MLVOT Project Report

Machine Learning for Visual Object Tracking

Oscar Le Dauphin

SCIA - 2026

Contents

1 Introduction 3
2 Dataset and Experimental Setup 3
2.1 Data Overview 3
3 TP 1: Single Object Tracking with Kalman Filter 4
3.1 Objective 4
3.2 Methodology 4
3.3 System Diagram 4
3.4 Real Data Testing 4
4 TP 2: IoU-based Tracking (Bounding-Box Tracker) 5
4.1 Objective 5
4.2 Methodology 5
4.3 System Diagram 6
4.4 Real Data Testing 6
5 TP 3: Kalman-Guided IoU Tracking 7
5.1 Objective 7
5.2 Methodology 7
5.3 System Diagram 8
5.4 Real Data Testing 8
6 TP 4: Appearance-Aware IoU-Kalman Tracker 9
6.1 Objective 9
6.2 Methodology 9
6.3 System Diagram 9
6.4 Real Data Testing and Conclusion 9

1 Introduction

This project aims to implement and analyze various object tracking algorithms, ranging from a simple Single Object Tracker (SOT) to a sophisticated Appearance-Aware Multi-Object Tracker (MOT). The goal is to understand the trade-offs between computational efficiency, geometric heuristics, and deep learning-based appearance features in maintaining object identities over time.

We progressively build a tracking system, evaluating each iteration against the `ADL-Rundle-6` dataset.

2 Dataset and Experimental Setup

2.1 Data Overview

The tracking algorithms are evaluated on the `ADL-Rundle-6` sequence from the MOT15 benchmark. This dataset features a busy pedestrian street captured by a static camera.

- **Detections (`det.txt`):** Provided pre-computed detections including bounding boxes (x, y, w, h) and confidence scores.
- **Ground Truth (`gt.txt`):** Used for quantitative evaluation (though primarily qualitative analysis is presented here).
- **Challenges:** The dataset includes frequent occlusions, crowded scenes with crossing trajectories, and variations in object scale.

Figure 1: Visual characteristics of the dataset: Crowded pedestrian street.

3 TP 1: Single Object Tracking with Kalman Filter

3.1 Objective

The objective of this first step is to implement a Kalman Filter to track a single moving object. In this specific experiment, we track a ball in a video sequence (`randomball.avi`) using a color/contour-based detector.

3.2 Methodology

We utilize a discrete linear Kalman Filter with a Constant Velocity (CV) model.

- **State Vector:** $x_k = \left[c_x, c_y, v_x, v_y\right]^T$, representing the centroid position and velocity.
- **Prediction:** The filter predicts the next state using the physical motion model:

$$\hat{x}_{k|k-1} = A\hat{x}_{k-1|k-1}$$

$$P_{k|k-1} = AP_{k-1|k-1}A^T + Q$$

- **Update:** Upon receiving a detection z_k (centroid), the filter corrects its prediction:

$$K_k = P_{k|k-1}H^T\left(HP_{k|k-1}H^T + R\right)^{-1}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\left(z_k - H\hat{x}_{k|k-1}\right)$$

3.3 System Diagram

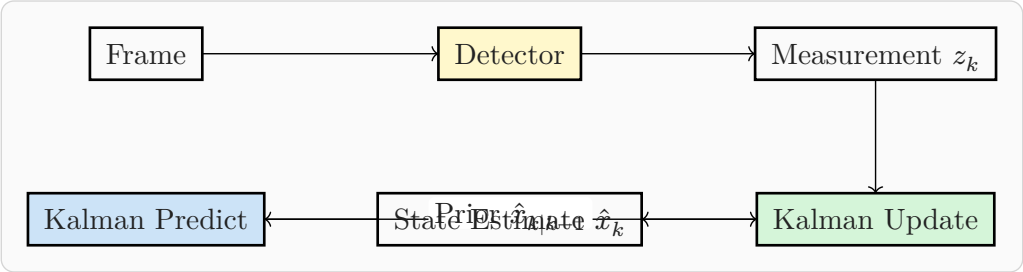

Figure 2: Kalman Filter predict-update cycle.

3.4 Real Data Testing

We tested the tracker on `randomball.avi`.

- **Performance:** The filter successfully smooths the noisy detections of the ball.
- **Occlusion:** When the ball is momentarily not detected, the prediction step allows the tracker to maintain a trajectory estimate, although it drifts if the signal is lost for too long.

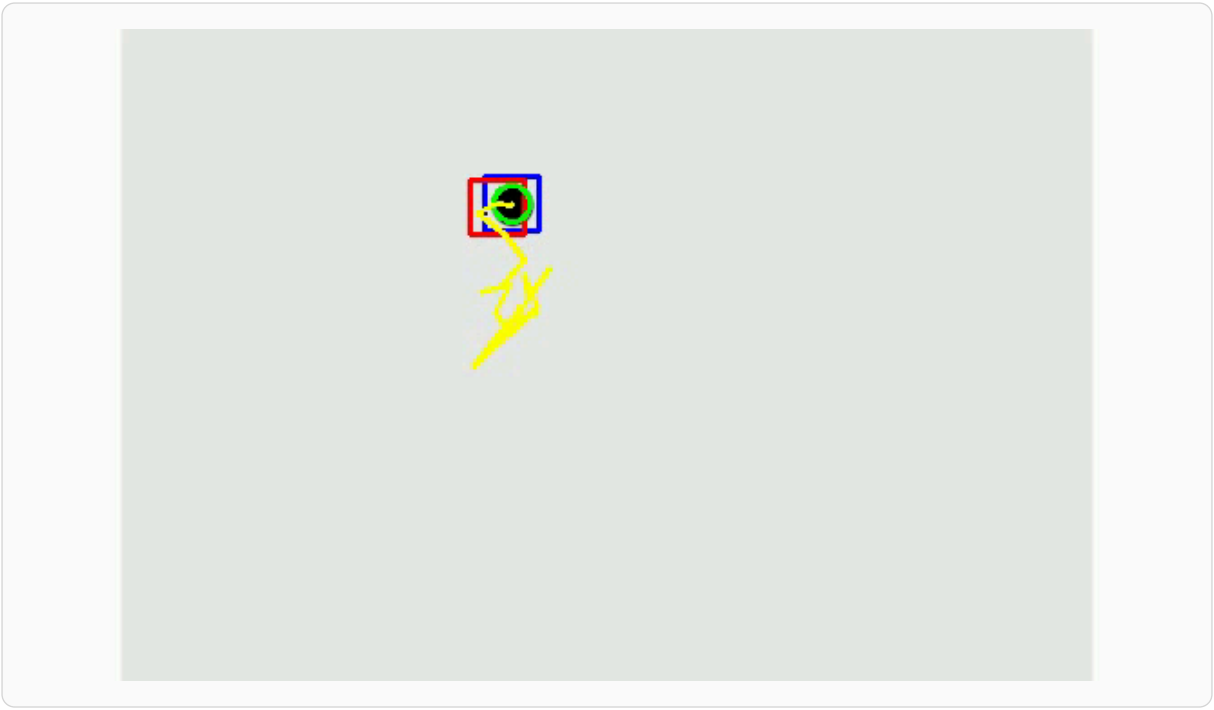

Figure 3: Kalman filter tracking results on the ball sequence.

4 TP 2: IoU-based Tracking (Bounding-Box Tracker)

4.1 Objective

Develop a basic Multi-Object Tracker (MOT) using Intersection over Union (IoU) as the sole metric for data association.

4.2 Methodology

This approach introduces track management:

1. **Association:** We compute an IoU matrix between existing tracks (last known position) and new detections.
2. **Hungarian Algorithm:** We maximize the total IoU to assign detections to tracks.
3. **Lifecycle:**
   - **Unmatched Detections** → New Tracks.
   - **Unmatched Tracks** → Incremented "missed" counter; deleted if threshold exceeded.

4.3 System Diagram

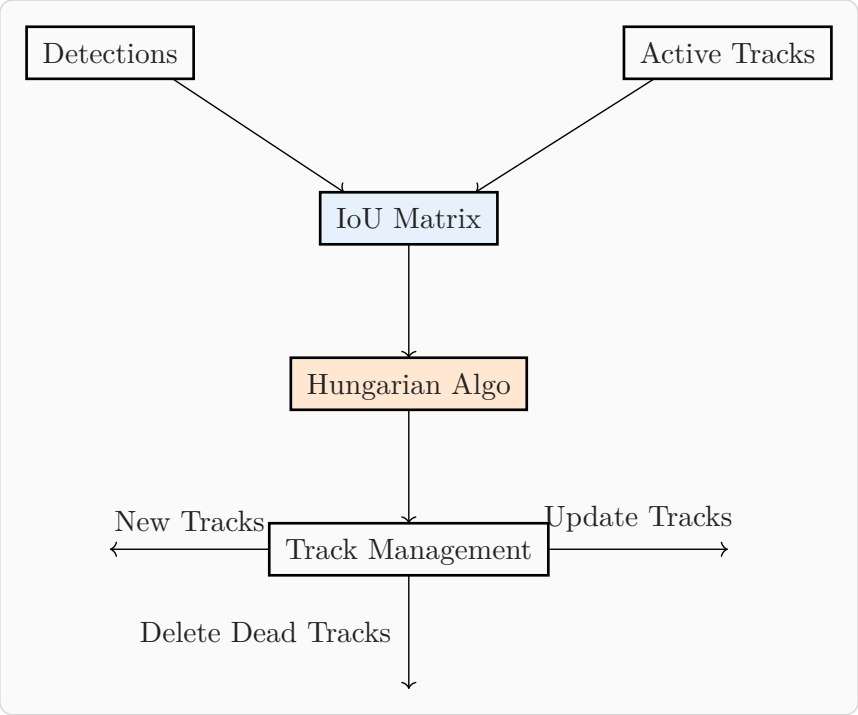

Figure 4: IoU-based association pipeline.

4.4 Real Data Testing

- **Success:** Works surprisingly well for pedestrians moving in distinct lanes without overlap.
- **Failure Cases:** The main limitation is the **Identity Switch**. When two pedestrians cross paths, their bounding boxes overlap significantly or one occludes the other. The IoU metric fails here, often swapping IDs or losing the occluded track.
- **Detector Sensitivity:** A significant limitation observed is the **over-sensitivity of the detector**. The bounding boxes often jitter or appear on background clutter, creating "ghost" tracks that exist for only a few frames. Since our IoU tracker initializes tracks immediately upon detection, this leads to a high number of short-lived, false positive tracks (ID fragmentation).

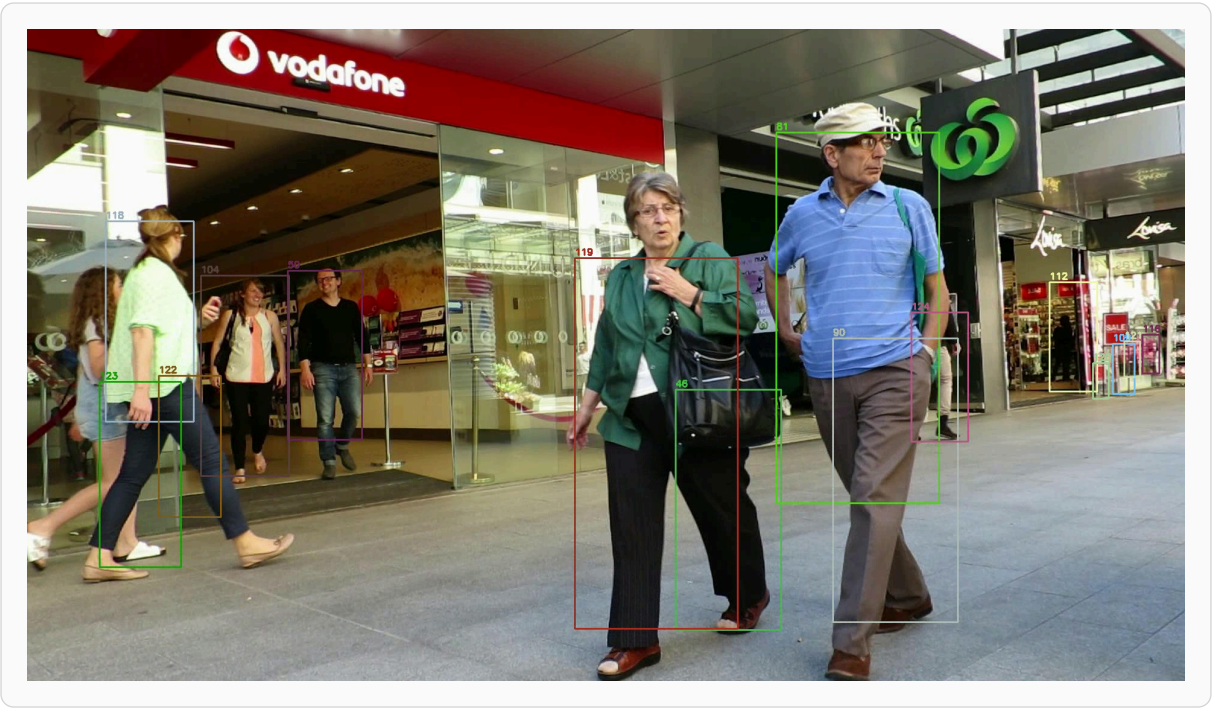

Figure 5: Identity switch occurring during a crossing event in IoU tracker.

5 TP 3: Kalman-Guided IoU Tracking

5.1 Objective

Improve the IoU tracker by predicting the future bounding box location using a Kalman Filter for each track before association.

5.2 Methodology

Instead of comparing detections to the **last known** position of a track, we compare them to the **predicted** position.

- Each track has its own Kalman Filter (x, y, v_x, v_y).
- **Prediction:** Before association, `track.predict()` moves the bounding box to where the object is expected to be.
- **Association:** IoU is calculated between `predicted_box` and `detection_box`.

This allows the association gate to "follow" the object, making it robust to faster movements where the IoU between frame t and $t-1$ might be low.

5.3 System Diagram

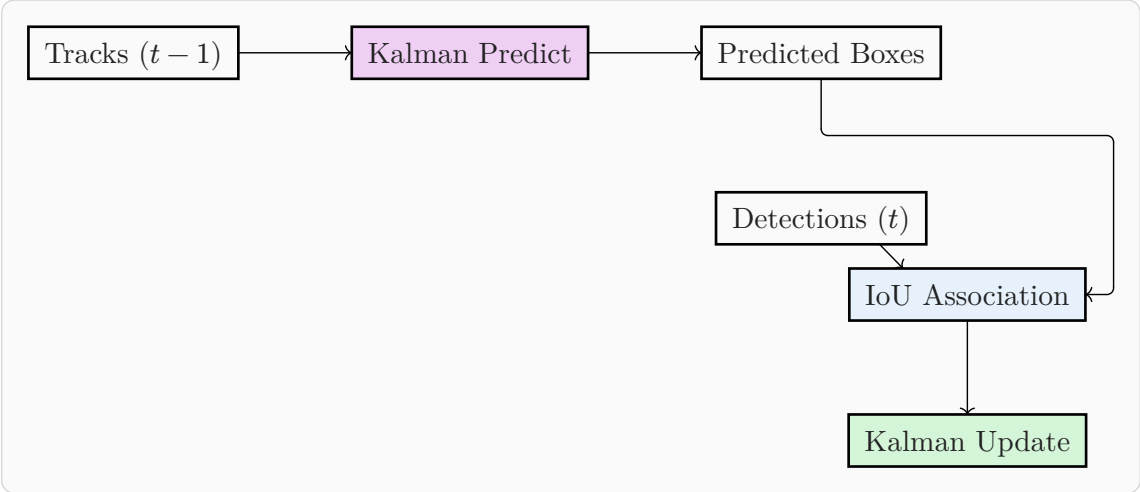

Figure 6: Kalman-guided tracking: Prediction precedes association.

5.4 Real Data Testing

- **Improvements:** Tracking is smoother. The system handles brief occlusions better because the Kalman filter continues to predict motion, maintaining a valid search region for re-association.
- **Limitations:** It still relies purely on geometry. If two objects have similar predicted positions (e.g., walking close together), the IoU metric alone cannot distinguish them. Furthermore, the **detector noise** issue persists; the Kalman filter tries to smooth the jittery bounding boxes, but large jumps in detection can still cause the filter to diverge or fail association.

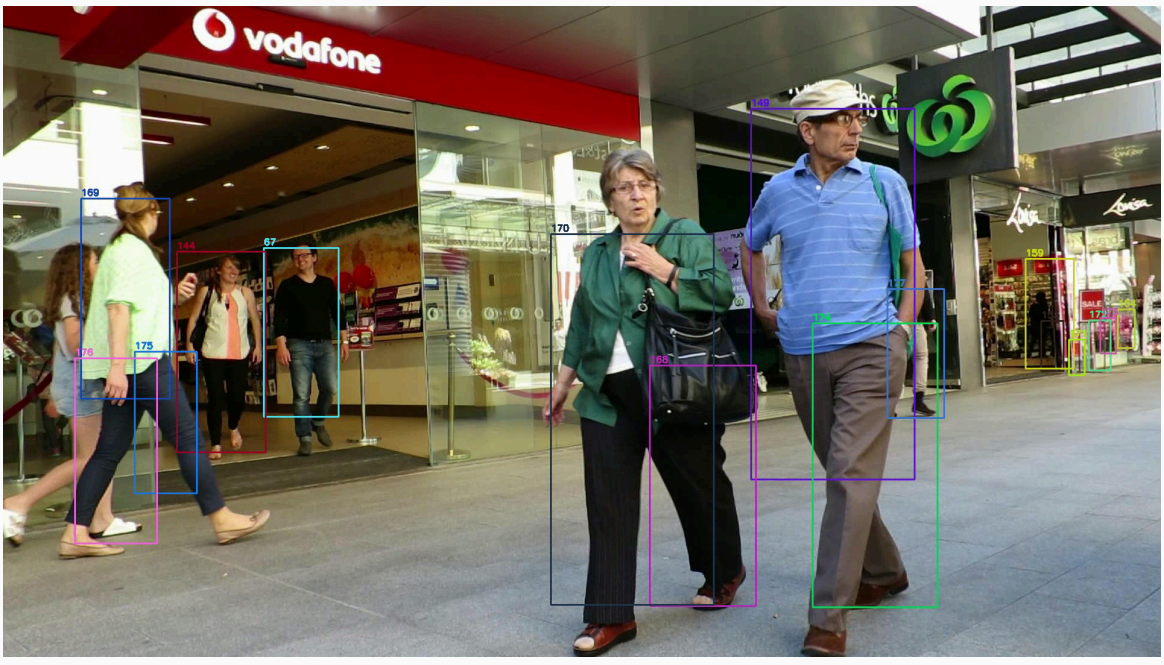

Figure 7: Kalman-Guided tracker maintaining ID through linear motion.

6 TP 4: Appearance-Aware IoU-Kalman Tracker

6.1 Objective

Integrate visual features (Re-Identification or ReID) to resolve identity switches and handle long-term occlusions where geometric prediction becomes unreliable.

6.2 Methodology

We utilize an **OSNet** model to extract a 512-dimensional feature vector for each detection crop.

- **Combined Cost:** The association cost matrix is a weighted sum:

$$C_{ij} = \alpha \cdot \left(1 - \text{IoU}_{ij}\right) + \beta \cdot \text{CosineDist}\left(\text{Feat}_i, \text{Feat}_j\right)$$

- **Workflow:**
  1. Predict track positions (Kalman).
  2. Extract visual features from current frame detections.
  3. Compute Combined Cost Matrix.
  4. Solve assignment (Hungarian).
  5. Update Kalman states and visual feature memory (moving average).

6.3 System Diagram

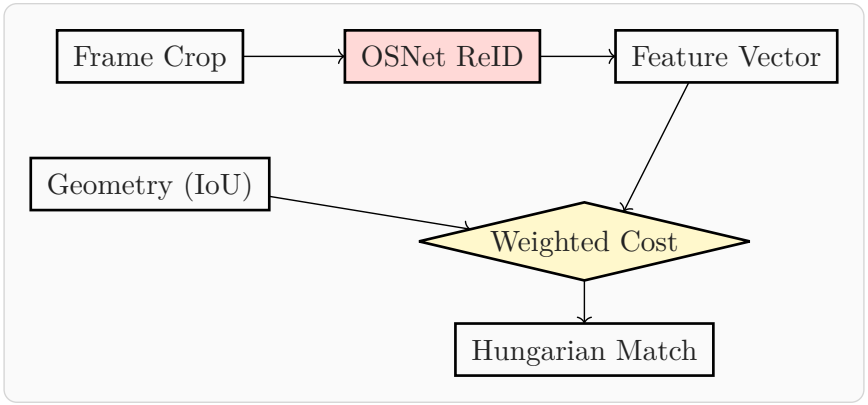

Figure 8: Appearance-aware association fusing geometry and deep features.

6.4 Real Data Testing and Conclusion

- **Robustness:** This tracker is the most robust. In the `ADL-Rundle-6` sequence, we observed that even after a full occlusion behind a pillar/object, the tracker could re-identify the person correctly upon reappearance because their visual appearance hadn't changed.
- **Trade-off:** The inference speed is significantly lower due to the heavy CNN forward pass for every detection crop.

- **Conclusion:** While geometric tracking (TP3) is sufficient for simple, sparse scenarios, appearance information (TP4) is critical for crowded, complex environments where trajectories intersect.

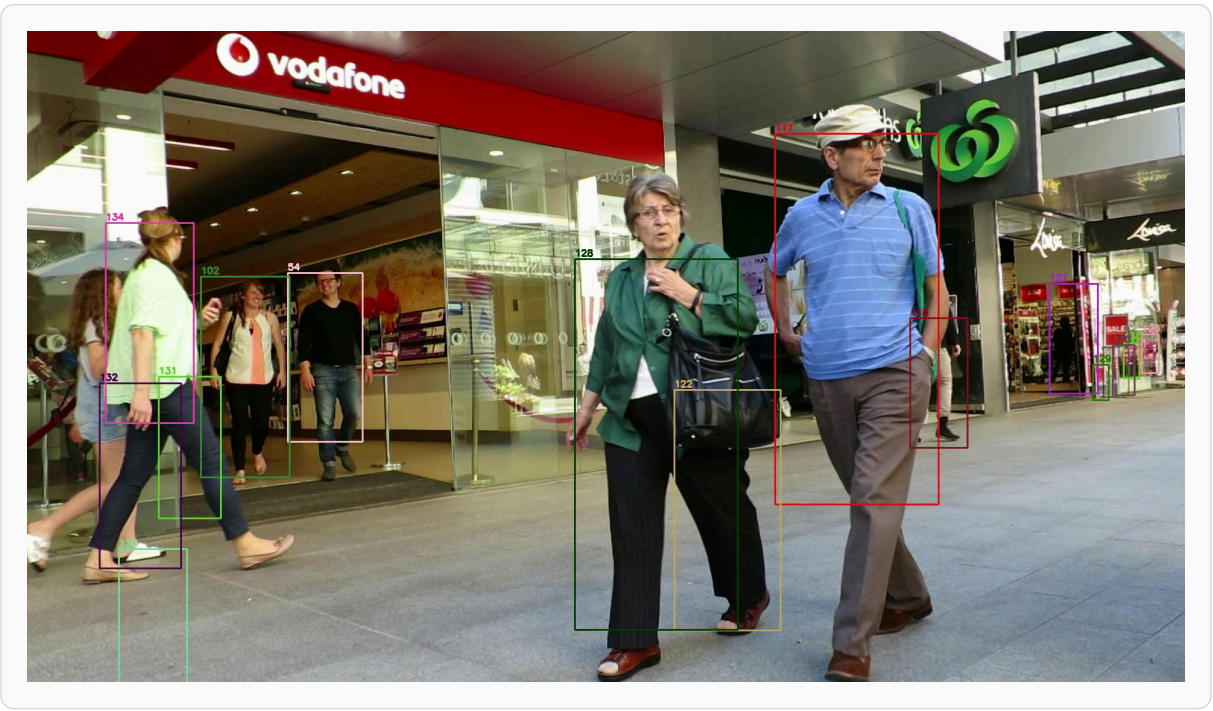

Figure 9: Successful re-identification after occlusion using visual features.