mathematics

MDPI

Article

TensorTrack: Tensor Decomposition for Video Object Tracking

Yuntao Gu [1], Pengfei Zhao [2], Lan Cheng [1], Yuanjun Guo [3,*], Haikuan Wang [2], Wenjun Ding [4] and Yu Liu [4]

[1] College of Electrical and Power Engineering, Taiyuan University of Technology, Taiyuan 030024, China; guyuntao789@gmail.com (Y.G.); chenglan@tyut.edu.cn (L.C.)
[2] School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200444, China; zhaopengfei@shu.edu.cn (P.Z.); hkwang@shu.edu.cn (H.W.)
[3] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences Shenzhen, Shenzhen 518055, China
[4] China Construction Third Bureau Digital Engineering Co., Ltd., Shenzhen 518106, China; dingwenjun188@126.com (W.D.); liu-yu@cscec.com (Y.L.)
* Correspondence: yj.guo@siat.ac.cn

Abstract: Video Object Tracking (VOT) is a critical task in computer vision. While Siamese-based and Transformer-based trackers are widely used in VOT, they struggle to perform well on the OTB100 benchmark due to the lack of dedicated training sets. This challenge highlights the difficulty of effectively generalizing to unknown data. To address this issue, this paper proposes an innovative method that utilizes tensor decomposition, an under-explored concept in object-tracking research. By applying L1-norm tensor decomposition, video sequences are represented as four-mode tensors, and a real-time background subtraction algorithm is introduced, allowing for effective modeling of the target–background relationship and adaptation to environmental changes, leading to accurate and robust tracking. Additionally, the paper integrates an improved multi-kernel correlation filter into a single frame, locating and tracking the target by comparing the correlation between the target template and the input image. To further enhance localization precision and robustness, the paper also incorporates Tucker2 decomposition to integrate appearance and motion patterns, generating composite heatmaps. The method is evaluated on the OTB100 benchmark dataset, showing significant improvements in both performance and speed compared to traditional methods. Experimental results demonstrate that the proposed method achieves a 15.8% improvement in AUC and a ten-fold increase in speed compared to typical deep learning-based methods, providing an efficient and accurate real-time tracking solution, particularly in scenarios with similar target–background characteristics, high-speed motion, and limited target movement.

Keywords: video object tracking; correlation filter; tensor decomposition; background subtraction algorithm

MSC: 68T10

check for updates

Academic Editor: Juan Gabriel Avina-Cervantes

Received: 10 January 2025
Revised: 3 February 2025
Accepted: 7 February 2025
Published: 8 February 2025

Citation: Gu, Y.; Zhao, P.; Cheng, L.; Guo, Y.; Wang, H.; Ding, W.; Liu, Y. TensorTrack: Tensor Decomposition for Video Object Tracking. *Mathematics* **2025**, *13*, 568. https://doi.org/10.3390/math13040568

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Object tracking plays a vital role in various applications, enabling the reliable identification and localization of objects in real-world scenarios. Its importance extends across a range of fields, including but not limited to security surveillance, industrial automation, autonomous vehicles, and augmented reality. Object tracking is an essential aspect of ensuring efficient and safe operation, enhancing user experiences, and improving overall performance. With advancements in computer vision and machine-learning techniques, object tracking continues to evolve, driving the creation of new and innovative applications.

Traditional methods for object tracking have typically relied on the use of correlation filters and hand-crafted features or heuristics to locate and track objects. Correlation filters are filters that operate by comparing the correlation of an object template with the input image to locate and track the object. However, these filters are often limited in their ability to handle complex object appearances, changes in illumination, background clutter, and occlusion, making them unreliable in real-life scenarios. To address these limitations, researchers have turned to machine-learning techniques such as tensor decomposition for more robust and effective object tracking. Tensor decomposition can learn a model of the object's appearance and adapt to changes in the environment, improving accuracy and efficiency.

Although correlation filters have gained popularity for their effectiveness and ease of implementation, tensor decomposition remains an emerging technique in the domain of robust object tracking. However, recent studies have demonstrated the versatility of tensor decomposition in different fields, including spatiotemporal hotspot detection, stress recognition [1], georeferenced data analysis [2], and low-rank tensor approximation using compressive sensing [3]. These applications showcase the potential and capabilities of tensor decomposition for solving real-world problems. While not as widely used in object tracking as correlation filters, tensor decomposition remains a promising technique for improving the accuracy and robustness of object-tracking algorithms. Therefore, further research is necessary to explore its full potential in this domain.

A video sequence can be represented as a four-mode tensor of dimensions $w \times h \times c \times b$, where w and h denote the width and height of each frame, c represents the number of color channels, and b corresponds to the frame index of the current image. Tensor decomposition aims to reveal the intrinsic structure of high-dimensional data, providing an effective mathematical framework for video data modeling and analysis. Since the target object in a video sequence is typically distinguishable from the background, video data exhibits a high degree of structural regularity, making it particularly suitable for tensor decomposition techniques. By leveraging tensor decomposition, a low-dimensional intrinsic representation of the data can be extracted, enhancing both interpretability and computational efficiency in video processing and analysis.

The majority of existing correlation filter (CF) methods in single-object tracking (SOT) focus on configuring various features within a single frame, such as those found in [4–8]. Some methods even incorporate deep perceptive pooling (DPP) features into the CF method, as seen in [9]. Although many of these methods have achieved state-of-the-art (SOTA) results on OTB100, they have not been able to compete with Deep Neural Networks (DNNs) in terms of processing speed.

Through extensive experimentation and evaluation, the effectiveness of the proposed method has been demonstrated, particularly in improving tracking robustness across various challenging scenarios. The integration of tensor decomposition with correlation-based tracking enhances the adaptability of the framework, allowing it to handle complex tracking conditions more effectively. By introducing a novel decomposition strategy into video object tracking, this work advances the field and lays the foundation for future research. The key contributions of this paper are as follows:

1. Developed a novel framework that integrates tensor decomposition techniques with single-object tracking methods, achieving superior performance even on platforms with limited computational resources.

2. Designed a fast and efficient algorithm based on Real-Time Tensor-Based Background Subtraction, leveraging tensor decomposition to dynamically learn the target's appearance model and quickly adapt to changing environments.

3. Integrated appearance and motion patterns via Tucker2 decomposition, generating composite heatmaps that enhance localization precision and robustness in visually ambiguous, high-motion, and low-motion scenarios.

2. Related Work

Recent advances in object-tracking approaches have been heavily influenced by CF methods, which have been widely applied since the pioneering work by Bolme in 2010 [10]. Over the past eight years, CF methods have gained popularity SOT [4]. Various improved versions of CF algorithms have been proposed to enhance tracking performance. For instance, Tang et al. [5] developed a tracker based on Multi-Kernel Correlation Filters (MKCF), leveraging the discriminatory power spectra of different features. Lukezic and Vizivc [6] introduced a novel formulation that effectively handles visual and geometric constraints using a constellation model of CFs. Moreover, Nikouei et al. [11] proposed a lightweight hybrid tracking algorithm called Kerman, aiming to implement intelligent surveillance as an edge service. Additionally, Liu et al. [12] conducted a survey on CF algorithms for object tracking, discussing the research background and recent advances in the field and introducing major object-tracking datasets.

To address the limitations of CF-based methods in handling complex target appearances, illumination variations, background clutter, and occlusions, researchers have proposed innovative approaches. Fu et al. [7] presented a novel Object-Aware Saliency-Guided Dually Regularized CF (DRCF) that utilizes saliency maps to improve the learning and adaptation of filters, particularly in occlusion situations. Ma et al. [13] introduced Average Peak-to-Correlation Energy (APCE), a new standard for multiresolution translation filter frameworks, which aims to achieve more robust and accurate scale estimation. Furthermore, Li et al. [14] developed a tracker called ADTrack, based on Discriminative Correlation Filters (DCF), which incorporates illumination adaptation and anti-dark capability. Despite these advancements, the limited ability of CF-based methods in practical scenarios remains a challenge. Therefore, it is necessary to develop more expressive feature representations and robust optimization strategies, leading to the exploration of alternative paradigms.

Tensor decomposition into orthogonal factors is a common practice in many fields, including statistics, machine learning, and signal processing. Various tensor decomposition (TD) techniques have been proposed and applied in different scientific and engineering domains. De et al. [15] introduced a novel TD method called Constrained Factor (CONFAC) decomposition. Kolda [16] provided an overview of software and applications for higher-order TD. Grigis and Richard [17] proposed a novel High-Order (HO) TD suited for redirection. Kiraly and Sabharwal [18] studied Orthogonal Outer Product Decompositions, where factors in the addition are orthogonal between additions. Favier et al. [19] introduced a novel tensor model called Nested Tucker Decomposition (NTD), applicable to Tensor Space-Time Coding (TSTC) in MIMO relay systems. Xu et al. [20] proposed an efficient TD method for decomposing Vandermonde factor matrices for Direction of Arrival (DOA) estimation in transmit beamforming MIMO radar. Azaiez et al. [21] used TD and one-dimensional interpolation in data-driven order reduction modeling to reduce the dimensionality of high-dimensional systems. Zhao et al. [22] introduced a statistical method named SSR-Tensor that detects temporal consistency hotspots effectively in spatiotemporal datasets through TD. Hutter et al. [23] evaluated the performance of low-rank TD methods in modeling application performance.

While TD is not common in SOT, Ratre et al. [24] applied it in their research. They described the application of TD in object categorization tasks, where they combined Tucker Tensor Decomposition (TTD) with Gaussian Mixture Models (GMM). Although their study focused on classification, the TTD method also holds potential for application in SOT.

TensorTrack is a tracker that combines Template Tracking and Detection with SOT. It exhibits competitive performance compared to deep neural network (DNN) methods. Furthermore, TensorTrack has been optimized to run efficiently on resource-constrained platforms, which contributes to its improved performance.

3. Preliminary

This paper aims to provide a brief introduction to CF and TTD. Using the framework of TensorTrack as an example, this section will give an overview of their fundamental principles and concepts.

3.1. Overview of Correlation Filters in Object Tracking

Correlation filters compute the similarity between a target template and input images for object localization. In their seminal work, Bolme et al. [10] proposed the Minimum Output Sum of Squared Error (MOSSE) filter, which initializes a filter using a single frame to ensure robust tracking. Subsequent studies, such as Henriques et al. [25], introduced techniques like the Fast Fourier Transform (FFT) and the Kernel Trick to enhance filter accuracyb. Danelljan et al. [26] further improved filter performance by projecting RGB images into an 11-color channel representation, while Bertinetto et al. [27] proposed a probability-based correlation filter, advancing SOT capabilities.

Figure 1 illustrates the overall framework of TensorTrack, integrating TTD within a correlation filter pipeline for robust object tracking. The process starts with the input video frame sequence, where the target is manually annotated in the first frame using a bounding box. The region of interest (ROI) containing the target is cropped from subsequent frames to focus on relevant areas, reducing extraneous data. Next, features are extracted from the cropped frames, effectively representing the target's appearance and structural information. These extracted features are then processed using Tucker Tensor Decomposition to reduce dimensionality while preserving critical spatial and temporal information. This step significantly enhances the efficiency and robustness of the framework.

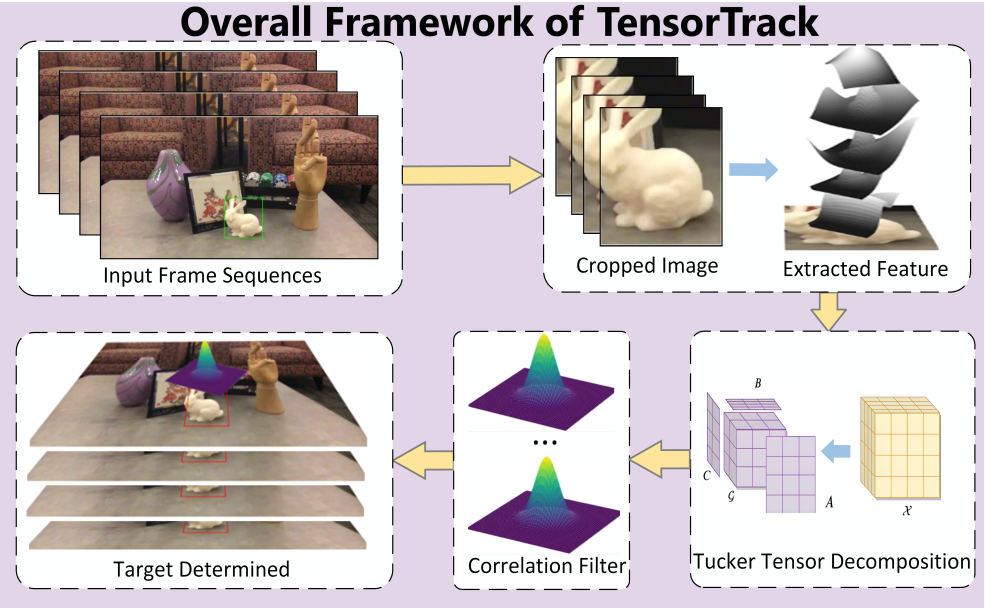


Figure 1. Overall framework of TensorTrack.

In the learning phase, the correlation filter leverages the decomposed features to construct a statistical model, capturing key patterns and relationships for accurate target tracking. This process is described by Equation (1):

$$\hat{H}_i = \eta * \frac{G_i \odot F_i}{F_i \odot F_i} + (1 - \eta) * \hat{H}_{i-1} \tag{1}$$

Here, $\hat{H}_i$ represents the learned filter at iteration i, where the hat notation $\hat{}$ denotes an estimation, $\odot$ indicates element-wise multiplication, and capital letters represent the Fourier transform. The iterative nature of this formula allows the correlation filter to adapt to changes in the target's appearance over time, ensuring robust and reliable tracking performance across frames.

During the detection stage, the correlation filter estimates the target's position in each subsequent frame by computing:

$$\hat{Y}_i = \hat{H}_i \bigodot X_i \tag{2}$$

where $\hat{Y}_i$ denotes the response map, and X_i represents the extracted features in the current frame. The tracking result is obtained by identifying the location with the highest response value within a predefined search region.

To ensure continuous tracking accuracy, periodic updates of the correlation filter are required. These updates refine the learned model $\hat{H}_i$ based on new observations, preventing model drift and adapting to appearance variations.

The score map ($\hat{Y}_i$), which represents the Fourier transform of the response, indicates the target's position based on the maximum response value, as depicted in Figure 2.

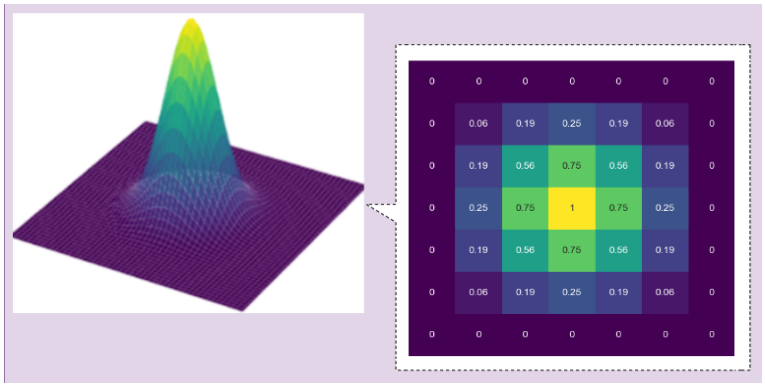


Figure 2. The object's position is determined by locating the maximum value on the score map. To obtain the score map, it is necessary to apply the inverse Fourier transform to the Fourier transformed score map $\hat{Y}$.

3.2. Principles of Tucker Tensor Decomposition

TTD is a mathematical approach used for analyzing multidimensional data. It breaks down a higher-dimensional tensor into lower-dimensional tensors, reducing the complexity of data representation. The Tucker decomposition identifies essential features and relationships within the data by factoring a tensor into a core tensor and factor matrices. This study employs an online Tucker2 decomposition, a specific type of tensor decomposition. In general, tensor decomposition can be classified into two types: CANDECOMP/PARAFAC (CP) tensor decomposition and Tucker tensor decomposition [28].

In general notation, a corresponds to a scalar, $\mathbf{a}$ represents a vector, A symbolizes a matrix, and $\mathcal{A}$ denotes a tensor. It should be noted that while a tensor is indexed using more than two index numbers, an n-mode tensor is indexed using n independent indices to represent its elements.

Tucker decomposition was originally proposed by Tucker in 1963 [29] and later refined in subsequent articles by Levin [30] and Tucker [28,31]. Among these works, the 1966 article by Tucker [28] is widely regarded as the most comprehensive and frequently cited reference.

Tensor-Train Decomposition reduces the rank of a tensor, explores hierarchical structures within a specific task, and minimizes redundancy in matrix representation [32,33]. The general formulation of TTD for a three-mode tensor is provided in Equation (3), where the tensor $\mathcal{X}$ is approximated by decomposing it into smaller core tensors and factor matrices using the i-mode product:

$$\mathcal{X} \approx \mathcal{G} \times_1 A \times_2 B \times_3 C \tag{3}$$

where the tensors $\mathcal{X} \in \mathbb{R}^{I_1 I_2 I_3}$ and $\mathcal{G} \in \mathbb{R}^{I_4 I_5 I_6}$ are considered, along with matrices $A \in \mathbb{R}^{I_1 I_4}$, $B \in \mathbb{R}^{I_2 I_5}$, and $C \in \mathbb{R}^{I_3 I_6}$. The goal is to decompose the tensor $\mathcal{X}$ into $\mathcal{G}$ by utilizing the i-mode product. Specifically, for this decomposition, it is assumed that $I_4 < I_1$, $I_5 < I_2$, and $I_6 < I_3$, as shown in Figure 3.

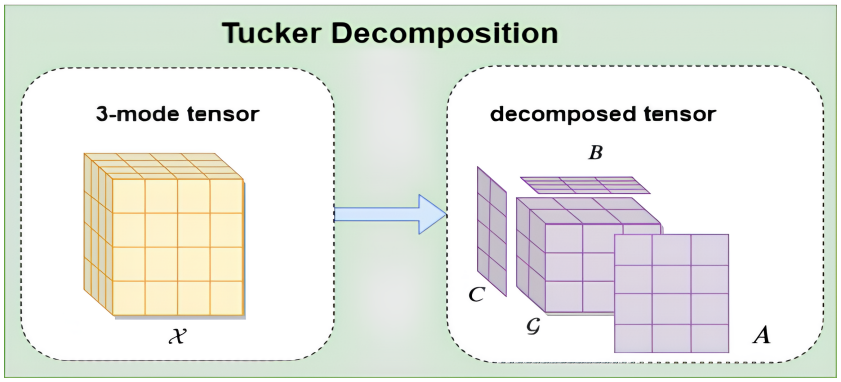


Figure 3. TTD reduces tensor rank, facilitates hierarchical structure analysis, and minimizes redundancy in matrix representations.

$$(\mathcal{G} \times_2 B)_{i_3 j_2 i_5} = \sum_{i_4=1}^{N_4} g_{i_3 i_4 i_5} \cdot b_{j_2 i_4} \tag{4}$$

Equation (4) represents the mode-2 tensor product, where $\mathcal{G}$ denotes the core tensor and B is the factor matrix associated with the second mode. Here, N_4 corresponds to the dimensionality of the decomposed mode. The operation $\times_2$ refers to the mode-2 product, which projects the high-dimensional tensor $\mathcal{G}$ into a lower-dimensional subspace. This operation is a fundamental step in Tucker decomposition, facilitating the extraction of key features and reducing data redundancy.

The Tucker2 decomposition itself can be framed as an optimization problem aimed at extracting the most informative features from the data, expressed as follows:

$$\max_{\substack{U \in \mathbb{R}^{H_1 \times d}; U^T U = I_d \\ \mathbf{v} \in \mathbb{R}^{1 \times d}; \mathbf{v}^T \mathbf{v} = 1}} \sum_{i=1}^{W_1 \cdot 3} \left\| U^T \mathbf{x_i} \mathbf{v} \right\|_1, \tag{5}$$

where U represents the projection matrix that maps the input features to a lower-dimensional subspace, $\mathbf{v}$ is a unit vector, and $\mathbf{x_i}$ denotes the input feature vectors. The objective of this optimization is to maximize the ℓ_1 norm of the projected features while preserving the most discriminative information in the data. This ensures dimensionality reduction while maintaining essential structure in the features.

The ℓ_1 norm promotes sparsity in feature selection by summing the absolute values of a vector's elements:

$$\sum_{i=1}^{N} \|a_i\|_1 = \sum_{i=1}^{N} \text{sign}(a_i)a_i = \text{sign}(\mathbf{a})^T\mathbf{a} = \max_{\mathbf{b}\in\{\pm1\}^N} \mathbf{b}^T\mathbf{a}, \tag{6}$$

where $\text{sign}(\cdot)$ denotes the sign function, which returns $\{\pm1\}$ depending on the sign of the input. This equivalence facilitates the computation of the ℓ_1 norm by reformulating it as an optimization problem involving binary variables.

It can be further shown that:

$$\sum_{i=1}^{W_1\cdot 3} \left\| U^T\mathbf{x}_i\mathbf{v} \right\|_1 = \max_{\mathbf{b}\in\{\pm1\}^{W_1\cdot 3}} U^T \left(\sum_{i=1}^{W_1\cdot 3} b_i\mathbf{x}_i \right)\mathbf{v}, \tag{7}$$

where $\mathbf{b} = [b_1, b_2, \ldots, b_{W_1\cdot 3}]$, and $b_i = \text{sign}(U^T x_i v)$. This equation demonstrates how the ℓ_1 norm of the projected features can be maximized by optimizing the binary weights $\mathbf{b}$. The formulation provides an efficient means of enhancing the feature representation by weighting the input vectors.

Finally, the optimization problem can be reformulated as:

$$\max_{\substack{U\in\mathbb{R}^{H_1\times d}; U^TU=I_d \\ \mathbf{v}\in\mathbb{R}^{1\times d}; \mathbf{v}^T\mathbf{v}=1}} \left\| U^T\mathbf{x}_i\mathbf{v} \right\|_1 = \sigma_{\max}\left(\sum_{i=1}^{W_1\cdot 3} b_i\mathbf{x}_i \right), \tag{8}$$

where $\sigma_{\max}(\cdot)$ returns the largest singular value of the input matrix. This reformulation indicates that the Tucker decomposition process can be efficiently solved as a singular value problem, enabling effective dimensionality reduction and feature extraction.

To construct the input matrix for the optimization, the feature vectors $\mathbf{x}_i$ are concatenated to form $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{W_1\cdot 3}] \in \mathbb{R}^{H_1\times W_1\cdot 3}$. The weighted sum $\sum_{i=1}^{W_1\cdot 3} b_i\mathbf{x}_i$ can then be expressed in matrix form as $X(\mathbf{b}\otimes I_M)$, where $\otimes$ denotes the Kronecker product. Thus, the final optimization can be written as:

$$\begin{aligned} &\max_{\substack{U\in\mathbb{R}^{H_1\times d}; U^TU=I_d \\ \mathbf{v}\in\mathbb{R}^{1\times d}; \mathbf{v}^T\mathbf{v}=1}} \sum_{i=1}^{W_1\cdot 3} \left\| U^T\mathbf{x_i}\mathbf{v} \right\|_1 \\ &= \max_{\mathbf{b}\in\{\pm1\}^{W_1\cdot 3}} \sigma_{\max}(X(\mathbf{b}\otimes I_M)). \end{aligned} \tag{9}$$

This final optimization problem highlights the efficient use of singular value decomposition in Tucker decomposition, providing a computationally feasible solution for handling high-dimensional data in resource-constrained settings.

4. Proposed Method

4.1. Video Decomposition Process

The video decomposition process is a crucial preprocessing step in TensorTrack, enabling efficient and robust tracking via Tucker Tensor Decomposition (TTD). As shown in Figure 4, video frames are converted to grayscale to reduce complexity while preserving structural information. The grayscale frames are cropped to isolate the ROI, focusing on the target and minimizing background noise. Tucker2 decomposition is then applied to extract key features and compress the data into a lower-dimensional representation. The resulting tensors are flattened for subsequent analysis, enhancing computational efficiency and forming the basis for accurate tracking in TensorTrack.

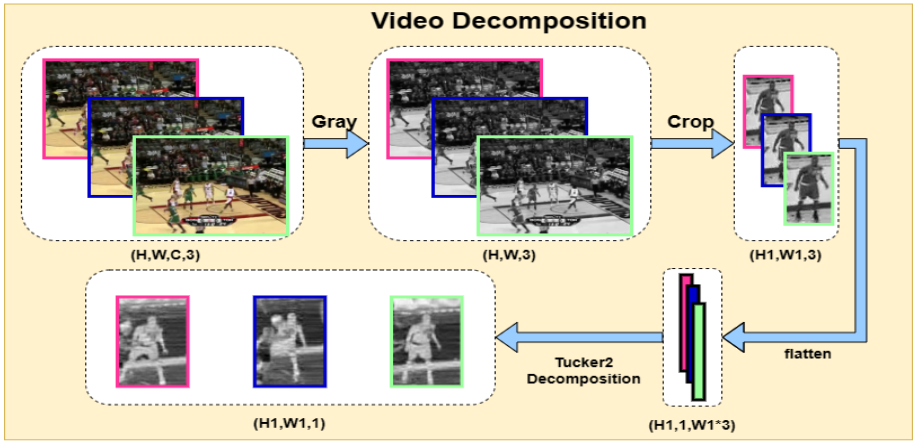


Figure 4. Video decomposition process.

4.2. Tucker2 Decomposition and Motion Pattern

The Tucker2 decomposition extends the conventional Tucker framework by incorporating the concatenation of (n-1)-mode tensors along each axis of the original n-mode tensor, a configuration that mirrors the complex data arrangements prevalent in computer vision (CV) tasks. This advanced decomposition is succinctly expressed as:

$$\mathcal{X} = \mathcal{G} \times_1 A \times_2 B \times_3 C, \tag{10}$$

where $\mathcal{X}$ denotes the multidimensional video data tensor subject to decomposition, $\mathcal{G}$ represents the core tensor, and A, B, and C are the factor matrices corresponding to the respective tensor modes. This methodology facilitates an efficient restructuring of voluminous video data, enabling a more analytical and structured approach reflective of segmentation practices within CV disciplines.

The adaptation of this framework significantly mitigates the challenges posed by data outliers, a prevalent issue identified within both Tucker and Tucker2 architectures, as documented by the literature. It also suggests a shift from the conventional L2-norm to a more robust L1-norm, a recommendation predicated on empirical evidence by Markopoulos et al. [34], which showcases an enhanced tracking accuracy within SOT frameworks employing correlation filters.

Departing from the conventional SOT methodologies, predominantly centered around feature extraction and decision fusion, this study pioneers a motion-oriented video tensor decomposition strategy. Drawing inspiration from the seminal work by Markopoulos et al. [34], the proposed method assumes a natural segregation between the target's motion and the static backdrop, implying that the background typically manifests a low-rank tensor structure, while dynamic entities, including the target, contribute significantly to the higher-rank components. Based on this theoretical underpinning, the study advocates for the bifurcation of video clips into distinct segments of high and low rank, predominantly encapsulating the moving target within the high-rank portion. In light of the minimal spatial variance across color channels, the initial conversion of image data to grayscale is recommended, thus reducing computational load while preserving essential structural detail. This scholarly exposition articulates a nuanced equilibrium between computational efficiency and the accurate delineation of dynamic versus static elements within video streams, advancing the field of object tracking through a meticulously crafted Tucker2 decomposition strategy. Considering the tensor decomposition is an np-hard problem. The exact solution to the L1 Tucker2 is applied:

$$\max_{\substack{U\in\mathbb{R}^{H_1 xd};U^TU=I_d \\ \mathbf{v}\in\mathbb{R}^{1xd};\mathbf{v}^T\mathbf{v}=1}} \sum_{1}^{W_1*3} \|U^T\mathbf{x_i}\mathbf{v}\|_\mathbf{1} \tag{11}$$

This paper proposes selecting $d = 1$ to project the tensor $\mathcal{X} \in R^{(H_1,1,W_1*3)}$ into a lower-rank space using the optimal matrices U and $\mathbf{v}$. In this approach, the width (W) and color (C) dimensions are combined into channels to maintain spatial consistency and ensure shared information across all channels. The vector $\mathbf{x_i} \in \mathbf{R^{H_1\times 1}}$. Decomposing the video tensor in this way preserves critical motion information, facilitating robust tracking.

To solve for the exact solution to the formula in Equation (11), the optimization problem is reformulated as follows:

$$\max_{\mathbf{b}\in\{\pm\mathbf{1}\}^{\mathbf{W_1*3}}} \sigma_{max}(X\mathbf{b}\otimes\mathbf{I_M}), \tag{12}$$

where $X = [\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_{W_1*3}}] \in \mathbf{R^{H_1\times W_1*3}}$, $\otimes$ denotes the Kronecker product, and $\mathbf{b}$ is a binary vector of length $W_1 * 3$ with entries in $\{\pm 1\}$.

The optimal solution to Equation (12), denoted as b_{opt}, is then used to reconstruct the background using the following equation:

$$B_i = U_{opt}U_{opt}^T X_i v_{opt} v_{opt}^T, \tag{13}$$

where U_{opt} and v_{opt}^T are obtained via singular value decomposition (SVD) of $(Xb_{opt} \otimes I_M)$.

4.3. Randomized Singular Value Decomposition for Efficiency

To improve computational efficiency, a randomized singular value decomposition (SVD) approach is employed. This method approximates the SVD of the original matrix using a random projection matrix, significantly reducing computational complexity while maintaining high accuracy. By leveraging randomized sketching, the proposed approach efficiently scales to large matrices, making it well-suited for real-time video processing. The corresponding Python-style pseudocode is presented in Algorithm 1.

Algorithm 1 Randomized singular value decomposition

Input: $A \in R^{m\times n}$ (target matrix), rank parameter k, power iteration parameter p
Output: $U \in R^{m\times k}$, $S \in R^{k\times k}$, $V \in R^{n\times k}$
Step 1: Randomized Initialization
$m, n = A.shape$
$\Omega = \text{np.random.randn}(n, k)$
$Y = A \times \Omega$
Step 2: Power Iterations (Optional)
for i **in range(**p**):**
$Y = A \times (A^T \times Y)$
Step 3: QR Decomposition
$Q, R = \text{np.linalg.qr}(Y)$
Step 4: Projection onto Lower-Dimensional Subspace
$B = Q^T \times A$
Step 5: Compute SVD on the Smaller Matrix
$\hat{U}, S, V = \text{np.linalg.svd}(B, \text{full_matrices=False})$
Step 6: Map $\hat{U}$ **Back to the Original Space**
$U = Q \times \hat{U}$
Step 7: Return the Decomposition
return U, S, V

The computational complexity of the traditional full SVD is $O(mn^2)$ or $O(m^2n)$, making it infeasible for large-scale video sequences due to excessive computational demands. In contrast, the proposed randomized SVD reduces the complexity to $O(mnk + k^3)$, where k is the rank parameter. This reduction is achieved by first projecting the original matrix onto a lower-dimensional subspace via a randomized sketching matrix, followed by performing SVD on the reduced matrix. The significantly lower computational overhead ensures that the method remains scalable and practical for real-time tracking applications.

Despite recent advancements in single-object tracking (SOT), conventional pipelines often fail to exploit motion dynamics effectively. For instance, Bertinetto et al. [27] proposed a statistical appearance model that relies primarily on visual features while overlooking temporal information. Similarly, Zhu et al. [35] introduced object detection datasets to extract high-level features, yet their approach exhibited limited capacity for temporal modeling. Danelljan et al. [36] integrated the IoU-Net to refine target state estimation, but their framework predominantly focused on appearance-based cues. In contrast, this study introduces a tensor-decomposition-based framework that leverages Tucker2 decomposition to extract and isolate dynamic motion components from the video tensor. This approach effectively addresses critical challenges such as tracking in visually ambiguous environments where the target shares similar colors with the background and cases of rapid motion.

4.4. Integrating Appearance and Motion

When appearance models and motion patterns are fused, they can complement each other and provide richer information for inferring the position and motion of the target. Appearance models are typically based on the texture and appearance features of the target, which help determine its appearance and shape. Motion patterns, on the other hand, focus on the movement characteristics of the target, such as speed, direction, and acceleration. By combining these two models, it is possible to consider both appearance and motion information, resulting in more accurate heatmaps. A more compact heatmap means that the peaks of the heatmap are more concentrated and focused, without obvious deviation or diffusion. This compactness enables tracking algorithms to more accurately locate the target and track its movement. Compared to scattered heatmaps, compact heatmaps provide clearer and more reliable target position information, which helps improve the accuracy and robustness of target-tracking algorithms. The principle behind the motion module is pretty simple yet effective. Suppose the estimation of the appearance model to be $\hat{A} = A + n_A$, and the estimation of the motion pattern to be $\hat{M} = M + n_M$, where n_A and n_M denote the estimation errors. Assume $n_A \sim N(0, \sigma_A)$ and $n_M \sim N(0, \sigma_M)$. Then, it follows that:

$$P(Motion\ Module) = P(\hat{A}) * p(\hat{M}) \implies Motion\ Module \sim N(0, \frac{\sigma_A^2 * \sigma_M^2}{\sigma_A^2 + \sigma_M^2}) \quad (14)$$

The above formula indicates that the composite estimation outperforms the original estimation by having a smaller variance. Furthermore, this algorithm demonstrates excellent performance in processing speed in real-time scenarios. Real-time capability is crucial in target-tracking tasks as it requires timely identification and tracking of the target to meet the demands of real-time applications.

To efficiently implement this fusion, an adaptive algorithm dynamically balances appearance and motion contributions based on their uncertainties, as outlined in Algorithm 2:

Algorithm 2 Integration of appearance and motion for heatmap fusion

Input: $A \in R^{m \times n}$, $M \in R^{m \times n}$
Output: $H_{fused} \in R^{m \times n}$ **Step 1: Compute Standard Deviation for Weighting**
$\sigma_A = \text{np.std}(A)$
$\sigma_M = \text{np.std}(M)$
Step 2: Normalize Heatmaps for Consistency
$A_{norm} = A/\text{np.max}(A)$
$M_{norm} = M/\text{np.max}(M)$
Step 3: Compute Weighting Factors
$w_A = \sigma_M^2/(\sigma_A^2 + \sigma_M^2)$
$w_M = \sigma_A^2/(\sigma_A^2 + \sigma_M^2)$
Step 4: Compute Fused Heatmap
$H_{fused} = w_A * A_{norm} + w_M * M_{norm}$
Step 5: Normalize the Fused Heatmap
$H_{fused} = H_{fused}/\text{np.max}(H_{fused})$
Step 6: Return Fused Heatmap
return H_{fused}

Figure 5 illustrates the effect of integrating appearance models and motion patterns within a compact motion module. This fusion generates a more focused and compact heatmap, enhancing its ability to accurately capture the target's position and motion. As a result, the proposed approach improves tracking robustness and precision, particularly in dynamic and visually challenging environments.

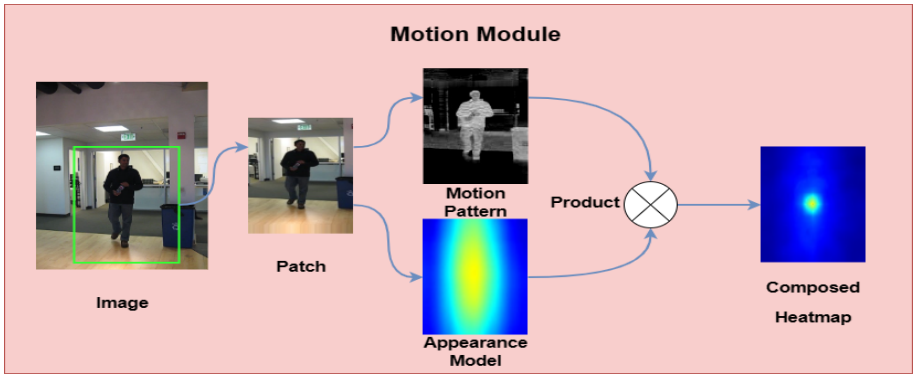


Figure 5. Fusion of appearance and motion models for compact heatmap generation.

5. Experiments

5.1. OTB100 Dataset

OTB100 is a widely used benchmark dataset for object tracking, consisting of 100 video sequences covering 22 object categories, including humans, animals, vehicles, faces, sports, and electronic devices. These sequences exhibit significant variations in appearance, scale, deformation, and occlusion, providing a diverse and challenging evaluation framework for tracking algorithms. OTB100 defines 11 tracking attributes to assess tracking performance under different challenges. The average video resolution is 356 × 530, with sequence lengths ranging from 71 to 3872 frames. Regarding background differentiation, OTB100 includes both simple and cluttered backgrounds. Some sequences feature low-contrast targets, increasing the difficulty for appearance-based trackers, while others present dynamic backgrounds and camera motion, requiring enhanced adaptability from tracking algorithms.

5.2. Implementation Details

All experiments were conducted on a workstation equipped with an Intel Core i9-10900K CPU and 32GB of RAM. To evaluate computational efficiency across different hardware settings, we conducted experiments on two different GPUs: an Nvidia RTX 1060 (6GB VRAM) and an Nvidia RTX 4070 (12GB VRAM). The proposed tracking framework and all benchmarked methods were implemented in Python using PyTorch (version 1.13) and NumPy for numerical operations. To ensure consistency, classical CF trackers and deep learning-based Siamese trackers were re-implemented based on their official source codes. To guarantee fair comparisons, all correlation filter-based trackers and our proposed method were executed on the RTX 1060 GPU. However, deep learning-based trackers (e.g., SiamR-CNN, OSTrack, TransT) exhibited significantly higher computational demands and were unable to run efficiently on RTX 1060 due to VRAM limitations. Therefore, these models were evaluated on the RTX 4070 GPU to ensure proper execution.

5.3. Impact of Tucker2 Decomposition

The proposed framework significantly enhances tracking performance on OTB100 (Table 1), achieving a 46.4% precision gain and a 16.2% AUC improvement over the CF baseline, demonstrating robustness in fast motion and backgrounds with high visual similarity to the target. Compared to the CF+Appearance Model, it further improves precision by 22.0% and AUC by 3.6%, highlighting the Tucker2 motion model's role in refining accuracy and mitigating drift. These results validate the effectiveness of integrating motion and appearance cues for more reliable target localization.

Table 1. Performance and efficiency comparison on the OTB100 dataset.

Method	Precision	AUC	FPS
CF (Baseline)	0.484	0.699	42
CF+Motion Model	0.507	0.729	39
CF+Appearance Model	0.581	0.784	32
CF+Appearance Model+Tucker2 (Ours)	**0.709**	**0.812**	**48**

To assess heatmap quality, we evaluated peak compactness and information entropy (Table 2). The CF+Appearance Model+Tucker2 framework achieves the highest compactness (1.08) and lowest entropy (1.62), indicating superior noise suppression and enhanced localization, outperforming all baselines.

Table 2. Analysis of heatmap quality for different methods.

Method	Peak Compactness	Information Entropy
CF (Baseline)	-	-
CF+Motion Model	1.45	2.01
CF+Appearance Model	1.31	1.86
CF+Appearance Model+Tucker2 (Ours)	1.08	1.62

As illustrated in Figure 6, the first row presents the original images, while the second row displays Tucker2 decomposition outputs, effectively isolating high-rank foreground components. The third row depicts heatmaps from the appearance model, which, despite capturing target-specific features, are prone to false responses in visually similar backgrounds and exhibit susceptibility to target drift in rapid motion or occlusion scenarios. Notably, in the fifth column, where the target and background share highly similar colors, the appearance model fails to distinguish the target, resulting in a diffuse and indistinct heatmap. The fourth row represents heatmaps generated by the proposed method, which integrates motion patterns extracted via

Tucker2 decomposition with appearance features. This fusion effectively mitigates challenges in scenarios with minimal relative motion, such as the Rubik clip (second-to-last column), where motion dynamics are difficult to extract due to limited target displacement. The fifth row shows the tracking results of the proposed method, which demonstrates improved target localization and robustness. By leveraging Tucker2 decomposition, the proposed approach enhances heatmap compactness and robustness, significantly improving target localization, suppressing noise, reducing false responses, and mitigating target drift, particularly in visually ambiguous or low-motion conditions.

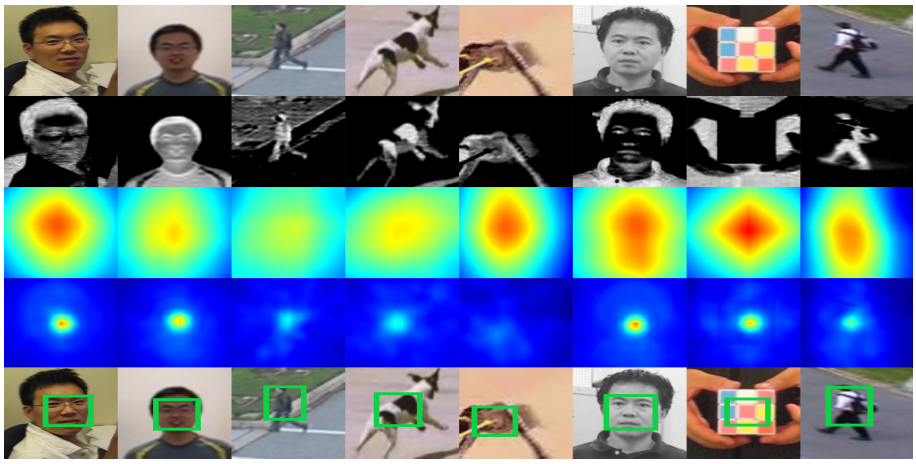


Figure 6. Comparison of original images, Tucker2 decomposition, and fused heatmaps.

5.4. Impact of Randomized SVD

As shown in Table 3, randomized SVD achieves an efficiency improvement of two orders of magnitude compared to Regular SVD while maintaining a very small reconstruction error. Specifically, the speed increases from 0.85 s to 0.007 s, with only a slight increase in reconstruction error. This drastic reduction in computational time demonstrates the feasibility of randomized SVD for real-time or large-scale applications, such as high-resolution video tracking or embedded systems. While randomized SVD introduces a slightly higher reconstruction error (3.78×10^{-13} vs. 5.45×10^{-16}), the impact on tracking performance is negligible, making it a suitable choice for scenarios with constrained computational resources.

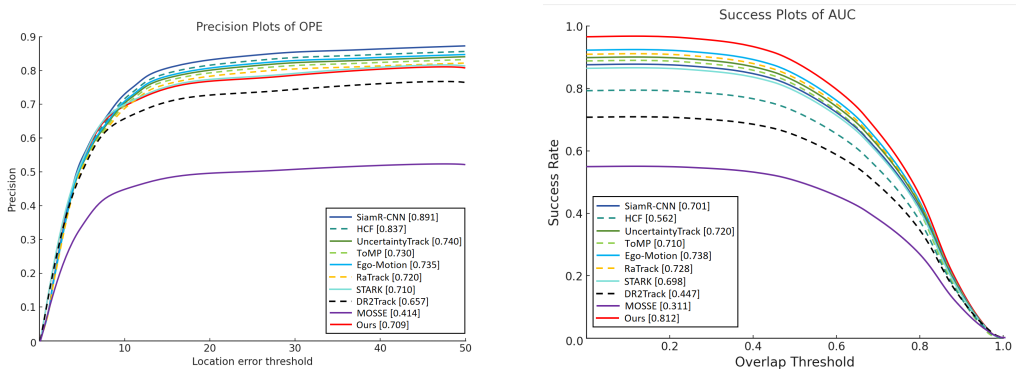
Table 3. Comparative experiment results of regular SVD and randomized SVD.

	Reconstruction Error (L2)	**Speed (Seconds)**
Regular SVD	5.45×10^{-16}	0.85
Randomized SVD	$\mathbf{3.78 \times 10^{-13}}$	**0.007**

5.5. Benchmark Evaluations

The proposed tracking framework was evaluated on the OTB100 dataset, comparing its performance and computational efficiency against classical correlation filter (CF) trackers and deep learning-based methods. As shown in Table 4, the proposed method achieves an AUC of 0.812 on the RTX 1060, demonstrating a significant 14.4% improvement over MixFormer (AUC = 0.720) in terms of tracking accuracy. While the precision of our method is slightly lower than that of some deep learning-based methods, such as SiamR-CNN (Precision = 0.891), this minor sacrifice is justified given the substantial AUC improvement, which directly contributes to tracking robustness and stability. Furthermore, compared to the CF (Baseline) method, our approach improves AUC by 16.2% and precision by

46.4%, demonstrating a clear advantage in overall tracking performance, as shown in the experimental comparison with various frameworks in Figure 7.



(**a**) Precision plots of OPE (**b**) Success plots of AUC

Figure 7. Performance comparison.

Regarding computational complexity, classical CF-based trackers typically exhibit linear to quadratic complexity, $O(N)$ to $O(N^2)$, allowing real-time inference but lacking robustness. In contrast, deep learning-based methods often operate with cubic complexity, $O(N^3)$, leading to substantial computational costs and a strong reliance on high-end hardware. For instance, SiamR-CNN runs at only 4.7 FPS on the RTX 4070, while even relatively efficient models like OSTrack and MixFormer only achieve 30–35 FPS, highlighting the dependency of deep learning methods on high-performance GPUs. In contrast, the proposed tracking framework maintains quadratic complexity, $O(N^2)$, enabling efficient inference on a low-end RTX 1060 GPU, reducing computational overhead by 66.7% compared to deep learning-based approaches. This significantly lowers hardware requirements, making the proposed method more accessible for deployment in real-world applications.

In terms of frame rate, the proposed method demonstrates remarkable real-time performance. On the RTX 1060, it achieves 48 FPS, representing a 60% increase over MixFormer (30 FPS) and a 37% increase over OSTrack (35 FPS), significantly surpassing deep learning-based trackers. Moreover, while classical CF trackers such as MOSSE achieve extremely high inference speeds (e.g., 669 FPS), their low accuracy limits their applicability in scenarios with high target–background similarity. The proposed framework balances accuracy and real-time performance, providing an efficient solution for object tracking, as shown in the efficiency comparison with various frameworks in Figure 8.

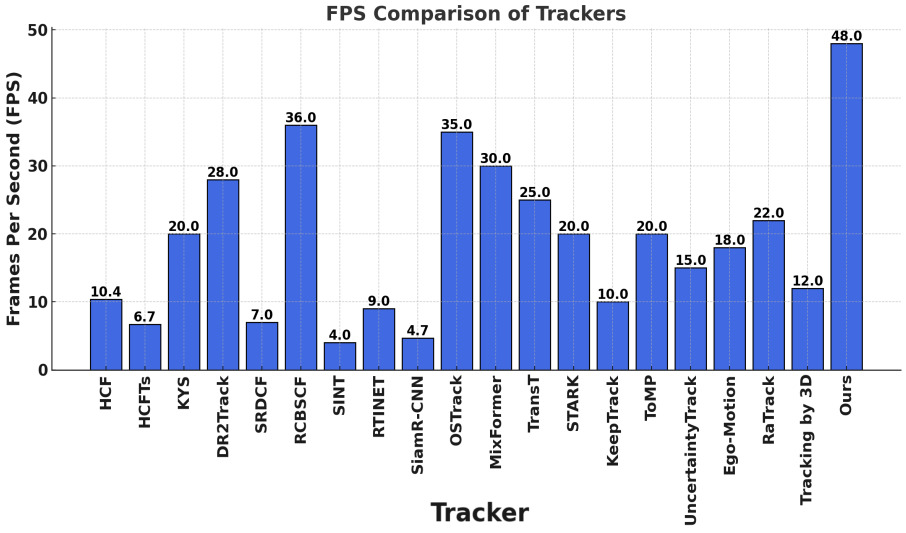


Figure 8. FPS Comparison of Trackers.

Table 4. Performance and complexity analysis on the OTB100 Dataset.

Tracker (Year)	Precision	AUC	Time Complexity	FPS	GPU Requirements
MOSSE (2010)	0.414	0.311	$O(N)$	**669**	RTX 1060
HCF (2015)	0.837	0.562	$O(N^2)$	10.4	RTX 1060
KYS (2020)	-	0.695	$O(N^2 \log N)$	20	RTX 1060
DR2Track (2021)	0.657	0.447	$O(N^3)$	28	RTX 1060
RCBSCF (2019)	0.711	0.485	$O(N^2)$	36	RTX 1060
Siamese Trackers	**Precision**	**AUC**	**Time Complexity**	**FPS**	**GPU Requirements**
SINT (2016)	0.788	0.592	$O(N^3)$	4	RTX 1060
RTINET (2019)	-	0.682	$O(N^3)$	9	RTX 1060
SiamR-CNN (2020)	**0.891**	0.701	$O(N^3)$	4.7	RTX 4070
Deep Learning-Based	**Precision**	**AUC**	**Time Complexity**	**FPS**	**GPU Requirements**
OSTrack (2022)	0.710	0.690	$O(N^3)$	35	RTX 4070
MixFormer (2022)	0.720	0.700	$O(N^3)$	30	RTX 4070
TransT (2021)	0.715	0.691	$O(N^3)$	25	RTX 4070
STARK (2021)	0.710	0.698	$O(N^3)$	20	RTX 4070
KeepTrack (2021)	0.725	0.685	$O(N^3)$	10	RTX 4070
ToMP (2022)	0.730	0.710	$O(N^3)$	20	RTX 4070
UncertaintyTrack (2024)	0.740	0.720	$O(N^3)$	15	RTX 4070
Ego-Motion (2024)	0.735	0.738	$O(N^3)$	18	RTX 4070
RaTrack (2023)	0.720	0.728	$O(N^3)$	22	RTX 4070
Tracking by 3D (2023)	0.725	0.730	$O(N^3)$	12	RTX 4070
Ours (2024)	0.709	**0.812**	$O(N^2)$	48	RTX 1060

6. Conclusions

This paper presents TensorTrack, an innovative tracking method that integrates tensor decomposition techniques with single-object tracking, offering a promising solution for real-time tracking in resource-constrained environments. Unlike deep learning-based methods that require significant computational resources for marginal accuracy improvements, TensorTrack achieves state-of-the-art performance while maintaining computational efficiency. The method excels in scenarios involving fast motion, limited movement, and similar target–background environments, outperforming traditional methods and demonstrating competitive advantages over deep learning-based approaches, especially with significantly higher speed. Future improvements may include incorporating additional information, such as optical flow and depth data, to further enhance robustness and accuracy for more complex tracking tasks.

Author Contributions: Conceptualization, Y.G. (Yuntao Gu); methodology, Y.G. (Yuntao Gu) and L.C.; software, Y.G. (Yuntao Gu) and H.W.; validation, Y.G. (Yuntao Gu), P.Z. and Y.G. (Yuanjun Guo); formal analysis, Y.G. (Yuntao Gu) and P.Z.; investigation, Y.G. (Yuntao Gu) and Y.G. (Yuanjun Guo); resources, P.Z., H.W. and Y.G. (Yuanjun Guo); data curation, Y.G. (Yuntao Gu), L.C. and Y.G. (Yuanjun Guo); writing—original draft preparation, Y.G. (Yuntao Gu) and Y.G. (Yuanjun Guo); writing—review and editing, P.Z., L.C. and H.W.; visualization, Y.G. (Yuntao Gu); supervision, Y.G. (Yuanjun Guo), W.D. and Y.L.; project administration, Y.G. (Yuanjun Guo), W.D. and Y.L.; funding acquisition, Y.G. (Yuanjun Guo), W.D. and Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by the Science and Technology Project of Shenzhen (grant numbers GJHZ20210705141401005, CJGJZD20220517141405012, and KCXST20221021111402006), Guangdong Special Support Plan Science and Technology Innovation Young Top-notchTalents (No. 2023T007L745), the "Nanling Team Project" of Shaoguan City, CGNProject (grant numbers KJZD20230923115215032), the Shenzhen-HongKong Joint Funding Project (A) (grant number SGDX20230116092053005) and National Science Foundation of China (62073232).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset used in this study has been described in detail within the text. It encompasses key parameters relevant to the research objectives, ensuring comprehensive coverage of the variables under investigation. The data provides a robust foundation for analysis, facilitating the exploration of complex relationships and the validation of the proposed methodologies.

Conflicts of Interest: Authors Wenjun Ding and Yu Liu were employed by the company China Construction Third Bureau Digital Engineering Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Pham, T.T.; Déniz, H.R.; Pham, T.D. Tensor decomposition of non-EEG physiological signals for visualization and recognition of human stress. In Proceedings of the 2019 11th International Conference on Bioinformatics and Biomedical Technology, Stockholm, Sweden, 29–31 May 2019; pp. 132–136.
2. Henretty, T.; Baskaran, M.; Ezick, J.; Bruns-Smith, D.; Simon, T.A. A quantitative and qualitative analysis of tensor decompositions on spatiotemporal data. In Proceedings of the 2017 IEEE High Performance Extreme Computing Conference (HPEC), Waltham, MA, USA, 12–14 September 2017; IEEE: Piscataway, NY, USA, 2017; pp. 1–7.
3. de Almeida, A.L.F.; Favier, G.; da Costa, J.; Mota, J.C.M. Overview of tensor decompositions with applications to communications. *Signals Images Adv. Results Speech Estim. Compress. Recognit. Filter. Process.* **2016**, *12*, 325–356.
4. Zhang, M.; Xing, J.; Gao, J.; Hu, W. Robust visual tracking using joint scale-spatial correlation filters. In Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP), Quebec City, QU, Canada, 27–30 September 2015; IEEE: Piscataway, NY, USA, 2015; pp. 1468–1472.
5. Tang, M.; Feng, J. Multi-kernel correlation filter for visual tracking. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 3038–3046.
6. Lukežič, A.; Zajc, L.Č.; Kristan, M. Deformable parts correlation filters for robust visual tracking. *IEEE Trans. Cybern.* **2017**, *48*, 1849–1861. [CrossRef] [PubMed]
7. Fu, C.; Xu, J.; Lin, F.; Guo, F.; Liu, T.; Zhang, Z. Object saliency-aware dual regularized correlation filter for real-time aerial tracking. *IEEE Trans. Geosci. Remote Sens.* **2020**, *58*, 8940–8951. [CrossRef]
8. Lin, F.; Fu, C.; He, Y.; Guo, F.; Tang, Q. Learning temporary block-based bidirectional incongruity-aware correlation filters for efficient UAV object tracking. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *31*, 2160–2174. [CrossRef]
9. Yueyang, G.; Kunqi, G.; Yu, Q.; Xiaoguang, N.; Kuan, X.; Xingqi, F.; Jie, Y. Boosting correlation filter based tracking using multi convolutional features. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; IEEE: Piscataway, NY, USA, 2019; pp. 3965–3969.
10. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual object tracking using adaptive correlation filters. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; IEEE: Piscataway, NY, USA, 2010; pp. 2544–2550.
11. Nikouei, S.Y.; Chen, Y.; Song, S.; Faughnan, T.R. Kerman: A hybrid lightweight tracking algorithm to enable smart surveillance as an edge service. In Proceedings of the 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA , 11–14 January 2019; IEEE: Piscataway, NY, USA, 2019; pp. 1–6.
12. Liu, S.; Liu, D.; Srivastava, G.; Połap, D.; Woźniak, M. Overview and methods of correlation filter algorithms in object tracking. *Complex Intell. Syst.* **2021**, *7*, 1895–1917. [CrossRef]
13. Ma, H.; Acton, S.T.; Lin, Z. SITUP: Scale invariant tracking using average peak-to-correlation energy. *IEEE Trans. Image Process.* **2020**, *29*, 3546–3557. [CrossRef]
14. Li, B.; Fu, C.; Ding, F.; Ye, J.; Lin, F. All-day object tracking for unmanned aerial vehicle. *IEEE Trans. Mob. Comput.* **2022**. [CrossRef]

15. de Almeida, A.L.; Favier, G.; Mota, J.C.M. A constrained factor decomposition with application to MIMO antenna systems. *IEEE Trans. Signal Process.* **2008**, *56*, 2429–2442. [CrossRef]
16. Kolda, T.G.; Bader, B.W. Tensor decompositions and applications. *SIAM Rev.* **2009**, *51*, 455–500. [CrossRef]
17. Grigis, A.; Renard, F.; Noblet, V.; Heinrich, C.; Heitz, F.; Armspach, J.P. A new high order tensor decomposition: Application to reorientation. In Proceedings of the 2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, Chicago, IL, USA, 30 March–2 April 2011; IEEE: Piscataway, NY, USA, 2011; pp. 258–261.
18. Király, F.J. Efficient Orthogonal Tensor Decomposition, with an Application to Latent Variable Model Learning. *arXiv* **2013**, arXiv:1309.3233.
19. Favier, G.; Fernandes, C.A.R.; de Almeida, A.L. Nested Tucker tensor decomposition with application to MIMO relay systems using tensor space–time coding (TSTC). *Signal Process.* **2016**, *128*, 318–331. [CrossRef]
20. Xu, F.; Morency, M.W.; Vorobyov, S.A. DOA estimation for transmit beamspace mimo radar via tensor decomposition with vandermonde factor matrix. *IEEE Trans. Signal Process.* **2022**, *70*, 2901–2917. [CrossRef]
21. Azaïez, M.; Chacón Rebollo, T.; Gómez Mármol, M.; Perracchione, E.; Rincón Casado, A.; Vega, J. Data-driven reduced order modeling based on tensor decompositions and its application to air-wall heat transfer in buildings. *SeMA J.* **2021**, *78*, 213–232. [CrossRef]
22. Zhao, Y.; Yan, H.; Holte, S.; Mei, Y. Rapid detection of hot-spots via tensor decomposition with applications to crime rate data. *J. Appl. Stat.* **2022**, *49*, 1636–1662. [CrossRef] [PubMed]
23. Hutter, E.; Solomonik, E. Multi-Parameter Performance Modeling via Tensor Completion. *arXiv* **2023**, arXiv:2210.10184.
24. Ratre, A.; Pankajakshan, V. Tucker tensor decomposition-based tracking and Gaussian mixture model for anomaly localisation and detection in surveillance videos. *IET Comput. Vis.* **2018**, *12*, 933–940. [CrossRef]
25. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. Exploiting the circulant structure of tracking-by-detection with kernels. In Proceedings of the Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, 7–13 October 2012; Proceedings, Part IV 12; Springer: Berlin/Heidelberg, Germany, 2012; pp. 702–715.
26. Danelljan, M.; Shahbaz Khan, F.; Felsberg, M.; Van de Weijer, J. Adaptive color attributes for real-time visual tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1090–1097.
27. Bertinetto, L.; Valmadre, J.; Golodetz, S.; Miksik, O.; Torr, P.H. Staple: Complementary learners for real-time tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1401–1409.
28. Tucker, L.R. Some mathematical notes on three-mode factor analysis. *Psychometrika* **1966**, *31*, 279–311. [CrossRef]
29. Tucker, L.R. Implications of factor analysis of three-way matrices for measurement of change. *Probl. Meas. Chang.* **1963**, *15*, 3.
30. Levin, J. Three-mode factor analysis. *Psychol. Bull.* **1965**, *64*, 442. [CrossRef]
31. Tucker, L.R. The extension of factor analysis to three-dimensional matrices. In *Contributions to Mathematical Psychology*; Holt, Rinehart and Winston: New York, NY, USA, 1964; pp. 110–127.
32. Yu, X.; Luo, Z. A sparse tensor optimization approach for background subtraction from compressive measurements. *Multimed. Tools Appl.* **2021**, *80*, 26657–26682. [CrossRef]
33. Lebedev, V.; Ganin, Y.; Rakhuba, M.; Oseledets, I.; Lempitsky, V. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. *arXiv* **2014**, arXiv:1412.6553.
34. Markopoulos, P.P.; Kundu, S.; Chamadia, S.; Pados, D.A. Efficient L1-norm principal-component analysis via bit flipping. *IEEE Trans. Signal Process.* **2017**, *65*, 4252–4264. [CrossRef]
35. Zhu, Z.; Wang, Q.; Li, B.; Wu, W.; Yan, J.; Hu, W. Distractor-aware siamese networks for visual object tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 101–117.
36. Danelljan, M.; Bhat, G.; Khan, F.S.; Felsberg, M. Atom: Accurate tracking by overlap maximization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4660–4669.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.