustrated in Fig. 3. Given a sampling interval parameter \( s \), both reference image \( \mathbf{Y}^r \) and source image \( \tilde{\mathbf{Y}}^r \) are down-sampled by selecting pixels with interval \( s \). Then, the corresponding \( s^2 \) down-sampled images \( \{\mathbf{Y}^{r,s}_{i}i=1\ldots s^2\} \) and \( \{\tilde{\mathbf{Y}}^{r,s}_{i}i=1\ldots s^2\} \) of size \((I_1 + 2b)/s \times (I_2 + 2b)/s\) are obtained (here we assume that \( b \) is such that \( I_1 + 2b \) and \( I_2 + 2b \) are divisible by \( s \)). Block matching is performed on the down-sampled images \( \{\mathbf{Y}^{r,s}_{i}i=1\ldots s^2\} \) and \( \{\tilde{\mathbf{Y}}^{r,s}_{i}i=1\ldots s^2\} \) and finally the locations of the matched patches in \( \mathbf{Y}^r \) and \( \tilde{\mathbf{Y}}^r \) are obtained. The detailed steps of the proposed ISBM method are described as follows.
To simplify the expression, we use the coordinate of the top left pixel of the patch to represent the patch location.

1) Build pixel-wise coordinate mapping between original image and down-sampled images: The mapping function \( f : \mathbb{R}^2 \rightarrow \mathbb{R}^3 \) is defined as \( f(\hat{x}, \hat{y}) = (\hat{x}', \hat{y}', c) \) where

\[
\begin{align*}
\hat{x}' &= \left\lfloor \frac{x - 1}{s} \right\rfloor + 1, \\
\hat{y}' &= \left\lfloor \frac{y - 1}{s} \right\rfloor + 1, \\
c &= \text{mod}(x - 1, s) + \text{mod}(y - 1, s) \times s + 1
\end{align*}
\]

with \( c \in [1, s^2] \), such that the pixel location \((x, y)\) in the original image is mapped to location \((\hat{x}', \hat{y}')\) in the \(c\)-th down-sampled image.

2) Perform block matching on down-sampled images: Given the patch located at \((x_0, y_0)\) in the reference image \( \bar{Y}_r \), we obtain its location \((x_0', y_0')\) in the \(t\)-th down-sampled image \( \bar{Y}_d^{t} \) using (4). The patch with size \( [m/s] \times [m/s] \times n \) at \((x_0', y_0')\) in \( \bar{Y}_d^{t} \) is utilized as the reference patch, and block matching is performed across all the \( s^2 \) down-sampled images \( \{\bar{Y}_i^{d} \}_{i=1}^{s^2} \) of \( \bar{Y}_r \) within a search region of size \( [1/s] \times [1/s] \) centered at \((x_0', y_0')\). The nearest \( K \) similar patches of the reference patch in \( \{\bar{Y}_i^{d} \}_{i=1}^{s^2} \) are obtained with locations \( \{(\hat{x}_i', \hat{y}_i', c_i)\}_{i=1}^{K} \).

3) Map the matched patches to original location: The locations of the nearest \( K \) similar patches in the source image \( \bar{Y}_o \) (and \( \bar{Y}_o' \)) can be computed using the following inverse mapping \( f^{-1} : (\hat{x}', \hat{y}', c) \rightarrow (x, y) \) where

\[
\begin{align*}
x &= (\hat{x}' - 1)s + 1 + \text{mod}(c - 1, s), \\
y &= (\hat{y}' - 1)s + 1 + \left\lceil \frac{c - 1}{s} \right\rceil,
\end{align*}
\]

In practice, at time \( t \), given a reference patch in \( \bar{Y}_t \), the above ISBM method is performed between \( \bar{Y}_t \) and all adjacent (source) images \( \{\bar{Y}_i^{t} \}_{i=t-d}^{t+d} \) such that \((2d + 1)K\) similar patches are obtained. Finally, the nearest \( K \) patches are selected and stacked to construct a tensor of size \( m \times m \times n \times K \). Note that the reference patch is also in the tensor since it is at zero distance from itself.

C. Hybrid tensor ring completion

With ISBM, we obtain a set of partially observed tensors composed of similar patches. It is shown in Fig.1 that the tensor constructed from similar patches can be well approximated as a low-rank tensor. Therefore, the tensor completion...
algorithm can be applied to recover the missing entries of each
tensor. In particular, given a tensor $M \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$
and its observed index set $\Omega$, the recovered full tensor $X$ can be
obtained by solving the following minimization problem
\[
\min_{X} \text{rank}_k(X), \quad \text{s.t.} \quad P \circ X = P \circ M
\]
where
\[
P_{i_1, \ldots, i_N} = \begin{cases} 
1, & \text{if } (i_1, \ldots, i_N) \in \Omega \\
0, & \text{otherwise}
\end{cases}
\]
and $\text{rank}_k(X)$ is a tensor rank of $X$. In this work, we use a
tensor ring decomposition-based model to capture the low
rank property of the tensors. Inspired by Theorem 2, according
to which the TR circular unfolding matrix has a low rank
structure, we use the nuclear norm $\|X_{(k,d)}\|_*$ as a surrogate
for the rank and reformulate the problem as
\[
\min_{X} \sum_{k=1}^{N} \beta_k \|X_{(k,d)}\|_*, \quad \text{s.t.} \quad P \circ M = P \circ X
\]
where $\beta_k$ are weighting parameters and $\lambda$ is the regular-
ization parameter. The minimization problem (8) can be solved with a tensor ring nuclear norm (TRNN) algorithm
using ADMM [31].

The above TRNN-based method has shown good completion
performance in many cases. However, for the patches with low percentage of observed pixels (i.e. 5%) as well as
rich texture, the tensor recovered using a nuclear norm-based
method may be of very low TR rank due to the scarcity of
observed entries. In turn, valuable texture information may
end up being discarded, thereby decreasing its completion
accuracy. To address this problem, in this case we apply an
alternative tensor ring decomposition-based completion
algorithm with fixed TR rank.

According to Definition 1, the cost function for the tensor
ring decomposition-based completion algorithm can be formu-
lated as
\[
\min_{Z_1, \ldots, Z_N} \|P \circ (M - \mathcal{R}(Z_1, \ldots, Z_N))\|_F^2.
\]
The above problem can be solved using tensor ring-based
alternating least squares (TRALS) [35]. Specifically, at each
iteration, the core tensor $Z_k$ is updated by fixing all core
tensors except $Z_k$, and based on Theorem 1, the following
problem is solved
\[
\min_{Z_k} \|P_{[k]} \circ (M_{[k]} - Z_k)\|_F^2.
\]
The final estimated tensor $X$ can be calculated by $X = \mathcal{R}(Z_1, \ldots, Z_N)$.

For each tensor, the usage of TRNN or TRALS is adaptively
selected by measuring the percentage of observed entries and
the texture variation of the tensor, which we calculate by
summing the distances of the $K$ image patches. If the observed
percentage is smaller than a predefined threshold $\tau_p$ and
meanwhile the texture variation is larger than a predefined threshold
$\tau_r$, we apply the TRALS algorithm using [32]. Otherwise,
the TRNN algorithm with [31] is used to complete the tensor. The
entire completion process of the proposed non-local tensor ring
completion (NL-TR) is summarized in Algorithm 1.

Algorithm 1 Non Local Patch-based Tensor Ring completion
(NL-TR)

Input: Partially observed video sequence $\{Y_t\}_{t=1}^T$, block
matching parameters $b, m, \alpha, \delta, l, d$ and $K$, completion
parameters $\tau_v, \tau_p$, and $\beta_k$.

1: Apply image padding on each frame in $\{Y_t\}_{t=1}^T$ and get
the padded video sequence $\{\tilde{Y}_t\}_{t=1}^T$.

2: for $t = 1, 2, \ldots, T$ do

3: Extract reference patches from $\tilde{Y}_t$ and obtain patch sets
$\{P_{i}\}_{i=1}^M$.

4: for $i = 1, 2, \ldots, M$ do

5: Search $K$ nearest patches for $P_i$ with searching range
$l \times l$ in $\{\tilde{Y}_{t}\}_{t=\max(t-d,1)}^{d}$. Stack similar patches and
form a partially observed tensor $M_i$.

6: Compute distance summation $\nu_i$ and missing percentage $\nu_1$ of $M_i$.

7: if $\nu_1 < \tau_p$ and $\nu_1 > \tau_v$ then

8: complete $M_i$ using [32].

9: else

10: complete $M_i$ using [31].

11: end if

12: end for

13: Aggregate all recovered patches belonging to frame $\tilde{Y}_t$.

14: Remove padded border pixels and obtain recovered
frame $\hat{Y}_t$.

15: end for

Output: Recovered sequence $\{\hat{Y}_t\}_{t=1}^T$.

Remark 1. The above NL-TR method can also be used for
image completion by setting the adjacent frame parameter $d = 0$,

i.e., the block matching is only performed in the spatial
domain within the image itself.

Remark 2. Other tensor rank models can be also used with
the proposed method by replacing the tensor ring model. In
particular, the TRNN algorithm can be replaced with other
nuclear norm-based algorithm such as TNN [19] and TT [9].
Similarly, the TRALS algorithm can be replaced with other
decomposition or factorization-based algorithms such as CP
Decomposition [30], PRTC [18], and TCTF [20].

D. Complexity analysis

Assume $m$ and $l$ can be divided by $s$, and all elements in
the TR rank are set to the same value $r$. The complexity of
block matching for each sub-sampled patch with size
$(m/s) \times (m/s) \times n$ with searching range $(l/s) \times (l/s) \times (2d+1)$
is $O(m^2 l^2 (2d+1)/s^2)$. The completion complexities for a
tensor of size $m \times m \times n \times K$ using TRNN and TRALS
are $O(m^2 K \max(m, K)^2)$ and $O(m^2 n K r^4)$, respectively,
where $p \in [0, 1]$ is the percentage of observed entries. Thus,
suppose there exist $M$ tensors in total, in which a fraction
$\alpha$ of the tensors utilize TRNN for completion, the overall
complexity for completion for a frame will be $O(m^2 l^2 (2d+1)/s^2)M + am^2 K \max(m, K)^2 M + (1 - \alpha)pm^2 n K r^4 M$.
It can be observed that using ISBM strategy can reduce the
computational complexity by a factor of $1/s^2$. Increasing
the overlap parameter $o$ will result in a relatively larger $M$ such that the overall complexity will also increase. Moreover, since block matching and tensor completion for each patch are independent, parallel computation can be used to further improve the computational efficiency.

IV. NON-LOCAL PATCH-BASED ONLINE TENSOR COMPLETION

In practice, many video completion applications are of a streaming nature where the frames arrive sequentially. Although one could use NL-TR to complete each frame according to its adjacent frames, only the recovered patches belonging to the reference frame are used. It is desirable that the previous block matching and completion results be shared for new frames to improve efficiency. Therefore, we develop a non-local patch-based online tensor completion framework, where both the block matching and tensor completion steps are processed in an online fashion.

The framework of the proposed online completion algorithm is illustrated in Fig. 4. A first-in-first-out (FIFO) input buffer with fixed-size $L$ is utilized to store the latest $L$ consecutive frames, and a tensor pool is constructed to store the core tensors required to recover the frames in the buffer. At time $t$, the newly arriving frame $Y^t$ is padded and then added to the buffer, meanwhile the oldest frame $Y^{t-L}$ is dropped from the buffer. The patches in $Y^t$ that are similar to patches in $Y^{t-1}$ are tracked, and the previous tensors are updated by adding the corresponding matched similar patches in $Y^t$, and the relevant core tensors in the core tensor pool are updated. Finally, the patches corresponding to frame $Y^{t-L+1}$ are recovered using the corresponding core tensors, and recovered frame $Y^{t-L+1}$ is completed with the recovered patches.

A. Strategies for different types of tensors

During the sequential process, there will be three types of tensors (core tensors), marked in red, blue and green in Fig. 4. As the figure shows, some tensors need to be updated as new similar patches arrive (marked as red). Other tensors should be created to cover those regions that are not covered by the matched patches (marked as blue). To control the growth of the number of tensors, some tensors should stop tracking when the patches they track have significant overlap with other tensors (marked as green). The specific operations for these different types of tensor are expressed as follows:

1) Initialize and update tensor using new frame: At time $t \geq 2$, suppose we are given the tracking patches of $Y^{t-1}$. As a new frame $Y^t$ arrives, for each tracking patch of $Y^{t-1}$, the ISBM method is applied to find the $K$ nearest similar patches in $Y^t$, and the first similar patch (i.e., having the smallest distance) is designated as a tracking patch of $Y^t$. Then, the tensors are formed by stacking all similar patches in the image set $\{Y^t\}_{i=\max(t-L+1, 1)}^{t}$. Since the tensors and their corresponding core tensors have already been formed at $t-1$, we can simply update the tensors by adding the new similar patches in $Y_t$ and removing patches corresponding to $Y^{t-L}$ if $t-L > 0$, and then update the corresponding core tensors using an online tensor completion algorithm developed in the next section.

2) Create new tensor for the uncovered regions: For the regions in the new frame $Y^t$ that are not covered by patches matched from $Y^{t-1}$, we create new overlapping patches of size $m \times m \times n$ with overlap pixels $o$ to cover them. These new patches are added to the tracking patches of $Y^t$. Then, interval sampling and block matching are applied to find the $K$ nearest similar patches in $Y^t$ and the tensors are created by stacking similar patches. Finally, the TRALS algorithm in (9) is applied to obtain the core tensors, and the new core tensors are then added to the core tensor pool. The first frame $Y^1$ can be treated as a frame with an entirely uncovered region, such that the above procedure can be applied to initialize the tracking patches, tensors and core tensors.

3) Prune overlapping tensors: It can be also observed that tracking patches from different tensors may have large overlap after several tracking iterations. Continuing to update all such tensors will result in no or marginal improvement in completion performance at the expense of extra computational and storage cost. Thus, we stop the growth of the number of tensors by detecting their degree of overlap. Specifically, at time $t$, the overlap degree of a tensor is measured by counting the number of times each pixel is shared with other tensors. If the minimum number is larger than a predefined threshold $\tau$, the tensor will not be tracked and updated.

B. Online tensor ring completion algorithm

In this section, we develop an online tensor completion algorithm. We adopt a tensor ring decomposition method to formulate the online tensor completion, since the core tensors store the information of the tensor and can be updated iteratively. Specifically, it can be observed from (9) that only the last core tensor $Z_N$ incorporates temporal information of $M$ (i.e., related to the temporal dimension). Thus, modeling $M \in \mathbb{R}^{1 \times \ldots \times I_N}$ as a sequence of $(N-1)$-way tensors $M_1, \ldots, M_{I_N} \in \mathbb{R}^{1 \times \ldots \times I_{N-1}}$, and denoting the lateral slice $Z_N(:, i, :)^t$ as a tensor $Z_N^{(t)} \in \mathbb{R}^{r_1 \times 1 \times r_1}$, and , at time $t$, the tensor ring completion problem in (9) can be rewritten as

$$\min_{Z_1, \ldots, Z_N} \sum_{i=1}^{I_N} \left| \mathcal{P}_i \circ (M_i - R(Z_1^{(t)}, \ldots, Z_{N-1}^{(t)}, Z_N^{(t)})) \right|^2_F$$  \hspace{1cm} (11)

To improve robustness, similar to (12), we add a regularization term to constrain the norm of each $Z_N^{(t)}$. Therefore, (11) can be further modified as

$$\min_{Z_1, \ldots, Z_N} \sum_{i=1}^{I_N} \left| \mathcal{P}_i \circ (M_i - R(Z_1^{(t)}, \ldots, Z_{N-1}^{(t)}, Z_N^{(t)})) \right|^2_F + \gamma \|Z_N^{(t)}\|_F^2$$ \hspace{1cm} (12)

Now we present our online tensor completion algorithm solving (12). Suppose at time $t$, we have the tensor $M^t$ and the previous estimated core tensors $\{Z_i^{t-1}\}_{i=1}^{N}$. First, we compute $\{Z_N^{(t)}\}_{i=1}^{N}$. Rather than updating all tensors $Z_N^{(t)}$, we use their previous estimates such that

$$Z_N^{(t)} = Z_N^{(t+K), t-1}, i = 1, \ldots, I_N - K$$ \hspace{1cm} (13)
and only compute the last $K$ tensors \( \{Z^{(i),t}_{N-1}\}_{i=I_{N}-K+1} \). Specifically, for each $i \in [I_{N}-K+1, I_{N}]$, the vectorized form $Z^{(i),t}_{N}$ of $Z^{(i),t}_{N}$ can be obtained by solving

$$
Z^{(i),t}_{N} = \arg \min_{z \in \mathbb{R}^{N \times r_{N}}} \| p_{i}^{T} \circ (m_{i}^{t} - U^{t-1}z) \|^2_{2} + \gamma \|z\|^2_{2} \tag{14}
$$

where $U^{t-1} = (Z^{t-1})^{[2]'}$, $p_{i}$ and $m_{i}^{t}$ are the vectorizations of $P_{i}$ and $M_{i}^{t}$, respectively. Eq. \((14)\) can be further simplified as

$$
Z^{(i),t}_{N} = \arg \min_{z \in \mathbb{R}^{N \times r_{N}}} \| m_{i}^{t} - U^{t-1}z \|^2_{2} + \gamma \|z\|^2_{2} \tag{15}
$$

where $I_{i}$ denotes the index set of the non-zero entries of $p_{i}$. $m_{i}^{t} \in \mathbb{R}^{I_{i} \times r_{N}}$ denotes the vector of elements in $m$ indexed by set $I_{i}$. The matrix $U_{I_{i}} \in \mathbb{R}^{I_{i} \times \left| I_{i} \times r_{N} \right|}$ is formed from the rows of $U$ with row index $I_{i}$, $|Z|$ is the cardinality of set $I$. Eq. \((15)\) has a closed-form solution

$$
Z^{(i),t}_{N} = \left( (U_{i}^{t-1})^{T}U_{I_{i}}^{t-1} + \gamma I \right)^{-1}(U_{i}^{t-1})^{T}m_{i}^{t}. \tag{16}
$$

After computing $Z^{(i),t}_{N}$ using \((13)\) and \((16)\), we obtain $Z^{t}_{N}$. Then, keeping $Z^{t}_{N}$ fixed, the remaining tensor cores $Z^{t}_{1}, \ldots, Z^{t}_{N-1}$ are alternatively solved by

$$
\min_{Z_{k}^{t}} \left\| P_{[k]}^{t} \circ (M_{[k]}^{t} - Z_{[k]}^{t} \left( Z_{[k]}^{t} \right)^{T} ) \right\|_{F}^{2} \tag{17}
$$

which has a similar procedure as TRALS. With the warm start using the previously estimated core tensors, and only updating the first $N-1$ core tensors using TRALS, the algorithm has lower computational cost and faster convergence than directly solving \((1)\).

The entire completion process of the proposed non-local online tensor ring completion (NL-TR-Online) is summarized in Algorithm 2.

**Algorithm 2 Non Local Patch-based Tensor completion-Online version (NL-TR-Online)**

**Input:** Partially observed video sequence $\{Y^{t}\}_{t=1}^{T}$, buffer size $L$, block matching parameters $b, m, o, s, l$ and $K$, tensor ring rank $\{r_{k}\}_{k=1}^{N}$, regularization parameter $\gamma$, overlap threshold $\tau_{c}$.

1. Initialize tracking patch set $I_{P}$, tensor set $I_{M}$, core tensor set $I_{Z}$ and overlap degree set $I_{c}$ as $\emptyset$.
2. for $t = 1, \ldots, T$ do
3. Pad the frame $Y^{t}$ and get $\tilde{Y}^{t}$.
4. for each patch $P_{i}$ in $I_{P}$ do
5. if overlap degree $c_{i} < \tau_{c}$ then
6. Search $K$ nearest patches of $P_{i}$ within search range $l \times l$ in $Y^{t}$. Replace $P_{i}$ in $I_{P}$ with its nearest similar patch in $\tilde{Y}^{t}$. Update corresponding tensor in $I_{M}$ with $K$ newly matched patches.
7. Compute $Z_{k}^{N}$ using \((13)\) and \((16)\).
8. Update $\{Z_{k}\}_{k=1}^{N-1}$ using \((17)\).
9. end if
10. end for
11. Create new tracking patches to cover the uncovered region in $\tilde{Y}^{t}$ and add them to $I_{P}$.
12. Search $K$ nearest patches for each new tracking patch within search range $l \times l$ in $Y_{t}$, create tensors with similar patches and include them in $I_{M}$.
13. Initialize core tensors for each new tensor using \((9)\) and add them to $I_{Z}$.
14. Compute overlap degree set $\{c\}$ according to $I_{P}$.
15. Remove elements in $I_{P}$, $I_{M}$, $I_{Z}$ and $I_{c}$ that are unrelated to $\{Y^{t}\}_{t=L}^{T}$.
16. Aggregate all the recovered patches belonging to $\tilde{Y}^{t-L+1}$.
17. Remove padded border pixels and output recovered frame $\tilde{Y}^{t-L+1}$.
18. end for

**Output:** Recovered sequence $\{\tilde{Y}^{t}\}_{t=1}^{T}$
C. Complexity analysis

For online NL-TR, when each frame arrives, block matching is only performed between the latest two frames, thus the complexity for block matching is $O(pn^2T/s^2)$. The time complexity for online tensor ring completion with size $m \times m \times n \times K$ is $O(pm^2n T^2 / s^2)$. Suppose the total number of tensors for the recovered frame is $T$, with a number of tensors to be updated and created $T_1$ and $T_2$, respectively. Then, the overall complexity for completing the frame is $O(pm^2n T^2 / s^2 + pm^2n K r^2 T_1 + pm^2n K r^2 T_2)$. Given that $K$ for NL-TR-Online is always set smaller than that for NL-TR, and $T_2$ is usually much smaller than $T_1$, the overall completion cost of each frame using NL-TR-Online will be smaller than NL-TR. Similar to NL-TR, parallel computation can also be applied to both the block matching and tensor completion steps to improve the computational efficiency.

V. EXPERIMENTAL RESULTS

In this section, we conduct experiments to verify the performance of the proposed methods and compare to existing non-local patch-based tensor completion algorithms, including NL-TNN [28] and NL-TT [29]. We also compare the performance to traditional global tensor completion algorithms, including the matrix factorization-based algorithm TMAC [18], tensor ring-based algorithms TRNN [31] and PRTC [32], and tensor SVD-based algorithms TNN [19] and TCTF [20].

The completion performance is evaluated by the average value of the peak signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM) over 20 Monte Carlo runs with different missing entries and noise realizations. In the experiments, the patch size for all patch-based algorithms is set to $m = 36$ unless stated otherwise. The search size $l$, interval sampling parameter $s$, padding pixels $b$ and nearest number of neighbors $K$ for the proposed methods are set to 20, 2, 20 and 50, respectively. The number of overlapping pixels $o$ for Algorithm 1 and 2 are set to 8 and 16. The number of iterations for TRNN and TRALS are set to 10 and 5, respectively. For Algorithm 1, the thresholds $\tau_p$ and $\tau_r$ for hybrid completion are set to 0.2 and 100, respectively. For the other algorithms, the parameters are adjusted so as to achieve their best performance. Further, the parameters are fixed during each simulation.

A. Image inpainting

In this part, we verify the performance of the proposed algorithm NL-TR on the image inpainting task using the Berkeley Segment Dataset [38] as the test data. We randomly select 30 images from the dataset and reshape their size to $320 \times 480$. For each image, $p \times 100\%$ of the pixels are randomly and uniformly selected as the observed data. The image inpainting problem is then formulated as a $320 \times 480 \times 3$ tensor completion task. The observation level $p$ is chosen from 0.05, 0.1 and 0.2, and the tensor rank of completion using TRALS in NL-TR is accordingly set to 4, 5 and 6, respectively. In this part, $d$ is set to 0 for NL-TR, i.e., only performing block matching in the spatial domain.

First, we compare the completion performance of all algorithms on all 30 test images. Fig. 8 shows the average PSNR and SSIM for the test images with $p = 0.1$. As shown, our proposed method achieves the best performance and NL-TNN comes second. The results verify that the non-local patch-based method can improve performance upon methods that treat the entire image as a tensor.

Second, we investigate the performance of the algorithms with different observation levels, and demonstrate the performance gains due to image dilation and hybrid completion. Table 1 shows the completion performance for each algorithm for different values of $p$ for the five randomly selected images. NL-TRNN denotes NL-TR with $\tau_p = 0$ (i.e., only TRNN is used for tensor completion), NL-TRALS denotes NL-TR with $\tau_p = 1$ and $\tau_r = 0$ (i.e., only TRALS is used for completion), NL-TR(ND) denotes NL-TR with no image dilation in the block matching step. As can be seen, NL-TR achieves the overall best performance for all $p$. Moreover, for the very low observability regime with $p = 0.05$, TRNN may not work very well for tensor completion, and the performance of block matching only using the observed pixels is rather poor. When the number of observed pixels is relatively large (i.e., $p = 0.2$), TRNN has better performance, and so does NL-TR which only uses TRNN for completion as we set $\tau_p = 0.2$. An example of recovered images with $p = 0.1$ is also provided in Fig. 9. As can be seen, the completed image using the proposed method has the best visual performance. It also shows that NL-TNN suffers from color inconsistencies in some pixels due to misguidance from wrong interpolation.

B. Video completion

We also evaluate the completion performance using color video sequences from the YUV dataset[1]. Four video sequences are selected as Fig. 10 shows. For each video, a sequence of 50 frames is selected, each frame with size $288 \times 352$ such that the size of the whole tensor is $288 \times 352 \times 3 \times 50$.

To evaluate the completion performance, for each tensor, 10% of the pixels are randomly and uniformly selected as the observed data. The tensor ranks of completion using TRALS in the proposed algorithms are all set to 5. The adjacent frame parameter $d$ for NL-TR is set to 2, while the buffer size for NL-TR-Online is set to 5. For NL-TT, the video is divided into 10 parts with size $288 \times 352 \times 3 \times 5$, and NL-TT is performed on each part. For PRTC and TRNN which favor high order tensors for better performance, we reshape the tensor to an 11-order tensor of size $2 \times 4 \times 4 \times 9 \times 2 \times 4 \times 11 \times 3 \times 5 \times 10$. For TNN and TCTF which only afford 3-order tensor completion, we reshape the tensor to a 3-order tensor of size $288 \times 352 \times 150$. To verify the performance improvement of NL-TR using temporal information, an additional algorithm NL-TR(Single) with $d = 0$ is also included for comparison.

Fig. 11 shows the curves of average PSNR and SSIM for each frame using different algorithms. It can be observed that the proposed NL-TR-Online achieves the best overall performance. NL-TR achieve the second best for the first three videos. The superior performance of these two algorithms over

1http://trace.eas.asu.edu/yuv/index.html
Fig. 5. Comparison of the average PSNR (top) and SSIM (bottom) for different algorithms on 30 selected test images with $p = 0.1$.

TABLE I

| Image    | $p$   | Metric | TMAC | TNN | TCF | TRNN | PTRC | NL-TNN | NL-TT | NL-TRNN | NL-TRD | NL-TR(ND) | NL-TR |
|----------|-------|--------|------|-----|-----|------|------|--------|-------|---------|--------|-----------|-------|
| Cruise   | 0.05  | PSNR   | 12.19| 16.95| 11.57| 16.05| 19.09| 17.34  | 14.69  | 18.50   | 20.65  | 16.30     | 20.79 |
|          |       | SSIM   | 0.2850| 0.4282| 0.1497| 0.4657| 0.6333| 0.5839 | 0.4455 | 0.6754  | 0.5724 | 0.5544    | 0.7514 |
|          | 0.1   | PSNR   | 17.83| 19.30| 16.32| 18.31| 21.07| 18.87  | 19.06  | 21.55   | 23.24  | 22.05     | 23.22 |
|          |       | SSIM   | 0.5659| 0.6190| 0.4445| 0.6197| 0.7289| 0.6787 | 0.6931 | 0.8034  | 0.8442 | 0.7898    | 0.8443 |
|          | 0.2   | PSNR   | 19.95| 22.64| 19.11| 21.47| 22.89| 20.89  | 22.72  | 25.27   | 25.81  | 24.47     | 25.27 |
|          |       | SSIM   | 0.6821| 0.7913| 0.6206| 0.7879| 0.8109| 0.7842 | 0.8458 | 0.9105  | 0.9012 | 0.8995    | 0.9105 |
| Bamboo   | 0.05  | PSNR   | 13.86| 17.18| 11.99| 17.24| 21.19| 22.16  | 16.50  | 20.93   | 22.90  | 18.69     | 23.09 |
|          |       | SSIM   | 0.1895| 0.2191| 0.0890| 0.5209| 0.6506| 0.7515 | 0.5535 | 0.7164  | 0.7684 | 0.6346    | 0.7761 |
|          | 0.1   | PSNR   | 20.04| 19.91| 18.80| 20.16| 23.19| 23.38  | 21.97  | 23.83   | 25.00  | 23.97     | 25.07 |
|          |       | SSIM   | 0.5841| 0.4948| 0.4321| 0.6626| 0.7608| 0.8022 | 0.7562 | 0.8161  | 0.8309 | 0.8046    | 0.8409 |
|          | 0.2   | PSNR   | 20.97| 23.37| 20.23| 23.46| 24.81| 25.24  | 25.73  | 27.14   | 27.23  | 26.40     | 27.14 |
|          |       | SSIM   | 0.6587| 0.7076| 0.5763| 0.7993| 0.8260| 0.8654 | 0.8744 | 0.9069  | 0.8928 | 0.8955    | 0.9069 |
| Farmyard | 0.05  | PSNR   | 14.28| 18.65| 12.15| 18.76| 18.66| 22.08  | 17.54  | 21.57   | 23.33  | 19.29     | 23.45 |
|          |       | SSIM   | 0.4205| 0.4621| 0.1179| 0.6314| 0.6629| 0.7208 | 0.6236 | 0.7352  | 0.7639 | 0.6696    | 0.7686 |
|          | 0.1   | PSNR   | 20.67| 21.01| 19.54| 21.47| 23.23| 23.21  | 22.53  | 24.15   | 25.06  | 24.41     | 25.03 |
|          |       | SSIM   | 0.5077| 0.5815| 0.5720| 0.7093| 0.7515| 0.6761 | 0.7517 | 0.8029  | 0.7852 | 0.8033    | 0.8260 |
|          | 0.2   | PSNR   | 22.83| 23.94| 22.14| 24.16| 25.30| 24.83  | 25.44  | 27.19   | 27.76  | 26.74     | 27.19 |
|          |       | SSIM   | 0.7440| 0.7297| 0.7015| 0.8129| 0.8288| 0.8323 | 0.8534 | 0.9039  | 0.8744 | 0.8962    | 0.9039 |
| Desert   | 0.05  | PSNR   | 16.88| 23.64| 15.68| 24.23| 24.33| 26.20  | 23.09  | 27.07   | 29.86  | 25.03     | 30.19 |
|          |       | SSIM   | 0.4246| 0.6330| 0.1539| 0.7075| 0.7172| 0.7849 | 0.6968 | 0.8384  | 0.5931 | 0.7664    | 0.9034 |
|          | 0.1   | PSNR   | 24.70| 25.45| 23.73| 25.95| 28.20| 27.85  | 29.69  | 30.93   | 32.62  | 30.17     | 32.85 |
|          |       | SSIM   | 0.7081| 0.7337| 0.6567| 0.7754| 0.8455| 0.8417 | 0.9018 | 0.9214  | 0.9460 | 0.9043    | 0.9483 |
|          | 0.2   | PSNR   | 26.51| 28.24| 25.71| 28.55| 29.92| 30.02  | 34.04  | 35.48   | 35.17  | 35.15     | 35.48 |
|          |       | SSIM   | 0.7957| 0.8488| 0.7547| 0.8652| 0.8936| 0.9010 | 0.9620 | 0.9711  | 0.9686 | 0.9692    | 0.9711 |

NL-TR(Single) verify the performance gain due to exploiting the temporal correlation. To further analyze the effect of the motion degree on the completion performance, we show the the average magnitude of the optical flow between frames from each video, and the eigenvalues of the matrix unfolded from the video tensor along the time dimension. Among the four videos, one can see that ‘akiyo’, which has the minimum motion degree (i.e., can be well approximated by a low-rank tensor), has the best overall completion performance for all algorithms. It can also be seen that the completion performance of the traditional completion algorithms are tightly dependent on the degree of motion between the frames. Specifically, for these algorithms, a relatively lower motion degree may result in higher completion performance. Further, as Fig. 1 shows, for the non-local patch-based algorithms, the tensor completion performance is improved by temporal consistency within the similar patches, thus the performance is not highly affected by the variance in motion degree.

Further, we investigate the performance of NL-TR and NL-TR-Online with different block matching parameters. The experiment is carried out on the bus video with 10% observation percentage. Fig. 10-12 depict the average PSNR of the entire recovered video for different parameters $a$, $m$ and $s$, along with the time cost of the block matching and tensor completion steps. In the figures, the suffix ‘BM’ denotes the block matching procedure, while ‘CA ’ denotes the completion and aggregation procedure. One can observe that both NL-TR and NL-TR-Online can work well with a wide range of param-
The completion time cost of NL-TR-Online is always lower than that of NL-TR, showing the advantage of online completion with regard to efficiency. Specifically, increasing the degree of overlap can improve the performance of NL-TR but at the expense of significantly higher computational complexity, whereas the NL-TR-Online is not sensitive to the selection of overlap because of the growth control of the number of patches. Further, Fig. 12 demonstrates the significant reduction in time cost using the proposed interval sampling-based block matching method.

VI. CONCLUSION

We proposed a new non-local patch-based tensor ring completion algorithm NL-TR, which exploits the high correlation between patches to improve the completion performance. To improve the block matching efficiency, an interval sampling method is proposed. A hybrid completion strategy using TRNN and TRALS is proposed to deal with patches with different degrees of texture variation. Further, a novel non-local patch-based online tensor completion algorithm NL-TR-Online is proposed. Equipped with a patch tracking method and an online tensor completion algorithm, the proposed online algorithm can obtain a lower computational complexity, as well as high completion accuracy. Experimental results demonstrate the superior performance of the proposed algorithm compared to existing state-of-the-art methods.

REFERENCES

[1] N. Cohen, O. Sharir, and A. Shashua, “On the expressive power of deep learning: A tensor analysis,” in Conference on learning theory. PMLR, 2016, pp. 698–728.
[2] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, “Tensor decomposition for signal processing and machine learning,” IEEE Transactions on Signal Processing, vol. 65, no. 13, pp. 3551–3582, 2017.
[3] K. Makantasis, A. D. Doulamis, N. D. Doulamis, and A. Nikitakis, “Tensor-based classification models for hyperspectral data analysis,” IEEE Transactions on Geoscience and Remote Sensing, vol. 56, no. 12, pp. 6884–6898, 2018.
[4] J. Kossaifi, Z. C. Lipton, A. Kolbeinsson, A. Khanna, T. Furlanello, and A. Anandkumar, “Tensor regression networks,” Journal of Machine Learning Research, vol. 21, pp. 1–21, 2020.
[5] M. Rupp and S. Schwarz, “A tensor lms algorithm,” in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 3347–3351.
[6] M. Boussé, O. Debals, and L. De Lathauwer, “Tensor-based large-scale blind system identification using segmentation,” IEEE Transactions on Signal Processing, vol. 65, no. 21, pp. 5770–5784, 2017.
[7] G. T. de Araújo, A. L. de Almeida, and R. Boyer, “Channel estimation for intelligent reflecting surface assisted mimo systems: A tensor modeling approach,” IEEE Journal of Selected Topics in Signal Processing, vol. 15, no. 3, pp. 789–802, 2021.
[8] A. Sobral, S. Javed, S. Ki Jung, T. Bouwmans, and E.-h. Zahzah, “Online stochastic tensor decomposition for background subtraction in multispectral video sequences,” in Proceedings of the IEEE International Conference on Computer Vision Workshops, 2015, pp. 106–113.
[9] J. A. Bengua, H. N. Phien, H. D. Tuan, and M. N. Do, “Efficient tensor completion for color image and video recovery: Low-rank tensor train,” IEEE Transactions on Image Processing, vol. 26, no. 5, pp. 2466–2479, 2017.
[10] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, L.-J. Deng, and Y. Wang, “A novel tensor-based video rain streaks removal approach via utilizing discriminatively intrinsic priors,” in Proceedings of the ieee conference on computer vision and pattern recognition, 2017, pp. 4057–4066.
[11] W. Li, X. Zhu, and S. Gong, “Harmonious attention network for person re-identification,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 2285–2294.
[12] S. Gandy, B. Recht, and I. Yamada, “Tensor completion and low-n-rank tensor recovery via convex optimization,” Inverse Problems, vol. 27, no. 2, p. 025010, 2011.
[13] 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.
Fig. 8. Curves of average PSNR (top) and SSIM (bottom) for each frame using different algorithms on four video sequences.

Fig. 9. Example of recovered frames corresponding to top row of Fig. Best viewed in ×2 sized pdf file.

Fig. 10. Left: curves of average PSNR with different overlap $o$. Right: average time cost with different $o$.

Fig. 11. Left: curves of average PSNR with different patch size $m$; Right: average time cost with different $m$.

[14] H. A. Kiers, “Towards a standardized notation and terminology in multi-way analysis,” *Journal of Chemometrics: A Journal of the Chemometrics Society*, vol. 14, no. 3, pp. 105–122, 2000.

[15] P. Jain and S. Oh, “Provable tensor factorization with missing data,” in *Advances in Neural Information Processing Systems*, 2014, pp. 1431–1439.
Fig. 12. Left: curves of average PSNR with different interval $s$; Right: average time cost with different $s$.

[16] L. R. Tucker, “Some mathematical notes on three-mode factor analysis,” *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[17] H. Kasai and B. Mishra, “Low-rank tensor completion: a riemannian manifold preconditioning approach,” in *International Conference on Machine Learning*, 2016, pp. 1012–1021.

[18] Y. Xu, R. Hao, W. Yin, and Z. Su, “Parallel matrix factorization for low-rank tensor completion,” *Inverse Problems and Imaging*, vol. 9, no. 2, pp. 601–624, 2015.

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

[20] P. Zhou, C. Lu, Z. Lin, and C. Zhang, “Tensor factorization for low-rank tensor completion,” *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1152–1163, 2017.

[21] X.-Y. Liu, S. Aeron, V. Aggarwal, and X. Wang, “Low-tubal-rank tensor completion using alternating minimization,” *IEEE Transactions on Information Theory*, vol. 66, no. 3, pp. 1714–1737, 2019.

[22] 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.

[23] 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,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 9151–9158.

[24] H. Huang, Y. Liu, J. Liu, and C. Zhu, “Provable tensor ring completion,” *Signal Processing*, vol. 171, p. 107486, 2020.

[25] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, “Image denoising by sparse 3-d transform-domain collaborative filtering,” *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[26] M. Maggioni, V. Katkovnik, K. Egiazarian, and A. Foi, “Nonlocal transform-domain filter for volumetric data denoising and reconstruction,” *IEEE Transactions on Image Processing*, vol. 22, no. 1, pp. 119–133, 2012.

[27] B. Wen, Y. Li, L. Pfister, and Y. Bresler, “Joint adaptive sparsity and low-rankness on the fly: an online tensor reconstruction scheme for video denoising,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 241–250.

[28] L. Zhang, L. Song, B. Du, and Y. Zhang, “Nonlocal low-rank tensor completion for visual data,” *IEEE Transactions on Cybernetics*, 2019.

[29] D. Meng, H. Tingzh, Z. Xile, K. N. Michael, and M. Tianhui, “Tensor train rank minimization with nonlocal self-similarity for tensor completion,” *Inverse Problems and Imaging*, vol. 15, no. 3, pp. 475–498, 2021.

[30] Q. Zhao, G. Zhou, S. Xie, L. Zhang, and A. Cichocki, “Tensor ring decomposition,” arXiv preprint arXiv:1606.05535, 2016.

[31] J. Yu, C. Li, Q. Zhao, and G. Zhao, “Tensor-ring nuclear norm minimization and application for visual data completion,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3142–3146.

[32] J. Yu, G. Zhou, C. Li, Q. Zhao, and S. Xie, “Low tensor-ring rank completion by parallel matrix factorization,” *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[33] Y.-W. Huang, C.-Y. Chen, C.-H. Tsai, C.-F. Shen, and L.-G. Chen, “Survey on block matching motion estimation algorithms and architectures with new results,” *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology*, vol. 42, no. 3, pp. 297–320, 2006.

[34] J. Hussain and Z. Ahmed, “A survey on video compression fast block matching algorithms,” *Neurocomputing*, vol. 355, pp. 215–237, 2019.

[35] L. Yuan, J. Cao, X. Zhao, Q. Wu, and Q. Zhao, “Higher-dimension tensor completion via low-rank tensor ring decomposition,” in *Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2018, pp. 1071–1076.

[36] 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.

[37] S. Smith, K. Huang, N. D. Sidiropoulos, and G. Karypis, “Streaming tensor factorization for infinite data sources,” in *Proceedings of the SIAM International Conference on Data Mining*, 2018, pp. 81–89.

[38] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in *Proc. 8th Int’l Conf. Computer Vision*, vol. 2, July 2001, pp. 416–423.