

# An Improved TLD Tracking Algorithm with RPCA-Based Occlusion Handling and a Simplified Nearest Neighbor Detection Model

Peng Zhang

School of Electrical and Information Engineering, Anhui University of Science and Technology, Huainan, China

**Abstract:** The Tracking-Learning-Detection (TLD) framework is a real-time long-term tracking system comprising three components: tracking, learning, and detection. However, tracking performance is inevitably degraded by severe illumination changes, occlusion, and rotation, which frequently lead to tracking failure. To address these challenges, this paper proposes an improved TLD algorithm that integrates Robust Principal Component Analysis (RPCA) into the detection stage to separate low-rank components from the image, thereby effectively suppressing background occlusion. In addition, the detection model is simplified by replacing the cascaded classifier with a Nearest Neighbor Classifier (NNC), which enhances both efficiency and effectiveness. By combining RPCA-based occlusion handling with a streamlined detection model, the proposed algorithm achieves more robust and efficient dynamic tracking in long-term scenarios..

**Keywords:** TLD; object tracking; occlusion; Robust Principal Component Analysis (RPCA); Nearest Neighbor Classifier (NNC); detection model.

## 1. Introduction

Zdenek Kalal et al. [1] propose a long-term tracking of unknown objects in a video stream. The proposed TLD [1] tracking algorithm is decomposed into three sub-tasks: tracking [2-4], learning [5-6] and detection [7-9] shown in Figure 1.

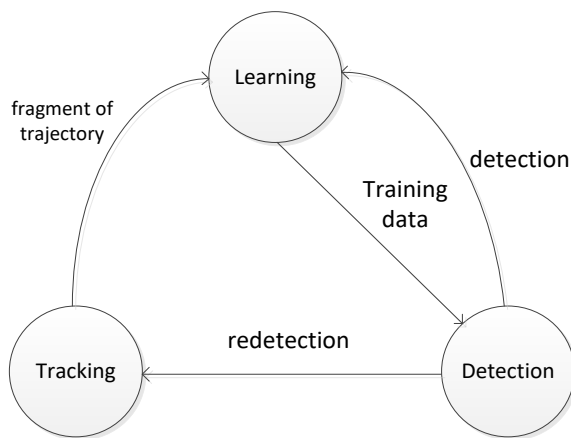


Figure 1: TLD framework

The tracker estimates the current target based on the last frame location under the assumption that the target is always visible. The detector scans every frame and find all the similar appearances to acquire positive and negative patches for learning. Meanwhile, estimating the tracker's errors and updating it if necessary. The learning decides the performance of tracker and detector based on the positive and negative patches, eliminating the detector's errors and update it.

However, background clutter, motion blur and partial occlusions are easy to make a tracker fail during long-term tracking, how to solve these core questions have become a determining factor for a successful tracker.

Mmanuel Candes et al. [10] prove that both the low-rank and the sparse components can be recovered by solving a very convenient convex [11-12] program under some suitable assumptions, every captured video sequence have low-rank component and sparse component.

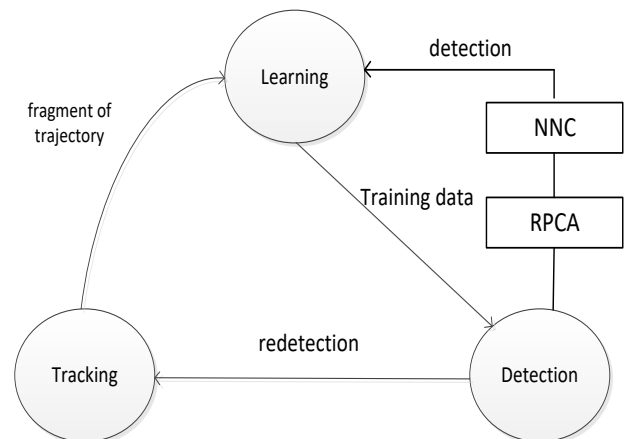


Figure 2: Enhanced TLD

If we take the video frames as a matrix  $D$ , the low-rank component  $A$  represents the stationary background and the sparse component  $E$  represents the moving target in the foreground. Robust PCA [10] aim to decompose the matrix  $D=A+E$  into the low-rank  $A$  and the sparse component  $E$ .

Meanwhile, the detector in TLD algorithm is a Cascaded classifier including patch variance, ensemble classifier [13] and nearest neighbor [14]. Although the patch variance and ensemble classifier that the first two stages in cascaded classifier can reduce a large number of bounding boxes for nearest neighbor, but the structure of detector is complicated that decreases the efficiency of classifier. So, our algorithm only uses an NNC (Nearest Neighbor Classifier) as a detector and decrease the dimensional space with RPCA.

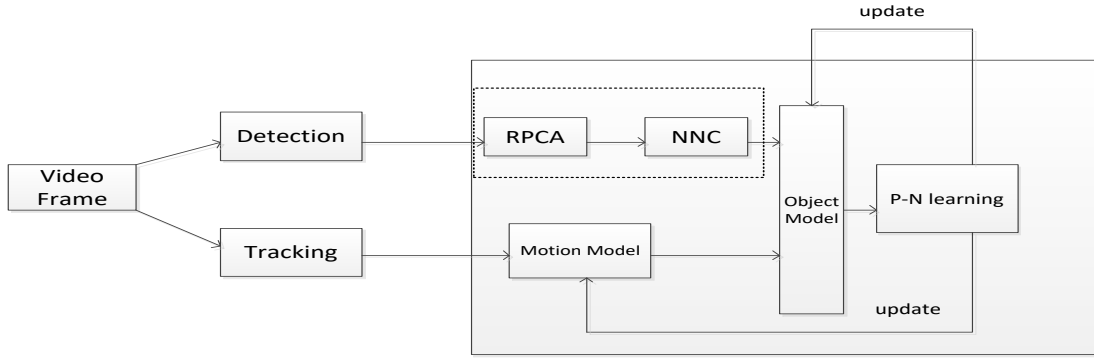


Figure 3: TLD-RPCA framework

In this paper, we proposed an enhanced TLD algorithm with Robust PCA to complete the tracking mission shown in Figure 2, which called TLD-RPCA in the rest of our paper. In our algorithm, the detector in TLD is divided into two stage: one stage for RPCA to recover a low-rank matrix and the second stage for Nearest Neighbor Classifier to detect.

## 2. Proposed Algorithm

In this section, we present the proposed TLD-RPCA algorithm in details.

### 2.1 TLD-RPCA Tracking Algorithm

Our algorithm are also decomposed into three parts shown in Figure 3: Object tracking produce smooth trajectories to construct a motion model, while the object detection is different from the TLD, which the detector use a scanning-window to scan the image after the RPCA and for each patch decide the presence or absence of the object using Nearest Neighbor Classifier. P-N learning is the decision model that comparing the two results from tracking and detection to decide the final target, then update the motion model and object model.

### 2.2 Implementation of RPCA

Robust PCA aim to recover a low-rank matrix  $A$  from a matrix  $D = A + E$ , for decomposition  $D = A + E$ , we can define by

$$\min_{A,E} \text{rank}(A) + \lambda E_0 \text{ s.t. } A + E = D \quad (1)$$

However, this formula is non-convex. But the question can be solved by relaxing  $L_0$ -norm to  $L_1$ -norm and replace rank with nuclear norm which can be defined by

$$\min_{A,E} \|A\|_* + \lambda E_1 \text{ s.t. } A + E = D \quad (2)$$

Where  $\|A\|_* = \sum_i \sigma_i(A)$ .

Finally, the formula is convex and exist a uniquely minimizer. And then we can take an optimization algorithm for RPCA, which use the Inexact ALM Method to solve the primal problem directly.

### 2.3 Nearest Neighborhood Classifier with RPCA

In detection model, the input image first be handled by RPCA to recover both the low-rank and the sparse components.

For the Observation matrix  $D \in R^{m \times n}$ , we do a fast iteration to SVD, In iteration  $k$ :

$$(U, S, V) = \text{svd}(D - E_{k+1} + \mu_k^{-1} Y_k) \quad (3)$$

where  $Y_0^* = \frac{\text{sgn}(D)}{J(\text{sgn}(D))}$ ;  $\mu_0 > 0$ ;  $\rho > 1$ ;  $k = 1$ .

Meanwhile, we get the low-rank components  $A_{k+1}$  and the sparse components  $E_{k+1}$ .

$$A_{k+1} = US_{\mu_k^{-1}}[S]V^T \quad (4)$$

$$E_{k+1} = S_{\lambda \mu_k^{-1}}[D - A_{k+1} + \mu_k^{-1} Y_k] \quad (5)$$

$$Y_{k+1} = Y_k + \mu_k (D - A_{k+1} - E_{k+1}) \quad (6)$$

Furthermore, the detector uses a scanning-window to scan the image handled by the RPCA and for each patch decides about presence or absence of the object using Nearest Neighbor Classifier.

A single instance of the object's appearance is represented by an image patch  $p$ . Similarity between two patches  $p_i, p_j$  is defined as:

$$S(p_i, p_j) = 0.5(\text{NCC}(p_i, p_j) + 1) \quad (7)$$

where NCC is a Normalized Correlation Coefficient.

we construct a dynamic data structure  $M$ , which is an object model that collects positive and negative patches.  $M = \{p_1^+, p_2^+, \dots, p_m^+, p_1^-, p_2^-, \dots, p_n^-\}$  where  $p_1^+$  represents the first positive patch added to the collection,  $p_n^-$  is the negative patch added last. Then, we calculate the relative similarity of each patch using the method to provide both positive and negative samples for learning model.

Similarity with the positive nearest neighbor

$$S^+(p, M) = \max_{p_i^+ \in M} S(p, p_i^+) \quad (8)$$

Similarity with the negative nearest neighbor,

$$S^-(p, M) = \max_{p_i^- \in M} S(p, p_i^-) \quad (9)$$

Relative similarity of each patch

$$S^r = \frac{S^+}{S^+ + S^-} \quad (10)$$

Relative similarity ranges from 0 to 1, higher values mean more confident that the patch depicts the object. we defined if the values are more than 0.6, it is the positive patch.

### 3. Experiments

We analyze our tracking algorithm with 6 different videos from OTB 100. For fair evaluations, we repeat the experiments 10 times on each sequence, and present the averaged results. Table 1 shows the dataset of our algorithm, while Table 2 shows the performance of our algorithm and the snapshot of our tracking algorithm is shown in Figure 4. Our tracker runs in MATLAB and the computer is a 2.40GHz CPU with 4 GB RAM.

In order to analyze the strength and weakness of the tracking algorithm, we also add the results of another two trackers including the KCF [15] and TLD. The tracking results compared to TLD and KCF are shown in Figure 5 and 6.

For the CarScale sequence shown in Figure 5, the appearance changes unexpectedly due to occlusion when the car goes through the tree in front of the camera. Our algorithms perform well on this sequence as the RPCA help to separate the low-rank composition from the image during the detection, avoiding the loss of target due to the occlusion of background. For the Tiger sequence shown in Figure 6, TLD even lose the target totally when the tracking object stay behind the tree leaf. TLD cannot acquire useful patch due to the occlusion. While after RPCA, the complex and immobile composition can be separated into low-rank, so the detection in our algorithm can acquire the positive and negative patch easier than others. More comparison details between these different tracking methods is shown in Table 3. For fair evaluations, we repeat the experiments 10 times on each sequence for one tracking algorithm, and present the averaged results.

**Table 1:** The dataset of our algorithm

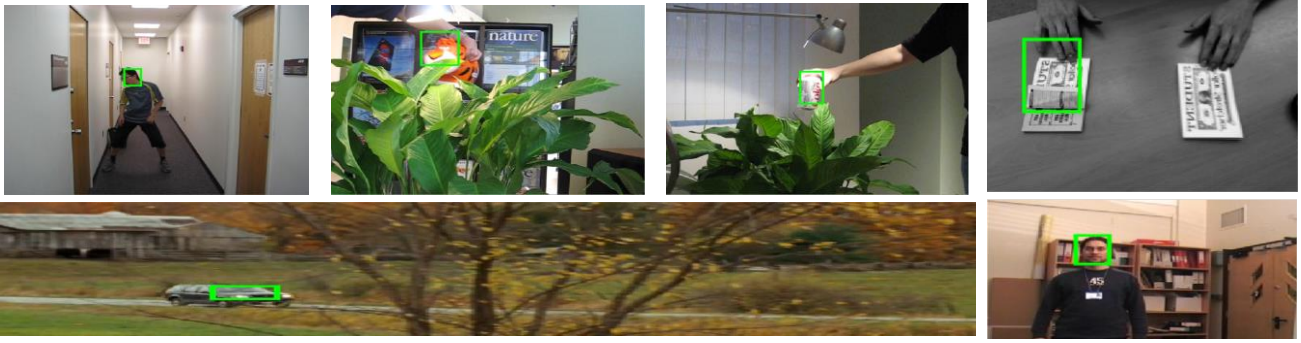
| <i>Name</i> | <i>Frames</i> | <i>Mov Camera</i> | <i>Partial occ</i> | <i>Full occ</i> | <i>Pose change</i> | <i>Illum change</i> | <i>Scale change</i> | <i>Similar objects</i> |
|-------------|---------------|-------------------|--------------------|-----------------|--------------------|---------------------|---------------------|------------------------|
| Boy         | 602           | yes               | no                 | no              | yes                | no                  | no                  | no                     |
| CarScale    | 252           | yes               | yes                | no              | yes                | no                  | yes                 | no                     |
| Coke        | 291           | no                | yes                | yes             | no                 | no                  | no                  | no                     |
| Coupon      | 327           | no                | no                 | no              | yes                | no                  | no                  | Yes                    |
| Man         | 134           | yes               | no                 | no              | yes                | yes                 | yes                 | no                     |
| Tiger       | 354           | no                | yes                | yes             | no                 | no                  | yes                 | no                     |

**Table 2:** The results of our algorithm

| <i>Number</i> | <i>Name</i> | <i>Frames</i> | <i>Not visible frames</i> | <i>visible frames</i> |
|---------------|-------------|---------------|---------------------------|-----------------------|
| 1             | Boy         | 602           | 0                         | 602                   |
| 2             | CarScale    | 252           | 0                         | 252                   |
| 3             | Coke        | 291           | 33                        | 258                   |
| 4             | Coupon      | 327           | 0                         | 327                   |
| 5             | Man         | 134           | 0                         | 134                   |
| 6             | Tiger       | 354           | 4                         | 350                   |

**Table 3:** More comparison details

| <i>Name</i> | <i>Frames</i> | <i>TLD</i> | <i>FCK</i> | <i>TLD-RPCA</i> |
|-------------|---------------|------------|------------|-----------------|
| Boy         | 602           | 602        | 503        | 602             |
| CarScale    | 252           | 153        | 161        | 252             |
| Coke        | 291           | 262        | 291        | 258             |
| Coupon      | 327           | 53         | 327        | 327             |
| Man         | 134           | 134        | 134        | 134             |
| Tiger       | 354           | 203        | 115        | 350             |



**Figure 4:** The snapshot of tracking result



**Figure 5:** Tracking results of the CarScale. Left: TLD-RPCA. Middle: TLD. Right: KCF.



**Figure 6:** Tracking results of the Tiger. Left: TLD-RPCA. Middle: TLD. Right: KCF.

#### 4. Conclusion

TLD is a real-time long-term tracking system including three models. These models operate simultaneously and support each other. Our algorithm applies the RPCA to TLD to eliminate the influence of occlusion and simplify the detection model to make the dynamic tracking algorithm more effective. Numerous experiments on different challenging video sequences demonstrated that our tracker achieves favorable performance in part or full occlusion and pose change.

#### References

- [1] Zdenek Kalal, Krystian Mikolajczyk and Jiri Matas, Tracking-Learning-Detection [J], IEEE Transactions on Pattern Analysis and Machine in Intelligence, 2010.
- [2] Songxiao Cao, Xuanyin Wang and Ke Xiang, Visual object tracking based on Motion-Adaptive Particle Filter under complex dynamics [J], EURASIP Journal on Image and Video Processing, 2017, 2017(1).
- [3] Mengjie Hu, Zhen Liu, Jingyu Zhang, et al., Robust object tracking via multi-cue fusion [J], Signal Processing, 2017, 139: 86-95.
- [4] Dandan Huang and Yi Sun, Object tracking using discriminative sparse appearance model [J], Signal Processing: Image Communication, 2015, 37: 1-18.
- [5] Zdenek Kalal, Jiri Matas and Krystian Mikolajczyk, P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints [J], IEEE, 2010.
- [6] Zdenek Kalal, Krystian Mikolajczyk and Jiri Matas, Forward-Backward Error: Automatic Detection of Tracking Failures [J], International Conference on Pattern Recognition, 2010.
- [7] Wenhua Fang, Jun Chen and Ruimin Hu, Structural sparse representation for object detection [J], Wuhan University Journal of Natural Sciences, 2017, 22(4): 318-322.
- [8] Muhammad Attamimi and Takayuki Nagai, Djoko Purwanto, Object detection based on particle filter and integration of multiple features [J], Procedia Computer Science, 2018, 144: 214-218.
- [9] Huicheng Zheng, Jiajie Chen, Lvran Chen, et al., Feature Enhancement for Multi-scale Object Detection [J], Neural Processing Letters, 2020, 51(1):1907-1919.
- [10] E. J. Candes, X. Li, Y. Ma, and J. Wright, Robust Principal Component Analysis, 2009.
- [11] Z. Lin, M. Chen, L. Wu, and Y. Ma, The augmented Lagrange multiplier method for exact recovery of a corrupted low-rank matrices [J], Mathematical Programming, 2009.
- [12] E. J. Candes and B. Recht, Exact Matrix Completion Via Convex Optimization[J], Found Comput Math 2009.

- [13] S. Avidan, "Ensemble Tracking," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 2, pp. 261–271, 2007.
- [14] Malkov Yu A and Yashunin D A, Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. IEEE transactions on pattern analysis and machine intelligence 2020, 42(4): 824- 836.
- [15] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," IEEE Trans. Pattern Anal, Mach. Intell., vol. 37, no. 3, pp. 583–596, Mar. 2015.