Online Robust Low-Rank Tensor Modeling for Streaming Data Analysis

Ping Li, Jiashi Feng, Xiaojie Jin, Luming Zhang, Xianghua Xu, and Shuicheng Yan, *Fellow, IEEE*

***Abstract*— Tensor data (i.e., the data having multiple dimensions) are quickly growing in scale in many practical applications, which poses new challenges for data modeling and analysis approaches, such as high-order relations of large complexity, gross noise, and varying data scale. Existing low-rank data analysis methods, which are effective at analyzing matrix data, may fail in the regime of tensor data due to these challenges. A robust and scalable low-rank tensor modeling method is heavily desired. In this paper, we develop an online robust low-rank tensor modeling (ORLTM) method to address these challenges. The ORLTM method leverages the high-order correlations among all tensor modes to model an intrinsic low-rank structure of streaming tensor data online and can effectively analyze data residing in a mixture of multiple subspaces by virtue of dictionary learning. ORLTM consumes a very limited memory space that remains constant regardless of the increase of tensor data size, which facilitates processing tensor data at a large scale. More concretely, it models each mode unfolding of streaming tensor data using the bilinear formulation of tensor nuclear norms. With this reformulation, ORLTM employs a stochastic optimization algorithm to learn the tensor low-rank structure alternatively for online updating. To capture the final tensors, ORLTM uses an average pooling operation on folded tensors in all modes. We also provide the analysis regarding computational complexity, memory cost, and convergence. Moreover, we extend ORLTM to the image alignment scenario by incorporating the geometrical transformations and linearizing the constraints. Extensive empirical studies on synthetic database and three practical vision tasks, including video background subtraction, image alignment, and visual tracking, have demonstrated the superiority of the proposed method.**

***Index Terms*— Background subtraction, image alignment, low-rank tensor, object tracking, online learning, robust recovery.**

I. Introduction

EFFICIENT data analysis methods are highly demanded due to the fast growth of multidimensional data in practical applications. During the last decade, low-rank tensor modeling methods [1], [2] have attracted a lot of interest since these methods can reveal intrinsic structures while consolidating knowledge learnable from the data. In addition, unlike the ones formatting tensors into matrices and employing low-rank matrix methods, low-rank tensor modeling methods directly process raw multidimensional data without destroying tensor structures and thus demonstrate a great potential in many multidimensional real-world applications, including video analysis, weather forecast data analysis, and multispectral image processing, to name a few.

However, a large number of tensor data often arrive continuously in a dynamic environment, such as video in public surveillance systems. This brings the following new challenges to low-rank tensor modeling methods: 1) how to devise scalable approaches for efficiently processing dynamic tensor data in a large size and 2) how to handle data contaminated by malicious outliers or gross noise.

In recent years, two *batch*-based robust tensor methods were developed to deal with noisy tensors, including high-order robust principal component analysis (HORPCA) [1] and tensor RPCA [2]. These two methods generalize RPCA [3] from matrix cases to tensors for learning the inherent low-rank structure by solving $\min_{\mathcal{Z},\mathcal{E}} \|\mathcal{Z}\|_*+\|\mathcal{E}\|_1$ under the constraint $\mathcal{X} = \mathcal{Z} + \mathcal{E}$. While those methods can handle noisy data, they suffer from two drawbacks: 1) they do not scale well to larger data because of heavy memory overheads which will become computationally unaffordable when the data size scales up and 2) their output a low-rank tensor is usually static which prevents them from capturing data dynamics, thus giving inferior performance.

To address these issues, we develop an online robust low-rank tensor modeling (ORLTM) approach that can sequentially learn low-rank tensor structures of noisy data, offering robustness to gross noise contained in the data. Different from those methods that must perform a data analysis in batch, ORLTM bifactorizes the low-rank component in each mode m of the tensor, i.e., $\mathbf{Z}^m = \mathbf{W}^m\mathbf{R}^m$. Here, $\mathbf{W}^m$ encodes the basis of low-rank subspace, while $\mathbf{R}^m$ denotes its tensor coefficient representation. One appealing feature of such reformulation is that the basis and data representations become decoupled.

Manuscript received September 18, 2017; revised April 17, 2018 and July 5, 2018; accepted July 25, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61502131, Grant 61572169, and Grant 61472266, in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LY18F020056, in part by the Open Project Program of the State Key Lab of CAD&CG under Grant A1802, Zhejiang University, in part by the National University of Singapore (Suzhou) Research Institute, Suzhou 215123, China, and in part by the Fundamental Research Funds for the Central Universities of China. The work of J. Feng was supported in part by NUS IDS under Grant R-263-000-C67-646, in part by ECRA under Grant R-263-000-C87-133, and in part by MOE under Grant MOE Tier-II R-263-000-D17-112. *(Corresponding author: Ping Li.)*

P. Li is with the School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China, and also with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China (e-mail: patriclouis.lee@gmail.com).

J. Feng and X. Jin are with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 119077 (e-mail: elefjia@nus.edu.sg; xiaojie.jin@u.nus.edu).

L. Zhang is with the College of Computer Science, Zhejiang University, Hangzhou 310027, China (e-mail: zglumg@gmail.com).

X. Xu is with the School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: xhxu@hdu.edu.cn).

S. Yan is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 119077, and also with the Qihoo 360 Artificial Intelligence Institute, Beijing 100016, China (e-mail: eleyans@nus.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2018.2860964

In this way, the basis usually in small size only requires very limited memory cost and data representations are updated dynamically. Hence, instead of calculating the nuclear norm (also known as trace norm) of the whole unfolding matrix $\mathbf{Z}^m$, ORLTM can significantly reduce memory cost. Such equivalent bifactor reformulation makes online processing possible on streaming data for low-rank tensor modeling with favorable scalability and adaptivity to the data dynamics.

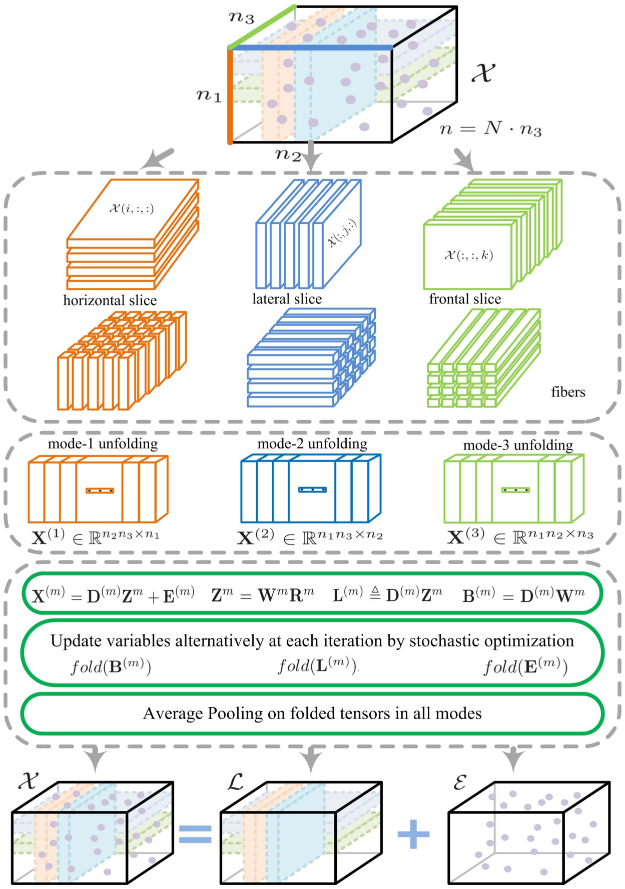
Typically, existing tensor RPCA methods make an assumption that tensor data populate in a *single* low-rank data subspace. Nevertheless, it might be not true for the practical data characterizing more sophisticated structures, e.g., those sampled from a mixed data space consisting of several subspaces. In this situation, both the vanilla ORLTM and previous approaches are unable to yield satisfying performances. Therefore, a dictionary learning module is further introduced so as to enhance ORLTM. With the learned dictionary, a more flexible basis set is available to encode the low-rank component of complicated tensor data. This allows ORLTM to learn complex low-rank structures, such as mixed structure of several low-rank subspaces, thus better modeling tensor data.

An overview of our proposed ORLTM method (with the dictionary learning part) is shown in Fig. 1. With ORLTM, each unfolding $\mathbf{X}^{(m)}$ (recall that m denotes tensor mode and the superscript m in parentheses denotes tensor unfolding matrix while that without parentheses denotes traditional matrix variable) is decomposed into one low-rank component revealing underlying low-rank subspace and one sparse component encoding noise or corruption. It is able to learn the low-rank component $\mathbf{Z}^m$ given a prior dictionary $\mathbf{D}^{(m)}$ (e.g., the data unfolding itself) through minimizing its nuclear norm $\|\mathbf{Z}^m\|_*$. As mentioned earlier, this convex formulation fails to offer a sequential way for data processing, and ORLTM employs its equivalent bilinear factorization form $\mathbf{Z}^m = \mathbf{W}^m\mathbf{R}^m$ to deal with this issue. To better model data dynamics, ORLTM further introduces an auxiliary variable $\mathbf{B}^{(m)}$ for $\mathbf{D}^{(m)}\mathbf{W}^m$ as *reinforced basis dictionary* to be learned. The reformulated problem can be solved via stochastic optimization. Thereafter, to recover the final tensors, ORLTM uses the average pooling operation in all modes of folded tensors.

Generally speaking, ORLTM can be widely applied to analyzing dynamic tensor data in a large size. This paper specifies its application in three real-world streaming data analysis tasks, including video background subtraction, image alignment, and visual tracking. For the foremost one, ORLTM employs the low-rank component $\mathcal{L}$ to model the background of a given video and the sparse component $\mathcal{E}$ to model the foreground (e.g., moving pedestrians and objects). For the latter two, ORLTM aims to capture the correct geometrical transformations so as to align images or track target objects.

This paper presents an extension of previous conference version [4] in terms of proposing an adaptive ORLTM for image alignment (ORLTM-IA) and visual object tracking. In short, we make the following contributions.

1) An ORLTM approach is developed to better model the low-rank structure of the tensor even in the presence of noise or malicious corruptions.



n_3
n_1
n_2
$\mathcal{X}$
$n = N \cdot n_3$
$\mathcal{X}(i,:,:)$
horizontal slice
$\mathcal{X}(:,j,:)$
lateral slice
$\mathcal{X}(:,:,k)$
frontal slice
fibers
mode-1 unfolding
mode-2 unfolding
mode-3 unfolding
$\mathbf{X}^{(1)} \in \mathbb{R}^{n_2 n_3 \times n_1}$
$\mathbf{X}^{(2)} \in \mathbb{R}^{n_1 n_3 \times n_2}$
$\mathbf{X}^{(3)} \in \mathbb{R}^{n_1 n_2 \times n_3}$
$\mathbf{X}^{(m)} = \mathbf{D}^{(m)}\mathbf{Z}^m + \mathbf{E}^{(m)}$ $\mathbf{Z}^m = \mathbf{W}^m\mathbf{R}^m$ $\mathbf{L}^{(m)} \triangleq \mathbf{D}^{(m)}\mathbf{Z}^m$ $\mathbf{B}^{(m)} = \mathbf{D}^{(m)}\mathbf{W}^m$
Update variables alternatively at each iteration by stochastic optimization
$fold(\mathbf{B}^{(m)})$
$fold(\mathbf{L}^{(m)})$
$fold(\mathbf{E}^{(m)})$
Average Pooling on folded tensors in all modes
$\mathcal{X}$
$\mathcal{L}$
$\mathcal{E}$

Fig. 1. ORLTM framework. Here, $\mathcal{X}$ is a third-order tensor ($m = 1, 2, 3$), which has n unit subtensors with each subtensor size as n_3. Thus, there are $N = (n/n_3)$ subtensors for online processing. The detailed notations are given in Sections III and IV. Best viewed in color.

2) ORLTM is able to handle samples sequentially and save the computational and memory cost significantly compared with batch methods.

3) We introduce a dictionary learning component that provides a more flexible set of basis for representing the low-rank component of complex tensors, hence capturing more sophisticated structures.

4) We extend ORLTM to image alignment as well as visual tracking by incorporating geometrical transformations and linearizing the tensor constraints.

5) Comprehensive empirical studies on both synthetic data and three practical tasks clearly demonstrate the superiority of ORLTM compared with several well-established batch and online approaches.

The remainder of this paper is organized as follows. Section II reviews the closely related works and Section III introduces some mathematical notations and tensor basics. Then, we describe the proposed ORLTM method in Section IV, including further analysis regarding memory cost, computational efficiency, as well as convergence. Moreover, we provide an extension of ORLTM to image alignment scenario in Section V. To investigate the performance of our approach, we carried out extensive experiments to verify its advantages and report the results in Section VI. Finally, we conclude this paper.

II. Related Work

This section reviews closely related methods in two aspects, i.e., low-rank learning methods and tensor decomposition methods, from both batch and online perspectives.

A. Low-Rank Learning Methods

Low-rank models are regarded as the useful tools to robustly handle data contaminated by gross corruption or malicious noise in a vast range of real-world applications, e.g., recovering the authentical samples from noisy and corrupted ones, modeling background of moving objects [5], tracking in video sequences [6], saliency detection [7], and hyperspectral image restoration [8]. A brief review on low-rank decomposition plus additive matrices for background/foreground separation can found in [9]. Generally speaking, most of the existing low-rank learning methods for robust recovery are typically developed upon the approach named RPCA in [3], which decomposes a given matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$ to a low-rank matrix $\mathbf{L} \in \mathbb{R}^{d\times n}$ and a sparse matrix $\mathbf{E} \in \mathbb{R}^{d\times n}$, i.e., $\mathbf{X} = \mathbf{L} + \mathbf{E}$. While the RPCA assumes the underlying data structure resides in a single low-rank subspace, the latent incoherent condition is actually not so consistent with the mixture structure of several subspaces, thus degrading its performance [10]. Motivated by this, Liu *et al.* [11] considered the data drawn from a union of multiple subspaces and proposed low-rank representation (LRR), i.e., $\mathbf{X} = \mathbf{DZ} + \mathbf{E}$, where $\mathbf{D} \in \mathbb{R}^{d\times n}$ is a given dictionary and $\mathbf{Z} \in \mathbb{R}^{n\times n}$ is the LRR. In this way, the self-expressiveness ability, i.e., each sample is a linear combination of the rest, is thus strengthened for a robust recovery. Toward the low-rank goal, a great many variants have emerged recently. From the dictionary construction perspective, Liu and Yan [12] attempted to improve the performance of the LRR using the observed data and hidden data together, while Zhang *et al.* [13] utilized the supervised method to construct a discriminative dictionary to discover semantic structure information resulting in strong identification capability for low-rank matrix recovery. From the manifold assumption perspective, Yin *et al.* [14] utilized dual-graph-regularized LRR to preserve the geometrical information in both the ambient space and the feature space, and in their following-up work [15], a variant of LRR regularized by nonnegative sparse hyper-Laplacian was proposed to consider both the global low-dimensional structure and the intrinsically geometrical information in data. From supervised learning perspective, Li and Fu [16] incorporated the low-rank constraint and the class label information to capture discriminative subspaces; Li *et al.* [17] proposed another constrained LRR method using the least-squares regularization technique; Zhou *et al.* [18] introduced the latent LRR into a classifier based on ridge regression to learn discriminative features for recognition. Moreover, Shen and Li [19] factorizes the matrix with the nuclear norm regularizer to learn structured LRR; Tang *et al.* [20] proved that the structure-constrained LRR with a pregiven weight matrix can exactly discover the relations among multiple linear disjoint subspaces. In addition, a kernel version of LRR was developed by Xiao *et al.* [21] to handle the data drawn from multiple nonlinear subspaces. However, the above-mentioned methods are typically based on batch optimization requiring large memory to store all the samples, preventing them from efficiently processing large-scale or streaming data due to memory bottleneck.

To alleviate this problem, several online learning approaches were proposed. For example, Feng *et al.* [22] developed an online RPCA using stochastic optimization which is provably robust to sparse noise; Shen *et al.* [23] put forward an online method to solve max-norm-regularized matrix decomposition problems and proved the fact that the solutions converge to a stationary point asymptotically; Zhan *et al.* [24] explored an online RPCA (ORPCA) by adopting recursive projected compressive sensing and showed correctness results; to speed up LRR, Shen *et al.* [25] designed its online implementation with guaranteed convergence. Admittedly, these methods can largely reduce the memory cost. However, they are still inappropriate for tensors since they only exploit one mode of data, far from sufficient to discover the low-rank structures of tensors.

B. Tensor Decomposition Methods

In a multidimensional data analysis, robust tensor recovery is of vital importance to handle arbitrary outliers, gross corruptions, and missing values. Unlike the above-mentioned approaches, robust tensor approaches make use of information in all modes [26] and have been shown to be capable of utilizing the multilinear structures for better decompositions in two forms [1], i.e., CANDECOMP/PARAFAC (CP) decomposition [27] and Tucker decomposition. For the former, CP-ALS [28] as a typical CP method adopts alternating least squares to solve CP tensor factorization, and recently, Zhou *et al.* [29] devised one accelerated CP to process large-scale tensors. For the latter, high-order singular value decomposition (HOSVD) is a typical approach. However, they both lack the guarantee of global optimality. This motivated Goldfarb and Qin [1] to present the alternating direction augmented Lagrangian algorithm and the accelerated proximal gradient algorithm to solve the exact constrained and the Lagrangian robust tensor recovery issue, both of which belong to HORPCA. Currently, there is no consistently unified framework for tensor analysis, e.g., Lu *et al.* [2] adopted the tensor framework of [30] to develop a tensor RPCA for image denoising, where an alternate tensor representation was exploited to show promise with respect to the tensor approximation problem; Cao *et al.* [31] presented an RPCA method in a tensor form to subtract video background, and particularly, it leverages spatial–temporal correlations to facilitate encoding video background while using spatial–temporal tensor continuity to model video foreground; Fu *et al.* [32] took advantage of both feature information and spatial structures to learn low-rank tensor representation and sparse data subspace for clustering so as to better model the inherent feature of data and its global structure; Tan *et al.* [33], [34] attempted to achieve low-n-rank tensor recovery using multilinear n-rank and ℓ_0-norm optimization in the presence of arbitrarily corrupted elements; Zhao *et al.* [35] designed a generative model under a fully Bayesian treatment for robust tensor factorization with both missing data and outliers; Yokota *et al.* [36] targeted improving the robustness of minimum description length in the Tucker model by exploiting the multilinear low-rank structure of tensors; Zhou *et al.* [37] developed an outlier-robust tensor principle component analysis method for simultaneous low-rank tensor recovery and outlier detection; Gandy *et al.* [38] regarded the n-rank of a tensor as a sparsity measure and considered the problem of finding the tensor of the lowest n-rank that satisfies some linear constraints and introduced a tractable convex relaxation; Liu *et al.* [39] addressed the missing value problem of tensor data through developing three low-rank tensor completion methods.

The above-mentioned methods all require heavy memory cost because of batch learning, so they are unable to process samples in a sequential way. To this end, several works have emerged for an online tensor analysis, such as recursive projected sparse matrix recovery [40], online tensor robust PCA [41], and online tensor subspace tracking with the CP decomposition using the recursive least squares [42]. Among them, the most closely related one is online stochastic tensor decomposition (OSTD) [43] built upon RPCA for background subtraction in multispectral video sequences. Similarly, as stated earlier for RPCA, there exists one inherent shortcoming for OSTD that it is incompetent to handle the data drawn from a union of multiple subspaces, which becomes one motivation of our ORLTM approach.

III. Notations and Preliminaries

Before introducing our method, we provide some necessary notations and definitions here. Throughout this paper, we use the calligraphic letter to represent tensor, e.g., $\mathcal{A}$, boldface uppercase letter to denote matrix, e.g., $\mathbf{A}$, boldface lowercase letter to denote vector, e.g., $\mathbf{a}$. The number of dimensions is the *order* or *mode* of a given tensor. For a third-order tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, its (i, j, k)-th *entry* is denoted as $\mathcal{X}(i, j, k)$ or x_{ijk}, and its *fiber* is a column vector defined as $\mathcal{X}(i, j, :)$; $\mathcal{X}(i, :, :)$ is the ith *horizontal slice*; $\mathcal{X}(:, j, :)$ is the jth *lateral slice*; $\mathcal{X}(:, :, k)$ or $\mathbf{X}_k$ is its kth *frontal slice*. A traditional matrix is represented by $\mathbf{A}^m$ without bracket, and $\mathbf{A}_{ij}$ denotes the (i, j)th entry of a matrix $\mathbf{A}$.

A. Tensor Unfolding [1], [28]

The mode-m unfolding of $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_M}$, $\mathbf{A}^{(m)}$, is derived by organizing the mode-m fibers into the matrix columns, i.e., $\text{unfold}_m(\mathcal{A}) = \mathbf{A}^{(m)} \in \mathbb{R}^{n_{M\backslash m} \times n_m}$, where $n_{M\backslash m} = \prod_{j \neq m, j=1}^{M} n_j$ and $\prod_{j=1}^{M} n_j = n_1 \times n_2 \times \ldots \times n_M$.

B. Tensor Folding [1]

The formula $\text{fold}_m(\mathbf{A}) = \mathcal{A}_m$ folds the mode-m unfolding of $\mathcal{A}$ and gives the corresponding tensor in mode m. In the following, we omit the subscript m for brevity when the matrix has indicated its unfolding mode.

C. Tensor Vectorization [1]

It aligns the elements of given tensor into a long column vector, denoted by $\text{vec}(\mathcal{A})$. The mode-m vectorization of $\mathcal{A}$ is $\text{vec}(\mathcal{A}_m)$ [also denoted as $\text{vec}(\mathbf{A}^{(m)})$] which arranges all columns of mode-m unfolding into a single column vector.

D. *Mode*-m *Product* [1]

The mode-m product of tensor $\mathcal{A}$ and matrix $\mathbf{U}^m \in \mathbb{R}^{n_m \times n_c}$ in mode m is represented as $\mathcal{A} \times_m \mathbf{U}$. Here, $(\mathcal{A} \times_m \mathbf{U}^m)^{(m)} = \mathbf{A}^{(m)}\mathbf{U}^m$, where $\mathbf{A}^{(m)} \in \mathbb{R}^{n_{M\backslash m} \times n_m}$ is the mth mode of the tensor.

E. Tensor Norms [1]

The ℓ_1-norm is $\|\mathcal{A}\|_1 = \|\text{vec}(\mathcal{A})\|_1 = \sum_{ijk} |a_{ijk}|$, and the Frobenius norm is $\|\mathcal{A}\|_F = (\text{vec}(A)^\top \text{vec}(A))^{1/2} = (\sum_{ijk} a_{ijk}^2)^{1/2}$. These norms will degenerate to the corresponding matrix or vector norms when $\mathcal{A}$ is a matrix or a vector.

F. Tensor Rank [1]

The mode-m rank of a tensor $\mathcal{A}$, denoted by $\text{rank}_m(\mathcal{A})$, is the column rank of the unfolding $\mathbf{A}^{(m)}$, and the set of M mode-m ranks is called *Tucker rank*. However, minimizing Tucker rank is always NP-hard, and thus, its convex surrogate $\text{CTrank}(\mathcal{A})$ is often used in practice [44], [45], i.e., $\text{CTrank}(\mathcal{A}) := \sum_{m=1}^{M} \|\mathbf{A}^{(m)}\|_*$.

IV. Online Robust Low-Rank Tensor Modeling

This section develops the objective function of our method at first and then elaborates on the proposed ORLTM, which is solved by a stochastic optimization technique. In addition, computational complexity, memory cost, and convergence analysis are provided.

A. Objective Function

We now introduce the objective formulation of our proposed approach. First, we review the batch-based HORPCA [1], which decomposes the given tensor into a low-rank component and a sparse one that encodes malicious outliers or corruptions. Its slicewise model is expressed by

$$\min_{\mathcal{L},\mathcal{E}} \sum_{m=1}^{M} \|\mathbf{L}^{(m)}\|_* + \lambda_1 \|\mathcal{E}\|_1, \quad \text{s.t. } \mathcal{X} = \mathcal{L} + \mathcal{E} \tag{1}$$

where $\mathcal{X}, \mathcal{L}, \mathcal{E} \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_M}$ denote the observed noisy tensor, its low-rank component, and the sparse component, respectively; the constant $\lambda_1 > 0$, $\mathcal{L} = \text{fold}(\mathbf{L}^{(m)})$, and $\sum_{m=1}^{M} \|\mathbf{L}^{(m)}\|_* = \text{CTrank}(\mathcal{L})$. In fact, the solutions to (1) can lead to distinct low-rank and sparse components for different mode-m unfoldings $\mathbf{X}^{(m)}$. Hence, we introduce a set of auxiliary tensor variables, i.e., $\{\mathcal{L}_m\}_{m=1}^{M}$ and $\{\mathcal{E}_m\}_{m=1}^{M}$, and utilize the variable-splitting strategy. Thus, the formulation (1) can be rewritten as

$$\begin{aligned} \min_{\substack{\mathcal{L}_m, \mathcal{E}_m \\ m=1,2,\ldots,M}} \; & \sum_{m=1}^{M} \|\mathbf{L}^{(m)}\|_* + \lambda_1 \|\mathbf{E}^{(m)}\|_1 \\ \text{s.t. } & \mathbf{X}^{(m)} = \mathbf{L}^{(m)} + \mathbf{E}^{(m)}, \quad m = 1, \ldots, M \\ & \mathcal{L}_m = \text{fold}(\mathbf{L}^{(m)}), \quad \mathcal{E}_m = \text{fold}(\mathbf{E}^{(m)}) \end{aligned} \tag{2}$$

where $\mathcal{X} = \text{fold}(\mathbf{X}^{(m)})$, and each unfolding $\mathbf{X}^{(m)}$ in all modes should return the common tensor $\mathcal{X}$ through the fold$(\cdot)$ operator. As a common practice indicated in [1], [33], and [43], we can obtain the relaxed solutions to the above tensor variables by average pooling on those auxiliary tensors, i.e., $\mathcal{L} = (1/M)\sum_{m=1}^{M} \mathcal{L}_m$ and $\mathcal{E} = (1/M)\sum_{m=1}^{M} \mathcal{E}_m$.

As shown in (2), the unfolding matrix is constrained to residing in a single low-rank subspace. This makes it unable to handle the data drawn from a union of multiple subspaces. Therefore, the constraints are too tight to hold when data are with a mixture structure of several subspaces, resulting in inferior performance for real-world applications. To capture the intrinsic low-rank structure more accurately, we introduce a pregiven dictionary $\mathbf{D}^{(m)} \in \mathbb{R}^{n_{M\backslash m} \times n_m}$ which can be simply set to the data. By imposing nuclear norm on $\mathbf{Z}^m \in \mathbb{R}^{n_m \times n_m}$, i.e., $\|\mathbf{Z}^m\|_*$, the low-rank property is well preserved. Since the inequality $\operatorname{rank}(\mathbf{D}^{(m)}\mathbf{Z}^m) \le \operatorname{rank}(\mathbf{Z}^m)$ always holds, minimizing the nuclear norm of $\mathbf{Z}^m$ can actually bound the rank of the clean data $\mathbf{L}^{(m)}$ when $\mathbf{L}^{(m)} \triangleq \mathbf{D}^{(m)}\mathbf{Z}^m$, leading to low-rank structure. Then, we have

$$\min_{\substack{\mathbf{Z}^m,\mathbf{E}^{(m)}\\ m=1,2,\ldots,M}} \sum_{m=1}^{M} \|\mathbf{Z}^m\|_* + \lambda_1 \|\mathbf{E}^{(m)}\|_1 \\ \text{s.t. } \mathbf{X}^{(m)} = \mathbf{D}^{(m)}\mathbf{Z}^m + \mathbf{E}^{(m)}, \quad m = 1, \ldots, M. \tag{3}$$

The multiplication of $\mathbf{D}^{(m)}$ and $\mathbf{Z}^m$ yields the unfolding matrix $\mathbf{L}^{(m)}$ which is the mode-m unfolding of the low-rank tensor $\mathcal{L}$. To process streaming data, instead of solving the above-constrained problem directly, we relax the constraints by regarding them as quadratic penalties to facilitate online optimization, resulting in

$$\min_{\substack{\mathbf{Z}^m,\mathbf{E}^{(m)}\\ m=1,2,\ldots,M}} \frac{1}{2}\sum_{m=1}^{M} \|\mathbf{X}^{(m)} - \mathbf{D}^{(m)}\mathbf{Z}^m - \mathbf{E}^{(m)}\|_F^2 \\ + \lambda_1 \|\mathbf{E}^{(m)}\|_1 + \lambda_2 \|\mathbf{Z}^m\|_* \tag{4}$$

where $\lambda_2 > 0$ controls the contribution of the nuclear norm regularizer.

However, calculating the nuclear norm often consumes large memory and is computationally unaffordable for analyzing the large-scale data. To sequentially process samples, we propose to adopt the bifactor factorization form $\mathbf{Z}^m = \mathbf{W}^m\mathbf{R}^m$ [46], where $\mathbf{W}^m \in \mathbb{R}^{n_m \times p}$ and $\mathbf{R}^m \in \mathbb{R}^{p \times n_m}$ with $p \ll \min(n_m, n_{M\backslash m})$. In consequence, the rank of $\mathbf{Z}^m$ is upper bounded by the constant p. As pointed out in [47] and [48], minimizing $\|\mathbf{Z}^m\|_*$ is equivalent to minimizing $\|\mathbf{W}^m\|_F^2$ and $\|\mathbf{R}^m\|_F^2$ at the same time. Thus, the unconstrained problem in (4) is converted into a nonconvex optimization problem

$$\min_{\substack{\mathbf{W}^m,\mathbf{R}^m,\mathbf{E}^{(m)}\\ m=1,2,\ldots,M}} \frac{1}{2}\sum_{m=1}^{M} \|\mathbf{X}^{(m)} - \mathbf{D}^{(m)}\mathbf{W}^m\mathbf{R}^m - \mathbf{E}^{(m)}\|_F^2 \\ + \lambda_1 \|\mathbf{E}^{(m)}\|_1 + \frac{\lambda_2}{2}\big(\|\mathbf{W}^m\|_F^2 + \|\mathbf{R}^m\|_F^2\big). \tag{5}$$

This is the objective function of our method, for which updating the entries in $\mathbf{Z}^m$ amounts to updating the corresponding rows of $\mathbf{W}^m$ and the columns of $\mathbf{R}^m$ on the fly.

It can be observed from (5) that unfolding matrices scale up as n_M gets larger and, at each iteration, the dictionary $\mathbf{D}^{(m)}$ is only partially accessed. Moreover, the rows in $\mathbf{W}^{(m)}$ are coupled together as being multiplied by the left dictionary. To address these shortcomings, we introduce another set of auxiliary variables $\mathbf{B}^{(m)} = \mathbf{D}^{(m)}\mathbf{W}^m \in \mathbb{R}^{n_{M\backslash m} \times p}$, $m = 1, 2, \ldots, M$, to approximate the recovery part $(\mathbf{X}^{(m)} - \mathbf{E}^{(m)})$ by $\mathbf{B}^{(m)}\mathbf{R}^m$. This indicates that the introduced dictionaries $\{\mathbf{B}^{(m)}\}_{m=1}^{M}$ can be regarded as *reinforced basis dictionaries*, while $\{\mathbf{R}^m\}_{m=1}^{M}$ are the low-dimensional coefficients. Compared with the pregiven $\mathbf{D}^{(m)}$, the learned dictionary $\mathbf{B}^{(m)}$ is consolidated by considering the two factored low-rank components of $\mathbf{Z}^m$ iteratively. Hence, (5) can be approximated as

$$\min_{\substack{\mathbf{B}^{(m)},\mathbf{W}^m,\mathbf{R}^m,\mathbf{E}^{(m)}\\ m=1,2,\ldots,M}} \sum_{m=1}^{M} \frac{1}{2}\|\mathbf{X}^{(m)} - \mathbf{B}^{(m)}\mathbf{R}^m - \mathbf{E}^{(m)}\|_F^2 \\ + \lambda_1 \|\mathbf{E}^{(m)}\|_1 + \frac{\lambda_2}{2}\big(\|\mathbf{W}^m\|_F^2 + \|\mathbf{R}^m\|_F^2\big) \\ + \frac{\lambda_3}{2}\|\mathbf{B}^{(m)} - \mathbf{D}^{(m)}\mathbf{W}^m\|_F^2 \tag{6}$$

where $\lambda_3 > 0$ governs the reconstruction ability of $\mathbf{B}^{(m)}$. The above-mentioned function is more beneficial and more informative for learning from streaming data, because it encodes the basis of the union of multiple subspaces explicitly across all modes of tensor data. Therefore, it enables more promising low-rank tensor modeling for tensor subspace recovery.

B. Online Implementation of ORLTM

In this section, we propose an online optimization algorithm to optimize the objective function in (6). It allows the low-rank tensor modeling to take in one sample or one minibatch at each time instance and adopts the stochastic optimization strategy. In the beginning, we define two functions in mode-m

$$\tilde{\ell}(\mathbf{x}^m, \mathbf{B}^{(m)}, \mathbf{r}^m, \mathbf{e}^m) \triangleq \frac{1}{2}\|\mathbf{x}^m - \mathbf{B}^{(m)}\mathbf{r}^m - \mathbf{e}^m\|_2^2 \\ + \lambda_1 \|\mathbf{e}^m\|_1 + \frac{\lambda_2}{2}\|\mathbf{r}^m\|_2^2 \\ \ell(\mathbf{x}^m, \mathbf{B}^{(m)}) = \min_{\mathbf{r}^m,\mathbf{e}^m} \tilde{\ell}(\mathbf{x}^m, \mathbf{B}^{(m)}, \mathbf{r}^m, \mathbf{e}^m) \tag{7}$$

$$\tilde{h}(\mathbf{D}^{(m)}, \mathbf{B}^{(m)}, \mathbf{W}^m) \triangleq \sum_{i=1}^{N} \frac{\lambda_2}{2}\|\mathbf{w}_i^m\|_2^2 \\ + \frac{\lambda_3}{2}\|\mathbf{B}^{(m)} - \sum_{i=1}^{N} \mathbf{d}_i^m \mathbf{w}_i^m\|_F^2 \\ h(\mathbf{D}^{(m)}, \mathbf{B}^{(m)}) = \min_{\mathbf{W}^m} \tilde{h}(\mathbf{D}^{(m)}, \mathbf{B}^{(m)}, \mathbf{W}^m) \tag{8}$$

where $\mathbf{x}^m$, $\mathbf{e}^m$, $\mathbf{d}^m$, and $\mathbf{r}^m$ denote the columns in matrices $\mathbf{X}^{(m)}, \mathbf{E}^{(m)}, \mathbf{D}^{(m)} \in \mathbb{R}^{n_{M\backslash m} \times n_m}$, and $\mathbf{R}^m \in \mathbb{R}^{p \times n_m}$, respectively, and $\mathbf{w}^m$ is the row vector of $\mathbf{W}^m \in \mathbb{R}^{n_m \times p}$. In virtue of these formulations, (6) can be expressed by

$$\min_{\mathbf{B}^{(m)}} \min_{\substack{\mathbf{E}^{(m)},\\ \mathbf{W}^m,\mathbf{R}^m}} \sum_{i=1}^{N} \tilde{\ell}\big(\mathbf{x}_i^m, \mathbf{B}_i^{(m)}, \mathbf{r}_i^m, \mathbf{e}_i^m\big) + \tilde{h}(\mathbf{D}^{(m)}, \mathbf{B}^{(m)}, \mathbf{W}^m) \tag{9}$$

where N is the number of samples to be handled. This adds to minimizing the following loss function:

$$f_N(\mathbf{B}^{(m)}) \triangleq \frac{1}{N}\sum_{i=1}^{N} \ell\big(\mathbf{x}_i^m, \mathbf{B}^{(m)}\big) + \frac{1}{N} h(\mathbf{D}^{(m)}, \mathbf{B}^{(m)}).$$

Next, we introduce the way to calculate those variables in (6) online by optimizing a jointly nonconvex problem. Here, we utilize the alternative optimization, i.e., solving one variable while fixing the rest. The entire algorithmic framework is described in Algorithm 1, which includes the following alternative variable updatings at each iteration t.

1) Updating r_t^m, e_t^m: Given the dictionary $\mathbf{B}_{t-1}^{(m)}$ in the previous iteration, we get the optimal $\{\mathbf{r}_t^m, \mathbf{e}_t^m\}$ from

$$\min_{\mathbf{r}^m, \mathbf{e}^m} \tilde{\ell}\big(\mathbf{x}_t^m, \mathbf{B}_{t-1}^{(m)}, \mathbf{r}^m, \mathbf{e}^m\big). \tag{10}$$

When $\mathbf{e}^m$ keeps still, there exists a closed-form solution of $\mathbf{r}_t^m$

$$\mathbf{r}^m = \big(\mathbf{B}_{t-1}^{(m)\top}\mathbf{B}_{t-1}^{(m)} + \lambda_2 \mathbf{I}_p\big)^{-1}\mathbf{B}_t^{(m)\top}\big(\mathbf{x}_t^m - \mathbf{e}^m\big) \tag{11}$$

and the local minimizer of $\mathbf{e}^m$ with respect to fixed $\mathbf{r}^m$ is obtained using the soft-thresholding operator [49]

$$\mathbf{e}^m = \mathcal{S}_{\lambda_1}\big[\mathbf{x}_t^m - \mathbf{B}_{t-1}^{(m)}\mathbf{r}^m\big]. \tag{12}$$

Generally, the solutions $\mathbf{r}_t^m$ and $\mathbf{e}_t^m$ could be efficiently learned by adopting a coordinate descent algorithm [50].

2) Updating w_t^m: We define an accumulation matrix $\mathbf{G}_{t-1}^{(m)} = \sum_{i=1}^{t-1}\mathbf{d}_i^m\mathbf{w}_i^m \in \mathbb{R}^{n_{M\backslash m}\times p}$, where $\mathbf{G}_0^m = 0$, and then get $\mathbf{w}_t^m$ through minimizing the following:

$$\begin{aligned}&\tilde{\ell}_2\big(\mathbf{d}_t^m, \mathbf{B}_{t-1}^{(m)}, \mathbf{G}_{t-1}^{(m)}, \mathbf{w}^m\big)\\ &\quad \triangleq \frac{\lambda_2}{2}\|\mathbf{w}^m\|_2^2 + \frac{\lambda_3}{2}\big\|\mathbf{B}_{t-1}^{(m)} - \mathbf{G}_{t-1}^{(m)} - \mathbf{d}_t^m\mathbf{w}^m\big\|_F^2.\end{aligned} \tag{13}$$

This leads to the closed-form solution

$$\mathbf{w}_t^m = \left(\big\|\mathbf{d}_t^m\big\|_2^2 + \frac{\lambda_2}{\lambda_3}\right)^{-1}\mathbf{d}_t^{m\top}\big(\mathbf{B}_{t-1}^{(m)} - \mathbf{G}_{t-1}^{(m)}\big). \tag{14}$$

Here, we provide the intuition behind (13) as follows. The variable $\mathbf{w}_t^m$ depends on the entire dictionary $\mathbf{D}^{(m)}$, and only the current atom $\mathbf{d}_t^m$ can be accessed. We thus update $\mathbf{w}_t^m$ by holding the rest given the observed tth atom $\mathbf{d}_t^m$, and each $\mathbf{w}_t^m$ is updated streamingly only once while revealing all the atoms. This technique is actually the one-pass block-coordinate descent method, which can be readily extended to its multiple-pass version in demanding real-world tasks.

3) Updating $B_t^{(m)}$: We update the reinforced basis dictionary by minimizing the surrogate function

$$\begin{aligned}h_t(\mathbf{B}^{(m)}) &\triangleq \frac{1}{t}\sum_{i=1}^{t}\tilde{\ell}\big(\mathbf{x}_i^m, \mathbf{B}^{(m)}, \mathbf{r}_i^m, \mathbf{e}_i^m\big)\\ &\quad + \frac{\lambda_2}{2t}\sum_{i=1}^{t}\big\|\mathbf{w}_i^m\big\|_2^2 + \frac{\lambda_3}{2t}\big\|\mathbf{B}^{(m)} - \mathbf{G}_t^{(m)}\big\|_F^2.\end{aligned} \tag{15}$$

For dictionary updates, the surrogate $h_t(\cdot)$ can be amounted by a few sufficient statistics updated sequentially, i.e., $h_t(\cdot)$ can be captured without explicitly storing earlier samples [51]. In general, there are two common ways to solve this problem: one is directly computing the closed-form solution and the other is adopting stochastic gradient descent.

Algorithm 1 ORLTM

```
Input: Observed $M$-th order tensor $\mathcal{X} \in \mathbb{R}^{n_1\times n_2\times\ldots\times n}$ and
    pregiven dictionary $\mathcal{D}$, tradeoff parameters $\{\lambda_1, \lambda_2, \lambda_3\}$,
    target rank $p$, sub-tensor size $n_M$.
Output: Low-rank tensor $\mathcal{L}$ and sparse tensor $\mathcal{E}$.
Initialize: Set all entries of $\{\mathcal{L}, \mathcal{E}\} \in \mathbb{R}^{n_1\times n_2\times\ldots\times n}$, $\mathbf{W}^m \in$
    $\mathbb{R}^{n_m\times p}$, $\mathbf{R}^m \in \mathbb{R}^{p\times n_m}$, $\mathbf{G}_0^{(m)}$, $\Theta_0^{(m)}$, $\Phi_0^m$ to zero, initialized
    $\mathcal{B} \in \mathbb{R}^{n_1\times n_2\times\ldots\times p}$, $N = \frac{n}{n_M}$.
for $t = 1$ to $N$ do
  for $m = 1$ to $M$ do
    Access the $t$-th sample $\mathbf{x}_t^m$ and the $t$-th atom $\mathbf{d}_t^m$.
    Obtain coefficients $\mathbf{r}_t^m$ and sparse vectors $\mathbf{e}_t^m$ by
    optimizing $\tilde{\ell}(\mathbf{x}_t^m, \mathbf{B}_{t-1}^{(m)}, \mathbf{r}^m, \mathbf{e}^m)$ and utilize coordinate
    descent algorithm for (11)(12) to derive the solutions.
    Compute coefficients $\mathbf{w}_t^m$ by (14) via minimizing
    $\tilde{\ell}_2(\mathbf{d}_t^m, \mathbf{B}_{t-1}^{(m)}, \mathbf{G}_{t-1}^m, \mathbf{w}^m)$.
    Update accumulation matrices:
    $\mathbf{G}_t^{(m)} \leftarrow \mathbf{G}_{t-1}^{(m)} + \mathbf{d}_t^m\mathbf{w}_t^m$,
    $\Theta_t^{(m)} \leftarrow \Theta_{t-1}^{(m)} + (\mathbf{x}_t^m - \mathbf{e}_t^m)\mathbf{r}_t^{m\top}$,
    $\Phi_t^m \leftarrow \Phi_{t-1}^m + \mathbf{r}_t^m\mathbf{r}_t^{m\top}$.
    Update dictionary $\mathbf{B}_t^{(m)}$ by (16).
    Update low-rank components: $\mathbf{L}_t^{(m)} \leftarrow \mathbf{B}_t^{(m)}\mathbf{r}_t^m$.
    Update sparse components: $\mathbf{E}_t^{(m)} \leftarrow \mathbf{e}_t^m$.
  end for
end for
Average pooling on mode-$m$ foldings, i.e., $\mathcal{L} =$
$\frac{1}{M}\sum_{m=1}^{M} fold(\mathbf{L}^{(m)})$, $\mathcal{E} = \frac{1}{M}\sum_{m=1}^{M} fold(\mathbf{E}^{(m)})$.
```

If we define two accumulation matrices as

$$\Phi_t^m = \sum_{i=1}^{t}\mathbf{r}_i^m\mathbf{r}_i^{m\top},\ \ \Theta_t^{(m)} = \sum_{i=1}^{t}\big(\mathbf{x}_i^m - \mathbf{e}_i^m\big)\mathbf{r}_i^{m\top}$$

then we have

$$\mathbf{B}_t^{(m)} = \big(\Theta_t^{(m)} + \lambda_3\mathbf{G}_t^{(m)}\big)\big(\Phi_t^m + \lambda_3\mathbf{I}_p\big)^{-1} \tag{16}$$

where $\Phi_t^m \in \mathbb{R}^{p\times p}$ and $\Theta_t^{(m)} \in \mathbb{R}^{n_{M\backslash m}\times p}$ are independent of the data size N, which allows ORLTM possibly to handle large-scale tensor data.

We also provide a dictionary update strategy using stochastic gradient descent optimization. In concrete, if we define $\boldsymbol{\phi}_t^m = \mathbf{r}_t^m\mathbf{r}_t^{m\top}$ and $\boldsymbol{\theta}_t^m = (\mathbf{x}_t^m - \mathbf{e}_t^m)\mathbf{r}_t^{m\top}$, then the gradient of a surrogate function $h_t(\mathbf{B}^{(m)})$ with respect to $\mathbf{B}^{(m)}$ is

$$\nabla_{\mathbf{B}^{(m)}}h_t(\mathbf{B}^{(m)}) = \mathbf{B}^{(m)}\big(\boldsymbol{\phi}_t^m + \lambda_3\mathbf{I}_p\big) - \lambda_3\big(\boldsymbol{\theta}_t^m + \mathbf{G}_t^{(m)}\big). \tag{17}$$

Given the recovered sample $(\mathbf{x}_t^m - \mathbf{e}_t^m)$ in iteration, the basis dictionary $\mathbf{B}_t$ in mode m can be updated as follows:

$$\mathbf{B}_t^{(m)} \leftarrow \mathbf{B}_{t-1}^{(m)} - \eta\nabla_{\mathbf{B}^{(m)}}h_t(\mathbf{B}^{(m)}) \tag{18}$$

where $\eta > 0$ is the learning rate during the optimization.

In Algorithm 1, for an Mth order tensor $\mathcal{X}$, the symbol n_M denotes the number of *unit subtensor* samples, i.e., one $(M-1)$th order subtensor. If the number of *unit* subtensor samples denotes the entire tensor size n, then $N = (n/n_M)$ is the number of subtensor samples. In each iteration, one subtensor is fed into ORLTM, either *single* unit subtensor sample ($n_M = 1$) or *multiple* unit subtensor samples. At the tth iteration, the vector $\mathbf{x}_t^m$ is the tth column of the mode-m unfolding matrix $\mathbf{X}^{(m)}$ and, similarly, $\mathbf{d}_t^m$ is that of $\mathbf{D}^{(m)}$. The pregiven dictionary $\mathcal{D}$ is set to the tensor $\mathcal{X}$ in this paper.

C. Complexity and Convergence Analysis

Here, we provide further analysis on computational complexity, memory cost, and convergence for our method.

1) Computational Complexity: In each iteration, there are four variables to be computed. First, it is cheap to calculate $\{\mathbf{r}_t^m, \mathbf{e}_t^m\}$ as one might adopt the block-coordinate descent algorithm with lienar convergence [52]. Next, it needs $\mathcal{O}(n_{M\backslash m}p)$ for $\mathbf{w}_t^m$ to do matrix-vector multiplication, where $p \ll \min(n_{M\backslash m}, n_m)$, and the subtensor size n_M appears to be small in practical settings. Moreover, updating accumulation matrices $\mathbf{G}_t^{(m)}$, $\Theta_t^{(m)}$, and $\Phi_t^{(m)}$ requires $\mathcal{O}(n_{M\backslash m}p)$. In addition, it costs $\mathcal{O}(n_{M\backslash m}p^2)$ to update $\mathbf{B}_t^m$. Hence, the overall computational cost is relatively limited.

2) Memory Cost: ORLTM requires $\mathcal{O}(n_{M\backslash m}p)$ to load $\mathbf{B}_t^{(m)}$ and $\mathbf{x}_t^m$ to obtain $\{\mathbf{r}_t^m, \mathbf{e}_t^m\}$, and also $\mathcal{O}(n_{M\backslash m}p)$ to employ $\mathbf{B}_t^{(m)}$, $\mathbf{G}_t^{(m)}$ for computing $\mathbf{w}_t^m$. Since the history information of $h_t(\mathbf{B}_t^{(m)})$ in (15) has been stored by those accumulation matrices $\mathbf{G}_t^{(m)}$, $\Theta_t^{(m)}$, and $\Phi_t^{(m)}$, it costs at most $\mathcal{O}(n_{M\backslash m}p)$. Therefore, ORLTM only desires $\mathcal{O}(n_{M\backslash m}p)$ for memory cost in each iteration. Thus, it is independent of the number of subtensor samples, saving substantial memory for large-scale streaming tensor data.

3) Convergence: For convergence analysis, three assumptions are necessary. 1) the observed subtensor samples are generated independent identically distributed from some distribution and there exist two constants α_0 and α_1, such that the conditions $0 < \alpha_0 \leq \|\mathbf{x}^m\|_2 \leq \alpha_1$ and $\alpha_0 \leq \|\mathbf{d}^m\|_2 \leq \alpha_1$ hold almost surely; 2) for $\mathbf{Q}_t^{(m)} = (1/t)\sum_{i=1}^t \mathbf{d}_i^m \mathbf{d}_i^{m\top}$, the smallest nonzero singular value is lower bounded away from zero; and 3) $\forall\ t \geq 0$, the surrogates $h_t(\mathbf{B}^{(m)})$ are strongly convex. Following proof techniques in [25] and [53], we derive several theoretical results: 1) the surrogate function $h_t(\mathbf{B}_t^{(m)})$ in (15) converges almost surely, while $\mathbf{B}_t^{(m)}$ is the solution produced by Algorithm 1; 2) if $\{\mathbf{B}_t^{(m)}\}_{t=1}^{\infty}$ is the sequence of optimal basis from Algorithm 1, then these sequences converge to a stationary point of $f(\mathbf{B}^{(m)})$ when t goes to infinity; 3) let $\{\mathbf{G}_t^{(m)}\}_{t\geq 0}$ be the sequence of matrices derived from Algorithm 1, and there exists some universal constant C_0, such that $\forall\ t \geq 0, \exists\ \|\mathbf{G}_t^{(m)}\|_F \leq C_0$. Also, the solution $\mathbf{B}_t^{(m)}$ is inherently determined by $\Theta_t^{(m)}/t$ and Φ_t^m/t when t becomes very large, since $\mathbf{G}_t^{(m)}/t \to 0$. Furthermore, the numerical convergence rate of $\mathbf{B}_t^{(m)}$ is nonasymptotic, since $\|\mathbf{B}_t^{(m)} - \mathbf{B}_{t-1}^{(m)}\|_2 = \mathcal{O}(1/t)$.

V. Extension to Image Alignment

Misalignments together with corruptions and occlusions often appear in unconstrained real-world streaming data. Image alignment is an essential problem in a multimedia analysis and computer vision and has received many algorithmic solutions. For example, Peng *et al.* [54] attempted to align images by seeking a set of transformations which minimizes the rank of warped images corrupted by sparse noise. However, as a batch-based algorithm, their method requires large memory cost when aligning a large number of images. For scalability, Wu *et al.* [55] developed an online image alignment method that uses a fixed rank model and updates the basis through thresholding and simple replacement. Similarly, Song *et al.* [56] proposed an online robust PCA method for image alignment. This section extends ORLTM to solve the image alignment problem from the perspective of robust tensor learning.

Algorithm 2 ORLTM-IA

```
Input: A third-order tensor consisting of unaligned images $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n}$, initial transformation $\Gamma$, tradeoff parameters $\{\lambda_1, \lambda_2, \lambda_3\}$, target rank $p$.
Output: Aligned image tensor $\tilde{\mathcal{X}} = \mathcal{X} \circ \Gamma$, low-rank tensor $\mathcal{L}$ and sparse tensor $\mathcal{E}$.
Initialize: Set $\{\mathcal{L}, \mathcal{E}\} \in \mathbb{R}^{n_1 \times n_2 \times n}$, $\mathbf{W}^m \in \mathbb{R}^{n_m \times p}$, $\mathbf{R}^m \in \mathbb{R}^{p \times n_m}$, $\mathbf{G}_0^{(m)}$, $\Theta_0^{(m)}$, $\Phi_0^m$ to zero, $\mathcal{B} \in \mathbb{R}^{n_1 \times n_2 \times p}$.
for $t = 1$ to $n$ do
  while not converged do
    Compute the warped image $\tilde{\mathbf{X}}_t = \mathbf{X}_t \circ \tau_t$.
    Update the Jacobian $\mathbf{J}_t = \frac{\partial}{\partial \zeta} vec(\mathbf{X}_t \circ \zeta)|_{\zeta = \tau_t}$.
    for $m = 1$ to 3 do
      $\tilde{\mathbf{x}}_t^m \leftarrow vec(\tilde{\mathbf{X}}_t^{(m)})$, $\tilde{\mathbf{x}}_t^m \leftarrow \frac{\tilde{\mathbf{x}}_t^m}{\|\tilde{\mathbf{x}}_t^m\|_2}$.
      $\mathbf{d}_t^m \leftarrow \tilde{\mathbf{x}}_t^m$.
      while not converged do
        $\Delta\tilde{\mathbf{x}}_t^m \leftarrow (\mathbf{J}_t \Delta\tau_t)^m$.
        Update $\mathbf{r}_t^m$ by (23).
        Update $\mathbf{e}_t^m$ by (24).
        Update $\Delta\tau_t$ by (22) in mode-3.
      end while
      Compute $\mathbf{w}_t^m$ using (14).
    end for
    $\tau_t \leftarrow \tau_t + \Delta\tau_t$.
  end while
  for $m = 1$ to 3 do
    Update the dictionary $\mathbf{B}_t^{(m)}$ by (16) or (18).
    $\mathbf{L}_t^{(m)} \leftarrow \mathbf{B}_t^{(m)}\mathbf{r}_t^m$, $\mathbf{E}_t^{(m)} \leftarrow \mathbf{e}_t^m$.
  end for
end for
```

Formally, a group of n images $\{\mathbf{X}_i\}_{i=1}^n$ with width n_1 and height n_2 ($d = n_1 \times n_2$), such as video frames, can be regarded as a third-order tensor ($M = 3$). Each image is a frontal slice of the tensor. Although a stack of well-aligned images would form a low-rank tensor where noise or corruptions can be modeled as sparse error, the misalignments existing in a few images would break the low-rank structure of the tensor. Thus, we introduce a set of geometrical transformations $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$ to align the 2-D slices of the tensor and reformulate the constraint in (1) as

$$\mathcal{X} \circ \Gamma = \mathcal{L} + \mathcal{E} \tag{19}$$

where $\mathcal{X} \circ \Gamma = \tilde{\mathcal{X}}$ applies the transformation $\tau_i \in \mathbb{R}^q$ to each frontal slice matrix $\mathcal{X}(:, :, i)$ or $\mathbf{X}_i$ from the tensor $\mathcal{X}$.

When the changes $\Delta\Gamma = \{\Delta\tau_1, \Delta\tau_2, \dots, \Delta\tau_n\}$ in Γ are small, the constraint in (19) can be approximated by linearizing the current estimate of Γ [57] as follows:

$$\mathcal{X} \circ (\Gamma + \Delta\Gamma) \approx \mathcal{X} \circ \Gamma + \text{fold}_3\left(\sum_{i=1}^{n} \mathbf{J}_i \Delta\Gamma \epsilon_i \epsilon_i^\top\right) \tag{20}$$

where $\mathbf{J}_i$ denotes the Jacobian of the frontal slice matrix $\mathcal{X}(:,:,i)$ with respect to the transformation parameter $\tau_i \in \mathbb{R}^q$ and $\epsilon_i \in \mathbb{R}^n$ is the standard basis. For brevity, we define $\Delta\mathcal{X} = \text{fold}_3(\sum_{i=1}^{n} \mathbf{J}_i \Delta\Gamma \epsilon_i \epsilon_i^\top)$, which reflects the change in the tensor $\mathcal{X}$ along the third mode.

To align images in an online manner, we follow the previous deductions and incorporate image warping into the formulation. Specifically, we warp the image by geometrical transformations and reformulate (7) to update $\Delta\tau$ as

$$\tilde{\ell}(\Delta\tau) \triangleq \min_{\mathbf{B}^{(m)}, \mathbf{r}^m, \mathbf{e}^m} \frac{1}{2}\left\|\tilde{\mathbf{x}}_t^m + (\mathbf{J}_t \Delta\tau)^m - \mathbf{B}^{(m)}\mathbf{r}_t^m - \mathbf{e}_t^m\right\|_2^2 + \lambda_1 \left\|\mathbf{e}_t^m\right\|_1 + \frac{\lambda_2}{2}\left\|\mathbf{r}_t^m\right\|_2^2 \tag{21}$$

where $\tilde{\mathbf{x}}_t^m = \text{vec}(\mathbf{X}_t \circ \tau_t)^m$ denotes the warped image in mode-m, $\mathbf{J}_t = (\partial/\partial\zeta)\text{vec}(\mathbf{X}_t \circ \zeta)|_{\zeta=\tau_t} \in \mathbb{R}^{d\times q}$ is the Jacobian of the tth image with respect to τ_t, and $\Delta\tilde{\mathbf{x}}_t^m = (\mathbf{J}_t \Delta\tau_t)^m$ reveals the image change. It is actually an unconstrained least squares problem to solve $\Delta\tau$, which has a concise solution

$$\Delta\tau_t = \left(\mathbf{J}_t^\top \mathbf{J}_t\right)^{-1} \mathbf{J}_t^\top (\mathbf{B}\mathbf{r}_t + \mathbf{e}_t - \tilde{\mathbf{x}}_t). \tag{22}$$

Also, we alternatively update $\mathbf{r}^m$, $\mathbf{e}^m$ and $\Theta_t^{(m)}$ as follows:

$$\mathbf{r}^m = \left(\mathbf{B}_{t-1}^{(m)\top}\mathbf{B}_{t-1}^{(m)} + \lambda_2 \mathbf{I}_p\right)^{-1}\mathbf{B}_t^{(m)\top}\left(\mathbf{x}_t^m + \Delta\tilde{\mathbf{x}}_t^m - \mathbf{e}^m\right) \tag{23}$$

$$\mathbf{e}^m = \mathcal{S}_{\lambda_1}\left[\mathbf{x}_t^m + \Delta\tilde{\mathbf{x}}_t^m - \mathbf{B}_{t-1}^{(m)}\mathbf{r}^m\right] \tag{24}$$

$$\Theta_t^{(m)} = \sum_{i=1}^{t} \left(\mathbf{x}_i^m + \Delta\tilde{\mathbf{x}}_i^m - \mathbf{e}_i^m\right)\mathbf{r}_i^{m\top}. \tag{25}$$

We summarize the proposed ORLTM-IA method in Algorithm 2. Note that to model dynamics, we directly use the warped image $\tilde{\mathbf{x}}_t^m$ as the atom $\mathbf{d}_t^m$ in the dictionary $\mathbf{D}_t^{(m)}$. Since poorly conditioned Jacobian matrix often incurs instable problem, we propose to compute its QR factorization [54]. Concretely, instead of using $\mathbf{J}$ in Algorithm 2, we use the orthogonal $\mathbf{Q}$ and multiple $\Delta\tau_i$ by the right component of such factorization. For each image, the basis $\mathbf{B}$ remains unchanged, and the Jacobian matrix $\mathbf{J}$ as well as $(\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top$ is calculated in advance. This significantly reduces the computational complexity and offers good scalability for aligning very long sequences of images in highly demanding applications.

ORLTM-IA can also be adapted to visual tracking tasks as a robust tracker, which essentially aims to track target object in nonstationary image streams that change over time [58]. Robust visual tracking attempts to handle the unconstrained environments containing a drastic change in the appearance of the object or large lighting variation in its surroundings. This paper explores the way to dynamically model the object and the changes in motion or appearance by the proposed ORLTM, which serves as one subspace representation-based tracker. Similar trackers include Incremental Visual Tracking [58] that learns a low-dimensional subspace representation, online robust image alignment (ORIA) [55] that uses well-aligned images to linearly and sparsely reconstruct newly arrived ones after image alignment, and ORPCA-IA [56] that considers geometrical transformation in ORPCA [22]. Unlike image alignment, the initial transformation of each frame is the updated one of the previous frame for tracking.

VI. Experiments

In this section, we conduct extensive experiments on both synthetic databases and three practical vision tasks, including video background subtraction, image alignment, and visual tracking. An input video or a long image sequence can be regarded as a long third-order tensor, where one frame forms a frontal slice and a subtensor is composed of multiple frames. We employ Algorithm 1 for synthetic data analysis and video background subtraction and utilize Algorithm 2 to align images and to track targets in the presence of large illumination variations, appearance, and view angle changes. For tracking, we simply adopt the closed-form solution to update the basis dictionary. In the practical tasks, all experiments were conducted on original image pixels directly.

A. Synthetic Data Analysis

To generate synthetic data with a low-rank structure, we use *rank*-3 factor matrices, e.g., $\mathbf{S}^m \in \mathbb{R}^{50\times 3}$ (here m denotes mode), to create a tensor $\mathcal{S}$ of size $50 \times 50 \times 30$. Every factor matrix is composed of three components $[\sin(2\pi m i_m/50), \cos(2\pi m i_m/50), \text{sgn}(\sin(0.5\pi i_m))]$, where i_m indexes the column element in mode m. Each frontal slice of *mode*-3 is generated by the product of two factor matrices. The entries of the tensor are corrupted at random by small noise from normal distribution $N(0, 0.05)$ and outliers from uniform distribution $U(-|H|, |H|)$ (H is the magnitude of interval bound). We adopt root relative square error (RRSE) as the evaluation metric computed by $(\|\hat{\mathcal{S}} - \mathcal{S}\|_F / \|\mathcal{S}\|_F)$, where $\hat{\mathcal{S}}$ is the learned low-rank tensor. For evaluation, we compare four batch approaches, i.e., RPCA [3], LRR [10], [11], HOSVD [1], and HORPCA [1], and four online approaches (where the subtensor size is set to 1), i.e., ORPCA [22], online low-rank subspace clustering (OLRSC) [25], OSTD [43], and the proposed ORLTM method. It should be noted that batch methods have the hindsight knowledge of all the sequential data, thus providing performance upper bound for online counterparts. For ORLTM, $\lambda_1 = 1/(\log(n_1\, n_2))^{1/2}$, $\lambda_2 = 1$, $\lambda_3 = \lambda_1(\log(t))^{1/2}$, and we utilize the default parameters for those alternatives as indicated in original papers. For all methods, the target rank p is set to 3.

The plots in Fig. 2 depict the results obtained by varying the corruption percentage η from 0% to 100% with an interval of 10 and by changing outlier intervals $2|H|$ in $[0 : 0.2 : 2]$. It can be easily observed that our method performs better than all online approaches when the corruption percentage and the outlier interval change greatly, and even comparable to batch ones. Moreover, ORLTM outperforms LRR consistently and HORPCA when the corruption exceeds approximately 55%. These results suggest the advantages of ORLTM which leverages the low-rank data structure across all tensor modes.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

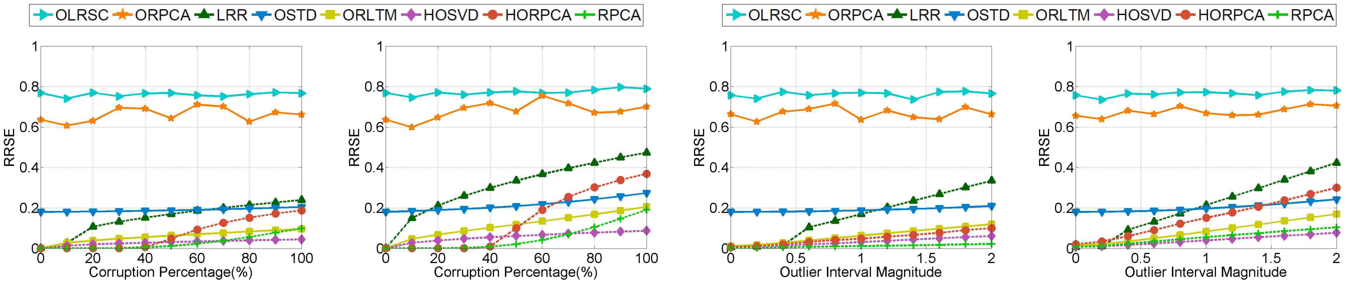


Fig. 2. RRSE averaged over 10 test runs on different corruption entries on synthetic data. Left: RRSE versus corruption percentage ($H = 0.5$ and $H = 1.0$). Right: RRSE versus outlier intervals ($2|H|$ with $\eta = 50\%$ and $\eta = 100\%$).

TABLE I

RRSE (↓) OF ONLINE METHODS WITH DIFFERENT TENSOR SIZES n_3 ($\eta = 50\%$, $H = 1$, $n_1 = 50$, AND $n_2 = 50$, AVERAGED OVER 10 TEST RUNS)

Methods	100	500	1000	5000	10000	Average
ORPCA [22]	0.6400	0.6060	0.5125	0.4745	0.4725	0.5673
OLRSC [25]	0.6731	0.5813	0.5385	0.4661	0.4324	0.5784
OSTD [43]	0.1186	0.0580	0.0431	0.0224	0.0176	0.0781
ORLTM	**0.0723**	**0.0370**	**0.0280**	**0.0156**	**0.0133**	**0.0475**

The reported results are finalized by averaging over 10 test runs due to the randomness of data generation. Furthermore, for online approaches, Table I shows the RRSE records derived from data with different sizes (n_3), and these results are also averaged over 10 test runs. As indicated in Table I, all methods exhibit more promising data recovery abilities when the tensor size increases. This has justified the fact that more unit subtensor samples contribute to achieving much improved robust tensor recovery. Simultaneously, ORLTM performs consistently the best among the compared online methods when the size of tensor data varies largely.

B. Video Background Subtraction

To subtract background of videos, we employ the low-rank component $\mathcal{L}$ to model the background (BG, i.e., static scene) while using the sparse component $\mathcal{E}$ to model the foreground (FG, i.e., walking pedestrians). The test bed is the I2R database [59] which contains a rich variety of both indoor and outdoor scenes, e.g., offices, campus, parking lot, shopping mall, restaurant, airport, and sidewalk. We use *eight* video sequences, including *Bootstrap*, *Campus*, *Curtain*, *Fountain*, *Lobby*, *Shopmall*, *Watersurface*, and *Hall*. There are totally over 15 000 video frames with a size ranging from 120 × 160 to 256 × 320. For each sequence, we have 20 ground-truth *Foreground Mask* (FM) of frames. For comparison, four online approaches, including ORPCA [22], OLRSC [25], OSTD [43], and online CP [29], and five batch approaches, including RPCA [3], LRR [11], HORPCA [1], and CP-ALS [28], were evaluated and their parameters are set according to original papers or released implementations. For ORLTM, each subtensor size is set to 1, $\lambda_1 = \alpha/(\log(n_1\ n_2))^{1/2}$ (α is 0.02 for *Curtain*, *Lobby*, and *Watersurface* and 0.1 for the rest), and $\lambda_2 = 1$ and $\lambda_3 = \lambda_1(\log(t))^{1/2}$ (t is the iteration number). The target rank p for all approaches is empirically set to 10 providing generally good performance. We capture the foreground masks by thresholding the sparse part of the frontal slice, i.e., $\text{FM}_{ij} = 1$ if $\mathbf{E}_{ij}^2/2 \geq (\text{std}(\text{vec}(\mathbf{E})))^2$, and zero otherwise, where std(·) is the standard deviation. All frames were passed *two* epochs to further refine the background modeling for online approaches. Batch approaches take in *small*-batch (500 frames) per time due to limited memory resource. For ORLTM, we utilize the bilateral random projection (BRP) [60] technique also used in [43] and [61] to initialize the dictionary $\mathbf{B}^{(m)}$ only for background subtraction. In particular, given a dense matrix $\Gamma \in \mathbb{R}^{s\times c}$, one can easily compute its low-rank approximation by $\tilde{\Gamma} = \Gamma\mathbf{Y}_1(\mathbf{Y}_2^\top\Gamma\mathbf{Y_1})^{-1}\mathbf{Y}_2^\top\Gamma$, where the rank $r \leq \min(s, c)$ and $\mathbf{Y}_1 \in \mathbb{R}^{c\times r}$, $\mathbf{Y}_2 \in \mathbb{R}^{s\times r}$ are random matrices. In addition, to examine how BRP influences the initialization, we use ORLTM2 to denote the proposed method accepting uniformly distributed random numbers for initialization. Note that LRR was found to perform very poorly when the dictionary is set to the database itself, and we thus employ its mean matrix to yield a descent performance.

To examine the performance of compared method, we use *F-score* [62] as the evaluation metric. If true positive (TP) is the total number of foreground pixels correctly classified, false positive (FP) is the total number of the pixels incorrectly classified as foreground, and false negative (FN) is the number of misclassified foreground pixels, then we define Precision = (TP/TP + FP), Recall = (TP/TP + FN), and F-score = (2 · Precision · Recall/Precision + Recall). The recorded F-score values are reported in Table II. As can be observed in Table II, ORLTM performs the best in various scenes, and the F-score is higher than the second best RPCA (batch method) by about 10%. In specific, ORLTM enjoys more promising performance on *Bootstrap*, *Curtain*, *Fountain*, *Lobby*, and *Hall* in comparison with others. Besides, our method shows the satisfying results on other sequences indicated by its almost the same F-score to that of the best ones. We attribute the superiority of ORLTM to two reasons: one is that dictionary learning can model video background better even in grossly corrupted and noisy situations and the other is that it takes full advantage of the underlying information of video sequences by learning the low-rank structures of data across all tensor modes. Here, batch methods handle 500 frames each time in the tests, marginally degrading its performance. In addition, it can be found that BRP indeed improves the performance of the proposed method on most video sequences, due to its nice low-rank approximation property.

TABLE II

F-Score (% ↑) COMPARISONS ON ALL VIDEO SEQUENCES FOR BACKGROUND SUBTRACTION. BEST IN BOLDFACE AND SECOND BEST UNDERLINED

Methods	Bootstrap	Campus	Curtain	Fountain	Lobby	Shopmall	Watersurface	Hall	Average
RPCA [3]	63.86	59.88	66.33	79.28	72.07	**75.87**	44.39	53.61	64.41
LRR [11]	55.02	32.75	84.92	62.87	35.47	70.30	61.14	55.60	57.26
HORPCA [1]	60.27	32.58	72.79	31.31	53.12	74.95	43.39	58.02	53.30
CP-ALS [28]	52.16	22.75	54.04	36.11	21.17	48.13	41.11	42.84	39.79
OCP [29]	51.43	21.99	70.12	37.48	17.27	47.36	65.71	46.91	44.78
ORPCA [22]	55.48	37.26	82.29	56.76	62.20	49.53	**89.86**	56.53	61.24
OLRSC [25]	44.75	23.01	77.76	26.96	15.54	21.49	79.16	32.53	40.15
OSTD [43]	58.06	**60.00**	56.11	71.18	40.08	74.47	48.36	61.95	58.78
ORLTM	64.33	59.20	**88.85**	**82.76**	**77.78**	74.40	89.59	**65.02**	**75.24**
ORLTM2	**64.73**	58.58	81.94	77.02	63.22	72.44	84.81	61.55	70.54

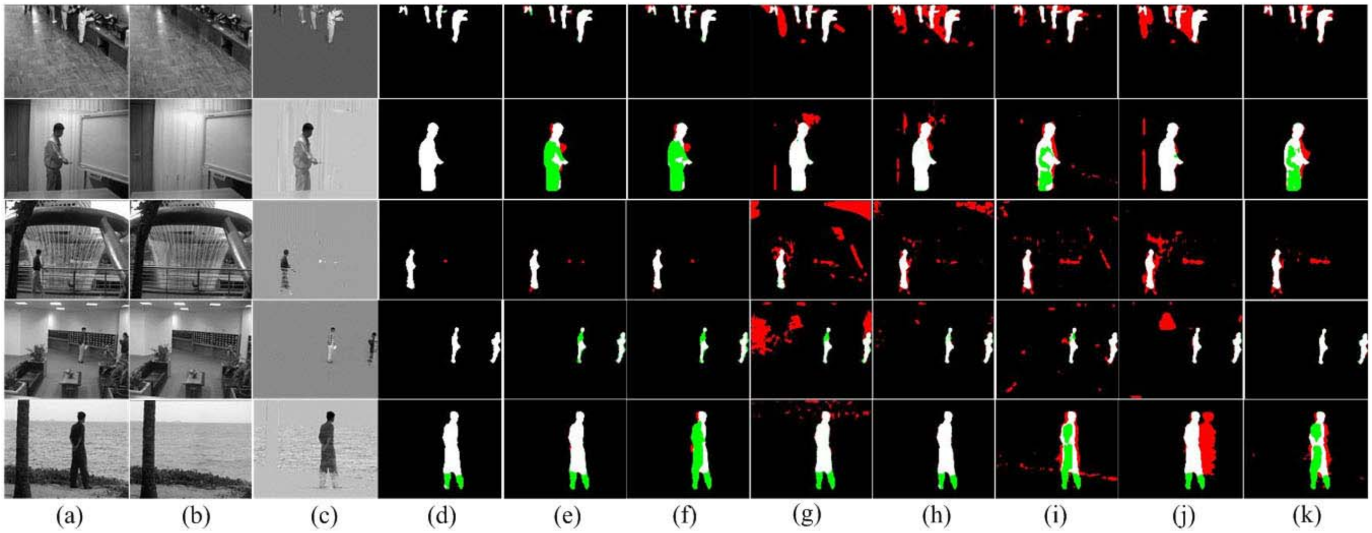


Fig. 3. Some randomly selected frame masks from video background subtraction in various scenes. (a) Input frame. (b) ORLTM (background). (c) ORLTM (foreground). (d) ORLTM. (e) ORLTM2. (f) OSTD. (g) OLRSC. (h) ORPCA. (i) HORPCA. (j) LRR. (k) RPCA. The TP pixels are in white, TN pixels in black, FP pixels in red and FN pixels in green.

TABLE III

SPEED COMPARISON (*fps* ↑) BETWEEN ORLTM AND TWO BATCH METHODS. THE VALUE IN PARENTHESES DENOTES THE ACCELERATION RATE OF ORLTM COMPARED WITH OTHERS. BEST IN BOLDFACE

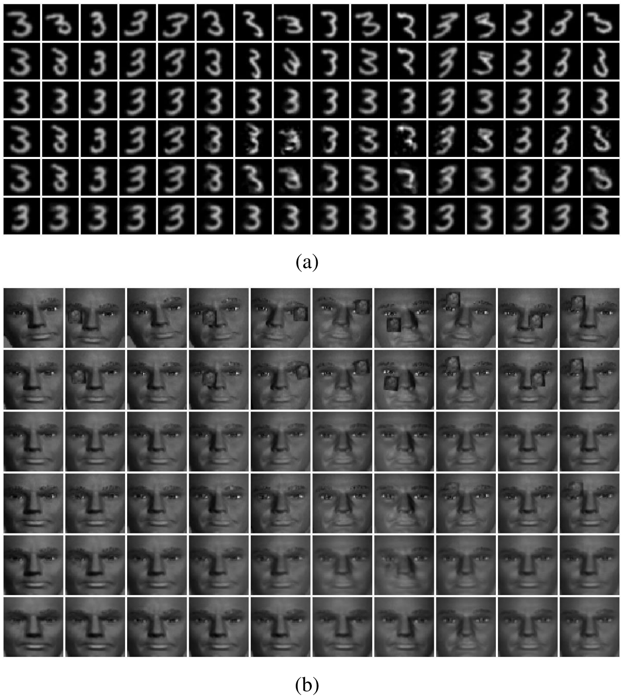
Video	$n_1 \times n_2 \times n_3$	RPCA [3]	HORPCA [1]	**ORLTM**
Bootstrap	120×160×3055	0.55 (12.2)	0.35 (18.9)	**6.64**
Campus	128×160×1439	2.22 (4.8)	0.36 (29.9)	**10.69**
Curtain	128×160×2964	2.74 (2.0)	0.40 (13.3)	**5.38**
Fountain	128×160× 523	2.56 (6.3)	0.35 (46.7)	**16.14**
Lobby	128×160×1551	2.62 (3.5)	0.25 (35.6)	**9.04**
Shopmall	256×320×1286	0.32 (8.5)	0.10 (27.0)	**2.73**
Watersurface	128×160× 641	3.46 (2.7)	0.41 (22.4)	**9.22**
Hall	144×176×3584	1.57 (2.8)	0.29 (14.9)	**4.32**

For computational efficiency, we adopt *frame rate (fps)* as the metric. The results are recorded in Table III, which demonstrates that ORLTM runs much more efficiently compared with batch ones, e.g., faster than RPCA on *Bootstrap* by over 12 times and up to 46 times quicker than HORPCA on *Fountain*. The reason is that ORLTM as an *online* algorithm only requires constantly small memory and its computational cost is very limited per time instance. In contrast, batch approaches shall store all frames in memory and singular value decompositions on batch frames need intensive computations. The tests were conducted on a machine with 3.06 GHz Core X5676 processor and 24-GB RAM.

To illustrate background subtraction, we display some foreground masks of those frames randomly chosen in Fig. 3. Here, these images have gone through binarization, i.e., the pixel values are converted to 0 or 255 for better visualization. Compared with the ground-truth mask, TP, true-negative (TN), FP, and FN pixels have been marked by different colors. As vividly shown in Fig. 3, our method captures much better masks than the rest in most situations, e.g., swinging curtain (*row* 2), running fountain (*row* 3), and varying illumination (*row* 4). This further validates the fact that ORLTM can provide more satisfying background and foreground modeling in various scenes.

C. Image Alignment

To evaluate ORLTM-IA, we conducted tests on both controlled images including *Al Gore*, *Digit 3*, and *Dummy* and unconstrained images taken from LFW [63] database including *Ariel Sharon*, *Colin Powell*, *George W. Bush*, *Laura Bush*, and *Hugo Chavez*. *Al Gore* talking video contains 140 frames with strong jitter across frames; *Digit-3* consists of 100 handwritten digits originally from the MNIST data set; *Dummy*



(a)

(b)

Fig. 4. Image alignment results for (a) Digit 3 and (b) dummy images. Row 1: unaligned image. Row 2: aligned image by ORLTM-IA. Row 3: low-rank image by RASL. Row 4: low-rank image by ORIA. Row 5: low-rank image by ORPCA-IA. Row 6: low-rank image by ORLTM-IA.

contains 100 perturbed and occluded images; LFW images suffer from large changes in pose and facial expression as well as illumination or occlusion variations. These images and corresponding initial transformation estimates are all provided in [54], and we adopt the same canonical image size, i.e., 29×29 for digits, 49×49 for dummy images, and 80×60 for face images. We use affine transformation $\mathbb{G} = \text{Aff}(2)$ for image geometrical warping. For comparison, we tested two online methods, including ORIA [55] and ORPCA-IA [56], in addition to one batch method Robust Alignment by Sparse and Low-Rank Decomposition (RASL) [54]. We set the parameters of compared methods as suggested in original works. For ORLTM, we set $\lambda_1 = \lambda_3 = 1/\sqrt{d}$ and $\lambda_2 = 1$; for all methods, the first 10 aligned images by RASL are utilized to initialize the basis (the rank $p = 10$).

To visualize results, some samples are randomly selected, and the aligned images are displayed in Figs. 4 and 5. From Fig. 4, one can observe that ORLTM-IA can well align the digits despite very different styles and orientations (*second row*), and the recovered low-rank images (*sixth row*) are more consistent than the compared ones. These results clearly justify that ORLTM-IA can well model and preserve the low-rank structures of these images. Moreover, the dummy faces with serious illumination variations and occlusions can be aligned nicely using the proposed method. Fig. 5 shows that ORLTM-IA provides much better alignments than the other two online methods (ORIA and ORPCA-IA), and its performance is even comparable to the batch method, RASL. This can be attributed to two advantages of ORLTM-IA: 1) our method utilizes the spatial information across different modes and can better capture the geometrical transformations of the unaligned images and 2) the introduced dictionary component is able to enhance the low-rank modeling ability.

For quantitative evaluation, we use the eye slop/angle and mouth slop/angle as the metric to compute concrete alignment accuracy. If the slop is ρ, the angle can be computed as $\omega = (\arctan(\rho) \cdot 360/2\pi)$. In detail, we first apply the facial point detector using a deep convolutional network model [64] to detect five points on both unaligned faces and aligned faces by the compared methods. Among these detected points, we utilize two eye center points to compute the eye-line slop/angle and two mouth corner points to compute the mouth-line slop/angle. Ideally, we expect the two slops or angles to be as small as possible, indicating better aligned images in a local view. However, occasionally, the detector fails to detect the points on all the facial images, because the bounding boxes cannot be located on some aligned images in a canonical size. Hence, we report the averaged absolute results of three successfully detected face data, i.e., *Al Gore*, *George W. Bush*, and *Laura Bush*, in Table IV. As seen in Table IV, all the image alignment methods can reduce the slop and the angle of both eye and mouth lines on average. ORLTM-IA can best rectify the eye line and the mouth line, since it can maximally reduce the bias angle $|\omega|$ from 7.63° to 3.76° for the former and from 7.08° to 3.54° for the latter on *Laura Bush*. However, except our method, the remaining ones perform poorly on the video data *Al Gore*, which might because the data have been roughly aligned and the low-rank structure of the video is better captured by robust tensor learning.

D. Visual Tracking

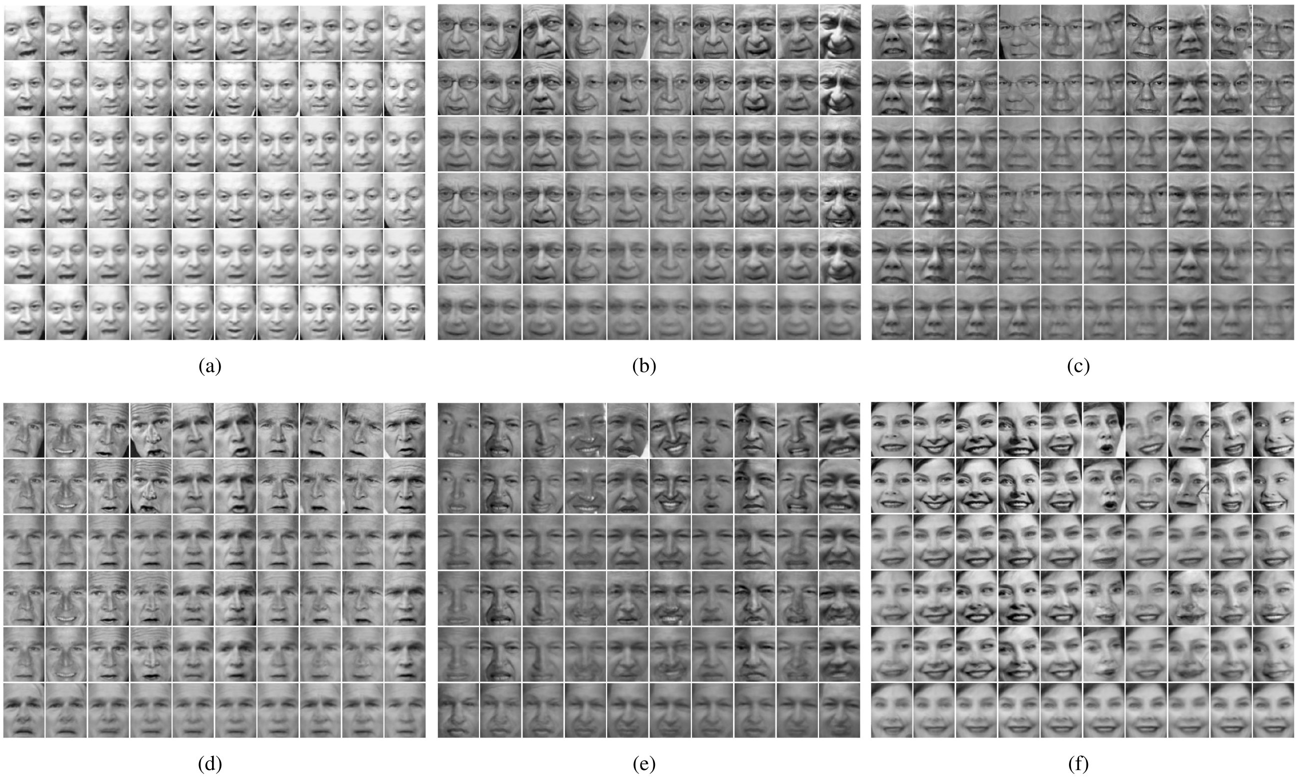
The task of visual tracking is generally to track the target by a bounding box, which is manually initialized in the first frame. Regarding the target, its initial transformation in the current frame is the estimated transformation of the same target in the previous frame. Here, we adopt the proposed online image sequence alignment strategy, ORLTM-IA, in Algorithm 2 to update the model for visual tracking. We compare our tracker with two online subspace trackers, i.e., ORIA [55] and ORPCA-IA [56], on 15 challenging video sequences [65] where the targets may undergo partial occlusions and pose, illumination, or appearance changes. The parameters of compared ones are set by default in the original papers, and we use $\lambda_1 = \lambda_3 = 1/\sqrt{d}$, $\lambda_2 = 1$, and affine transformations $\mathbb{G} = \text{Aff}(2)$ for our method. For all methods, we generate the initial basis or dictionary from the first frame, which provides the initial location of the target whose size is set as the canonical image size.

To evaluate the tracking performance, we adopt three widely used metrics, including *Overlap Ratio*, *Center Error*, and *Frame Rate*. *Overlap Ratio* is defined as $(\text{area}_{\text{es}} \cap \text{area}_{\text{gt}}/\text{area}_{\text{es}} \cup \text{area}_{gt})$, where the numerator is the intersection of the ground-truth bounding box and the estimated one while the denominator is the union of the two boxes; *Center Error* calculates the averaged Euclidean distance between the estimated center and the ground truth of those bounding boxes in all consecutive frames of a given video; *Frame Rate* computes the number of frames handled by the method in 1 s. These records are collected by a

TABLE IV

Quantitative Image Alignment Results in Terms of Eye and Mouth *Slop* $|\rho|(\downarrow)$ / *Angle* $|\omega|(\downarrow)$ on the Face Data Sets. Note That RASL Is a Batch Method While the Rest Ones Are Online Methods; "ORIGIN" Denotes the Originally Unaligned Images. The Smaller the Slop and the Angle, the Better the Facial Images Are Aligned. Best in Bold Face

Methods	Al Gore		George W. Bush		Laura Bush		Average	
	Eye	Mouth	Eye	Mouth	Eye	Mouth	Eye	Mouth
ORIGIN	0.0833 / 4.76	0.0657 / 3.76	0.1016 / 5.80	0.0993 / 5.67	0.1340 / 7.63	0.1243 / 7.08	0.1063 / 6.06	0.0964 / 5.50
RASL [54]	0.0925 / 5.29	0.0728 / 4.16	**0.0354 / 2.03**	**0.0326 / 1.87**	0.1116 / 6.37	0.0955 / 5.45	0.0798 / 4.56	0.0670 / 3.83
ORIA [55]	0.0952 / 5.44	0.0734 / 4.20	0.0391 / 2.24	0.0463 / 2.65	0.0969 / 5.53	0.0770 / 4.41	0.0771 / 4.40	0.0656 / 3.75
ORPCA-IA [56]	0.0940 / 5.37	0.0771 / 4.41	0.0508 / 2.91	0.0445 / 2.55	0.1025 / 5.85	0.0861 / 4.92	0.0824 / 4.71	0.0692 / 3.96
ORLTM-IA	**0.0430 / 2.46**	**0.0343 / 1.97**	0.0463 / 2.65	0.0535 / 3.06	**0.0657 / 3.76**	**0.0619 / 3.54**	**0.0517 / 2.96**	**0.0499 / 2.86**


(a) (b) (c)

(d) (e) (f)

Fig. 5. Image alignment results of facial images. (a) AI Gore. (b) Ariel Sharon. (c) Colin Powell. (d) George W. Bush. (e) Laura Bush. (f) Hugo Chavez. Row 1: unaligned image. Row 2: aligned image by ORLTM-IA. Row 3: low-rank image by RASL. Row 4: low-rank image by ORIA. Row 5: low-rank image by ORPCA-IA. Row 6: low-rank image by ORLTM-IA.

machine with 2.20 GHz Dual Core i5-5200U processor and 12-GB RAM. The results of the compared methods are reported in Table V, where Win/loss/tie reveals how well the method is competing, and high win or low loss represents success. The records in Table V indicate that our method achieves the most promising performance and still offers processing speed as fast as 25 frames/s on *CarDark* and *Man*, satisfying the real-time requirements. ORLTM-IA outperforms the second best ORPCA-IA by a significant percentage of 7.6 in terms of *Overlap Ratio* and meanwhile reduces the center error by a large magnitude of 8.5. This is because leveraging the information in all modes of data does strengthen the ability of ORLTM-IA to well model the pose, illumination, view angle changes, and even large shading in video frames. Moreover, we find that both ORLTM-IA and ORIA run almost twice faster than ORIA, due to the much slower convergence of objective function for ORIA to update the basis.

The bounding boxes of tracking results on some videos are shown in Fig. 6. Our method coded in red rectangle can robustly track the face in spite of large occlusion by the book on *FaceOcc1* (row 1). In contrast, ORIA fails to track the target on *Girl* sequence since frame #109 while ORPCA-IA drifts off since frame #69 when the head turns (row 2). For *Skating1* sequence, ORIA drifts after frame #83 when the rest ones perform better due to the satisfying basis updated by the tracker; for *Suv* video, ORIA drifts since frame #526 and ORPCA-IA drifts since frame #730 as displayed in *row 3*, but our method can track the vehicle until the end though there exists tolerable center bias. This suggests that low-rank tensor modeling can learn dynamics during the harsh circumstances, such as large occlusion by the tree or other

TABLE V

TRACKING RESULTS IN TERMS OF OVERLAP RATIO (↑), CENTER ERROR (↓), AND FRAME RATE (↑) ON VIDEO SEQUENCES. THE NUMBER OF FRAMES IN EACH VIDEO IS IN PARENTHESES. WIN/LOSS/TIE COMPARES THE CURRENT RECORD WITH THE BEST ONE IN BOLD FACE

Datasets	ORIA [55]			ORPCA-IA [56]			**ORLTM-IA**		
	Overlap Ratio	Center Error	Frame Rate	Overlap Ratio	Center Error	Frame Rate	Overlap Ratio	Center Error	Frame Rate
Car1 (1020)	0.582	184.391	4.904	0.797	1.434	**17.481**	**0.820**	**1.172**	15.367
CarDark (393)	0.462	24.993	14.369	**0.848**	**1.228**	22.919	0.835	1.414	**25.820**
Crowds (327)	0.096	308.262	11.684	0.731	3.757	10.281	**0.755**	**3.284**	**11.997**
Coupon (347)	0.911	2.631	3.729	**0.914**	**2.396**	**16.026**	0.908	2.660	14.288
Dancer2 (150)	**0.778**	**7.286**	2.600	0.771	8.528	**4.675**	0.770	8.432	3.078
Dog1 (1350)	**0.672**	9.510	7.089	0.667	9.727	14.729	**0.672**	**9.376**	**14.963**
FaceOcc1 (892)	0.670	22.451	0.833	0.644	26.466	**4.209**	**0.729**	**17.604**	3.154
Girl (500)	0.211	26.959	6.069	0.109	41.774	**19.056**	**0.496**	**18.960**	12.384
Man (134)	0.863	1.742	13.232	**0.890**	1.515	24.680	**0.890**	**1.459**	**25.376**
Mhyang (1490)	0.793	2.839	3.536	0.784	3.418	**10.097**	**0.866**	**2.455**	9.765
Moun.Bike (228)	0.573	**13.885**	4.594	0.474	25.935	6.748	**0.578**	15.740	**7.760**
Skating1 (400)	0.154	105.756	2.922	0.531	**28.008**	**4.351**	**0.541**	30.220	3.980
Suv (945)	0.463	87.239	6.656	0.574	43.708	**17.422**	**0.596**	**23.618**	14.995
Trellis (569)	0.521	17.286	1.313	0.192	74.170	**3.285**	**0.616**	**7.453**	1.894
Walking (412)	0.170	100.576	8.112	**0.632**	**3.789**	9.018	0.625	4.180	**10.280**
Average	0.527	61.053	6.109	0.637	18.390	**12.331**	**0.713**	**9.868**	11.673
Win↑/loss↓/tie	1/13/1	2/13/0	0/15/0	4/11/0	4/11/0	**9/5/0**	**10/3/2**	**9/6/0**	6/9/0

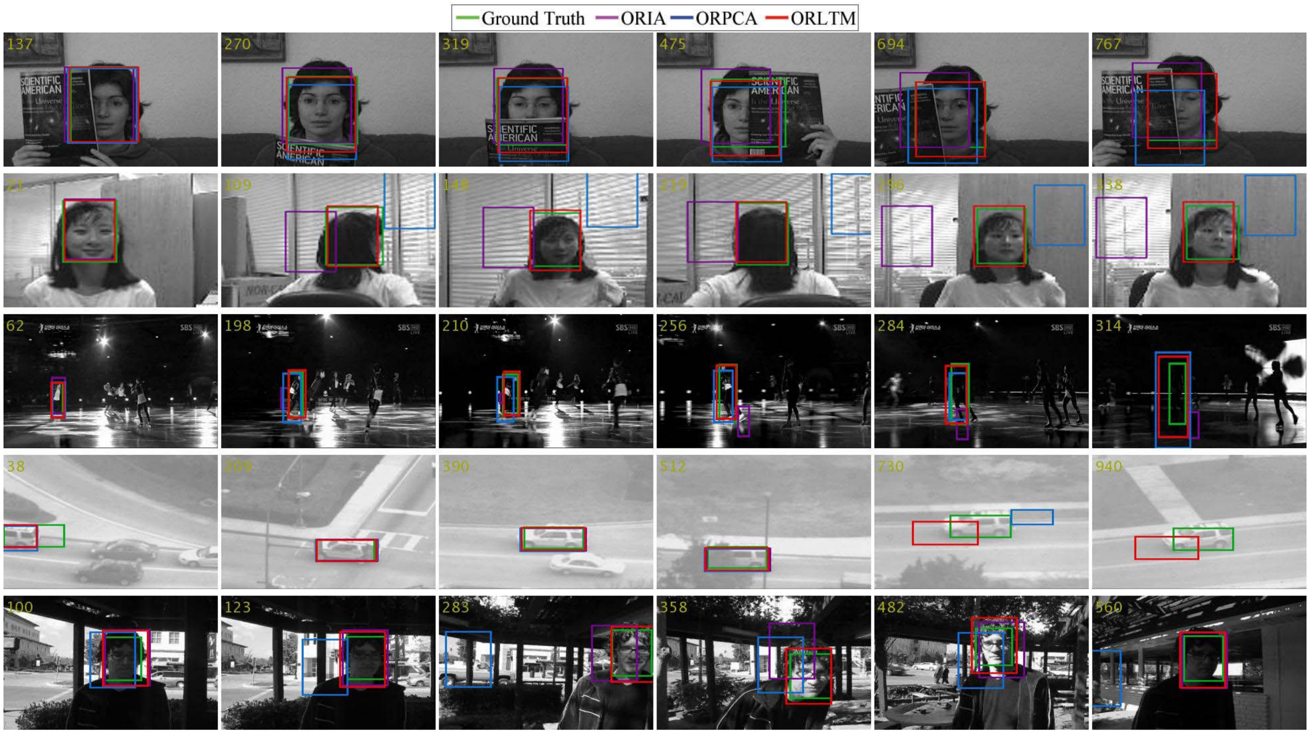


Fig. 6. Bounding boxes yielded by different trackers in frames. Row 1: FaceOcc1. Row 2: Girl. Row 3: Skating1. Row 4: Suv. Row 5: Trellis.

vehicles. On the *Trellis* sequence, ORIA drifts since frame #387 while ORPCA-IA losts the target since frame #192 due to the drastic motion or appearance variations of the target, which however can be nicely handled by our tracker. Overall, ORLTM-IA enables more favorable tracking results as well as the fast speed on these sequences in comparison with other methods.

VII. CONCLUSION

In this paper, we developed an ORLTM approach for learning low-rank structures of streaming noisy tensor data. ORLTM can handle samples in a sequential way via the equivalent bifactor formulation of the nuclear norm, which makes it possible to process large-scale streaming tensor data, such as image and video sequences. The objective function is reformulated as a nonconvex problem and solved by stochastic optimization. In contrast to batch methods, our method reduces memory consumption by a factor of n in each iteration. Moreover, we extend the proposed method to image alignment scenario and also adapt it for visual object tracking. Extensive experimental results have demonstrated that our method gains significant advantages and promising performances on synthetic data and several practical tasks. Empirical studies on synthetic data have shown more promising results of the proposed method in comparison with the well-established online approaches and ORLTM is even comparable to batch alternatives though the tensor data are grossly corrupted by noise. Practical studies have validated the superior performance of ORLTM in video background subtraction. Besides, ORLTM-IA achieves favorable image alignment results and also enables robustly tracking the target in consecutive frames in the presence of large variations, motion changes, and partial occlusions.

References

[1] D. Goldfarb and Z. Qin, "Robust low-rank tensor recovery: Models and algorithms," *SIAM J. Matrix Anal. Appl.*, vol. 35, no. 1, pp. 225–253, 2014.

[2] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, "Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5249–5257.

[3] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, 2011, Art. no. 11.

[4] P. Li, J. Feng, X. Jin, L. Zhang, X. Xu, and S. Yan, "Online robust low-rank tensor learning," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 2180–2186.

[5] S. Javed, A. Mahmood, T. Bouwmans, and S. K. Jung, "Spatiotemporal low-rank modeling for complex scene background initialization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1315–1329, Jun. 2018, doi: 10.1109/TCSVT.2016.2632302.

[6] T. Bouwmans and E. H. Zahzah, "Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance," *Comput. Vis. Image Understand.*, vol. 122, pp. 22–34, May 2014.

[7] C. Lang, J. Feng, S. Feng, J. Wang, and S. Yan, "Dual low-rank pursuit: Learning salient features for saliency detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 6, pp. 1190–1200, Jun. 2016.

[8] W. He, H. Zhang, L. Zhang, and H. Shen, "Total-variation-regularized low-rank matrix factorization for hyperspectral image restoration," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 1, pp. 178–188, Jan. 2016.

[9] T. Bouwmans, A. Sobral, S. Javed, S. K. Jung, and E.-H. Zahzah, "Decomposition into low-rank plus additive matrices for background/foreground separation: A review for a comparative evaluation with a large-scale dataset," *Comput. Sci. Rev.*, vol. 23, pp. 1–71, Feb. 2016.

[10] G. Liu, Q. Liu, and P. Li, "Blessing of dimensionality: Recovering mixture data via dictionary pursuit," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 1, pp. 47–60, Jan. 2017.

[11] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171–184, Jan. 2013.

[12] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1615–1622.

[13] Y. Zhang, Z. Jiang, and L. S. Davis, "Learning structured low-rank representations for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 676–683.

[14] M. Yin, J. Gao, Z. Lin, Q. Shi, and Y. Guo, "Dual graph regularized latent low-rank representation for subspace clustering," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 4918–4933, Dec. 2015.

[15] M. Yin, J. Gao, and Z. Lin, "Laplacian regularized low-rank representation and its applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 3, pp. 504–517, Mar. 2016.

[16] S. Li and Y. Fu, "Learning robust and discriminative subspace with low-rank constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2160–2173, Nov. 2016.

[17] P. Li, J. Yu, M. Wang, L. Zhang, D. Cai, and X. Li, "Constrained low-rank learning using least squares-based regularization," *IEEE Trans. Cybern.*, vol. 47, no. 12, pp. 4250–4262, Dec. 2017.

[18] P. Zhou, Z. Lin, and C. Zhang, "Integrated low-rank-based discriminative feature learning for recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 5, pp. 1080–1093, May 2015.

[19] J. Shen and P. Li, "Learning structured low-rank representation via matrix factorization," in *Proc. 19th Int. Conf. Artif. Intell. Statist.*, 2016, pp. 500–509.

[20] K. Tang, R. Liu, Z. Su, and J. Zhang, "Structure-constrained low-rank representation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2167–2179, Dec. 2014.

[21] S. Xiao, M. Tan, D. Xu, and Z. Y. Dong, "Robust kernel low-rank representation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2268–2281, Nov. 2016.

[22] J. Feng, H. Xu, and S. Yan, "Online robust PCA via stochastic optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 404–412.

[23] J. Shen, H. Xu, and P. Li, "Online optimization for max-norm regularization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1718–1726.

[24] J. Zhan, B. Lois, H. Guo, and N. Vaswani, "Online (and offline) robust PCA: Novel algorithms and performance guarantees," in *Proc. 19th Int. Conf. Artif. Intell. Statist.*, 2016, pp. 1488–1496.

[25] J. Shen, P. Li, and H. Xu, "Online low-rank subspace clustering by basis dictionary pursuit," in *Proc. 33rd Int. Conf. Int. Mach. Learn.*, vol. 48, 2016, pp. 622–631.

[26] W. Hu, Y. Yang, W. Zhang, and Y. Xie, "Moving object detection using tensor-based low-rank and saliently fused-sparse decomposition," *IEEE Trans. Image Process.*, vol. 26, no. 2, pp. 724–737, Feb. 2017.

[27] N. B. Erichson, K. Manohar, S. L. Brunton, and J. N. Kutz. (2017). "Randomized CP tensor decomposition." [Online]. Available: https://arxiv.org/abs/1703.09074

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

[29] S. Zhou, N. X. Vinh, J. Bailey, Y. Jia, and I. Davidson, "Accelerating online CP decompositions for higher order tensors," in *Proc. SIGKDD*, 2016, pp. 1375–1384.

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

[31] W. Cao *et al.*, "Total variation regularized tensor RPCA for background subtraction from compressive measurements," *IEEE Trans. Image Process.*, vol. 25, no. 9, pp. 4075–4090, Sep. 2016.

[32] Y. Fu, J. Gao, D. Tien, Z. Lin, and X. Hong, "Tensor LRR and sparse coding-based subspace clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 10, pp. 2120–2133, Oct. 2016.

[33] H. Tan, B. Cheng, J. Feng, G. Feng, and Y. Zhang, "Tensor recovery via multi-linear augmented Lagrange multiplier method," in *Proc. 6th Int. Conf. Image Graph.*, Apr. 2011, pp. 141–146.

[34] H. Tan, B. Cheng, J. Feng, G. Feng, W. Wang, and Y.-J. Zhang, "Low-n-rank tensor recovery based on multi-linear augmented Lagrange multiplier method," *Neurocomputing*, vol. 119, pp. 144–152, Nov. 2013.

[35] Q. Zhao, G. Zhou, L. Zhang, A. Cichocki, and S.-I. Amari, "Bayesian robust tensor factorization for incomplete multiway data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 4, pp. 736–748, Apr. 2016.

[36] T. Yokota, N. Lee, and A. Cichocki, "Robust multilinear tensor rank estimation using higher order singular value decomposition and information criteria," *IEEE Trans. Signal Process.*, vol. 65, no. 5, pp. 1196–1206, Mar. 2017.

[37] P. Zhou and J. Feng, "Outlier-robust tensor PCA," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3938–3946.

[38] 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.

[39] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 208–220, Jan. 2013.

[40] C. Qiu, X. Wu, and H. Xu, "Recursive projected sparse matrix recovery (ReProSMR) with application in real-time video layer separation," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2014, pp. 1332–1336.

[41] Z. Zhang, D. Liu, S. Aeron, and A. Vetro, "An online tensor robust PCA algorithm for sequential 2D data," in *Proc. IEEE Conf. Acoust., Speech Signal Process.*, 2016, pp. 2434–2438.

[42] H. Kasai, "Online low-rank tensor subspace tracking from incomplete data by CP decomposition using recursive least squares," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2016, pp. 2434–2438.

[43] 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 *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Dec. 2015, pp. 946–953.

[44] R. Yu, D. Cheng, and Y. Liu, "Accelerated online low-rank tensor learning for multivariate spatio-temporal streams," in *Proc. 32nd Int. Conf. Int. Conf. Mach. Learn.*, 2015, pp. 238–247.

[45] C. J. Hillar and L.-H. Lim, "Most tensor problems are NP-hard," *J. ACM*, vol. 60, no. 6, 2013, Art. no. 45.

[46] N. Srebro, J. D. M. Rennie, and T. S. Jaakkola, "Maximum-margin matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 17, 2004, pp. 1329–1336.

[47] M. Fazel, H. Hindi, and S. P. Boyd, "A rank minimization heuristic with application to minimum order system approximation," in *Proc. Amer. Control Conf.*, vol. 6. Jun. 2001, pp. 4734–4739.

[48] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM Rev.*, vol. 52, no. 3, pp. 471–501, 2010.

[49] E. T. Hale, W. Yin, and Y. Zhang, "Fixed-point continuation for ℓ_1-minimization: Methodology and convergence," *SIAM J. Optim.*, vol. 19, no. 3, pp. 1107–1130, 2008.

[50] S. J. Wright, "Coordinate descent algorithms," *Math. Program.*, vol. 151, no. 1, pp. 3–34, Jun. 2015.

[51] A. Mensch, J. Mairal, B. Thirion, and G. Varoquaux, "Dictionary learning for massive matrix factorization," in *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn.*, vol. 48, 2016, pp. 1737–1746.

[52] P. Richtárik and M. Takác, "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function," *Math. Program.*, vol. 144, nos. 1–2, pp. 1–38, 2014.

[53] M. Mardani, G. Mateos, and G. B. Giannakis, "Subspace learning and imputation for streaming big data matrices and tensors," *IEEE Trans. Signal Process.*, vol. 63, no. 10, pp. 2663–2677, May 2015.

[54] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2233–2246, Nov. 2012.

[55] Y. Wu, B. Shen, and H. Ling, "Online robust image alignment via iterative convex optimization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1808–1814.

[56] W. Song, J. Zhu, Y. Li, and C. Chen, "Image alignment by online robust PCA via stochastic gradient descent," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 7, pp. 1241–1250, Jul. 2016.

[57] X. Zhang, D. Wang, Z. Zhou, and Y. Ma, "Simultaneous rectification and alignment via robust recovery of low-rank tensors," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 1637–1645.

[58] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, 2008.

[59] L. Li, W. Huang, I. Y.-H. Gu, and Q. Tian, "Statistical modeling of complex backgrounds for foreground object detection," *IEEE Trans. Image Process.*, vol. 13, no. 11, pp. 1459–1472, Nov. 2004.

[60] T. Zhou and D. Tao, "Bilateral random projections," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2012, pp. 1286–1290.

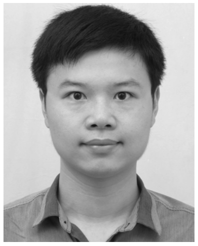
[61] T. Zhou and D Tao, "GoDec: randomized low-rank & sparse matrix decomposition in noisy case," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 33–40.

[62] S. Brutzer, B. Höferlin, and G. Heidemann, "Evaluation of background subtraction techniques for video surveillance," in *Proc. Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 1937–1944.

[63] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," in *Proc. Workshop Faces RealLife Images, Detection, Alignment, Recognit.*, Marseille, France, Oct. 2008. [Online]. Available: https://hal.inria.fr/inria-00321923

[64] Y. Sun, X. Wang, and X. Tang, "Deep convolutional network cascade for facial point detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3476–3483.

[65] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.



Ping Li received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China.

He was a Post-Doctoral Researcher with the Learning and Vision Group, Department of Electrical and Computer Engineering, National University of Singapore, Singapore. He is currently an Associate Professor with the School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou. His current research interests include machine learning, computer vision, and multimedia computing.

Jiashi Feng received the Ph.D. degree from the National University of Singapore, Singapore.

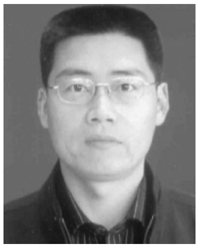
He was a Post-Doctoral Research Fellow, University of California at Berkeley, Berkeley, CA, USA. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, National University of Singapore. His current research interests include machine learning in general, including deep learning, robust learning, subspace learning, large-scale machine learning and their applications in computer vision, big data analysis, and artificial intelligence.

Xiaojie Jin is currently pursuing the Ph.D. degree with the NUS Graduate School for Integrative Science and Engineering, National University of Singapore, Singapore.

His current research interests include deep learning and computer vision.

Luming Zhang received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China.

He is currently with the College of Computer Science, Zhejiang University. His current research interests include visual perception analysis, image enhancement, and pattern recognition.



Xianghua Xu received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China.

He is currently a Professor with the School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou. He has authored or co-authored over 100 peer-reviewed journal and conference papers. His current research interests include data mining, wireless networks, and parallel and distributed computing.

Dr. Xu was a recipient of the Best Paper Award at the 2012 IEEE International Symposium on Workload Characterization.

Shuicheng Yan (F'17) is currently the Vice President and the Chief Scientist of Qihoo 360 Technology Co., Ltd., Beijing, China, and the Head of the 360 Artificial Intelligence Institute, Beijing. He is also a tenured Associate Professor with the National University of Singapore, Singapore. He has authored/co-authored about 500 high-quality technical papers, with Google Scholar citation over 25 000 times and H-index 70. His current research interests include computer vision, machine learning, and multimedia analysis.

Dr. Yan is an International Association for Pattern Recognition Fellow and an Association for Computing Machinery (ACM) Distinguished Scientist. His team received seven times winner or honorable-mention prizes in five years over the PASCAL Visual Object Classes and the ImageNet Large Scale Visual Recognition Challenge competitions which are the core competitions in the field of computer vision, along with over 10 times best (student) paper awards and especially a Grand Slam at the ACM Multimedia, the top conference in the field of multimedia, including the Best Paper Award, the Best Student Paper Award, and the Best Demo Award. He is a Thomson Reuters Highly Cited Researcher of 2014–2016.