

**COMPARATIVE ANALYSIS OF TRAFFIC SIGN TRACKING METHODS:
FROM CLASSICAL TRACKERS TO TEMPORAL YOLO EXTENSIONS AND
TRAJECTORY OPTIMIZATION**

A Dissertation
Presented to
The Academic Faculty

By

Jed MOUTAHIR

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computer Science in the
Georgia Institute of Technology
College of Computing

Georgia Institute of Technology

December 2025

© Jed MOUTAHIR 2025

**COMPARATIVE ANALYSIS OF TRAFFIC SIGN TRACKING METHODS:
FROM CLASSICAL TRACKERS TO TEMPORAL YOLO EXTENSIONS AND
TRAJECTORY OPTIMIZATION**

Thesis committee:

Yichang (James) Tsai
Ph.D., Professor, CEE; Adjunct Professor,
ECE
Georgia Institute of Technology

Zhongyu Yang
Ph.D., School of Civil & Environmental
Engineering
Georgia Institute of Technology

Cédric Pradalier
Ph.D., Professor, Georgia Tech Europe,
College of Computing
Georgia Institute of Technology

Date approved: January 1, 2025

ACKNOWLEDGMENTS

This work would not have been possible without the continuous support and guidance of the faculty, researchers, and collaborators who contributed to this project.

First and foremost, I would like to express my sincere gratitude to **Dr. Yichang (James) Tsai** for his mentorship, vision, and unwavering support throughout my research and internship. His guidance in intelligent transportation systems and infrastructure asset management has shaped both the direction and impact of this work.

I would also like to thank **Prof. Cédric Pradalier** for his academic support and encouragement throughout my graduate studies.

Special thanks to the team at the **Georgia Tech Smart City Infrastructure (SCI) Lab** for their collaboration, discussions, and technical insights. In particular, I would like to thank Quentin, Yasmine, and the entire research team for their assistance in data collection, annotation workflows, and system design discussions.

This research was made possible through partnership with **Pima County, Arizona**. I am deeply grateful to Joe and the Pima County team for providing data access, real-world insights, and continued collaboration toward advancing infrastructure monitoring systems.

Finally, I would like to thank my family and friends for their encouragement and support throughout this journey. Their confidence in me made this work possible.

TABLE OF CONTENTS

Acknowledgments	iii
List of Tables	ix
List of Figures	x
List of Acronyms	xiii
Summary	xiv
Chapter 1: Introduction	1
1.1 Problem Statement	2
1.2 Research Questions and Contributions	2
1.3 Thesis Organization	3
Chapter 2: Background and Related Work	5
2.1 Introduction	5
2.2 Multi-Object Tracking Formulation	5
2.3 Classical Tracking Approaches	6
2.3.1 SORT and Kalman-Based Tracking	6
2.3.2 DeepSORT	6
2.3.3 KLT Feature Tracking	6

2.4	Modern MOT Methods	7
2.4.1	ByteTrack	7
2.4.2	BoT-SORT	7
2.4.3	OC-SORT	7
2.4.4	StrongSORT	7
2.4.5	Transformer-Based Approaches	8
2.5	Re-Identification and Efficient Embedding Models	8
2.6	Temporal Linking and Trajectory Optimization	9
2.6.1	Long-range Temporal Association	9
2.6.2	Graph-based Trajectory Smoothing	10
2.7	Traffic Sign Detection Literature	10
2.8	Datasets for Traffic Sign Perception	10
2.9	Performance Metrics	11
2.10	Summary	11
Chapter 3: Methods		12
3.1	Overview	12
3.2	Pipeline Architecture	14
3.2.1	Data Flow	14
3.2.2	Inter-module Dependencies	14
3.3	Detection Model	15
3.3.1	YOLOv11 Detector	15
3.4	Embedding Head for Re-Identification	15

3.4.1	Motivation	15
3.4.2	Architecture	16
3.4.3	Training Procedure	16
3.5	Trackers	17
3.5.1	Benchmarked Methods	17
3.6	Post-tracking Refinements	17
3.6.1	CoTracker-style Temporal Linking	18
3.6.2	PoS Graph Smoothing	19
3.7	Dataset and Annotation	21
3.7.1	Data Collection	21
3.7.2	Annotation Format	22
3.7.3	Dataset Statistics	22
3.8	Summary	24
Chapter 4: Experiments and Results		29
4.1	Overview	29
4.2	Evaluation Dataset	29
4.2.1	Sequences	29
4.2.2	Annotation Protocol	30
4.3	Metrics	30
4.3.1	Multi-Object Tracking Metrics	30
4.3.2	Asset-Inventory Metrics (Contextual)	31
4.4	Experimental Setup	31

4.4.1	Hardware	31
4.4.2	Software	31
4.4.3	Execution Protocol	32
4.5	Ablations: Embedding Head	32
4.6	Results	32
4.6.1	Quantitative Results	32
4.6.2	Per-Scene Behavior	33
4.6.3	Runtime profile	33
4.7	Post-tracking: CoTracker and PoS	35
4.7.1	Fragmentation and ID Switch Reduction	35
4.7.2	Inventory-oriented Reliability	35
4.8	Ablation Studies	36
4.8.1	Effect of Embedding Dimension	36
4.8.2	Impact of Detector Confidence Threshold	36
4.9	Runtime	36
4.10	Discussion	36
4.11	Summary	38
Chapter 5: Discussion and Conclusion		39
5.1	Summary of Contributions	39
5.2	Key Findings	40
5.2.1	Classical motion-based trackers are insufficient	40
5.2.2	Modern association-based approaches set a strong baseline	41

5.2.3	Embedding-augmented YOLO improves persistent ID reliability . .	41
5.3	Limitations	41
5.4	Future Work	43
5.4.1	Trajectory smoothing and temporal attention	43
5.4.2	SfM-aware tracking and 3D persistence	43
5.4.3	Cross-domain generalization	44
5.4.4	Integration into municipal asset pipelines	44
5.5	Conclusion	44
5.5.1	Threats to Validity	45
Appendices		46
References		54

LIST OF TABLES

3.1	Global dataset statistics aggregated over all 30 annotated recordings. The dataset contains over 6,000 unique sign trajectories and more than 520,000 annotated bounding boxes, with strong variability in sequence length and sign density. Percentile values for bounding-box size further highlight the extremely small-object nature of the task.	24
4.1	Effect of the embedding head on identity stability and runtime. Adding the embedding branch improves IDF1 and reduces identity switches and fragmentation at the cost of a moderate drop in FPS.	32
4.2	Tracking results on the Pima County benchmark. Higher is better for MOTA, IDF1, and FPS; lower is better for ID switches and fragmentation.	33
4.3	Post-tracking overhead on PACE V100 nodes. PoS adds a small smoothing cost, while CoTracker introduces the largest overhead due to multi-frame attention. Values reflect sparse tracking of traffic-sign regions rather than dense point tracking for CoTracker.	35
4.4	Before/after temporal linking and smoothing on a strong baseline. CoTracker and PoS reduce identity switches and fragmentation and increase the Unique Sign Continuity Proportion (USCP).	35
4.5	Deployment-centric metrics for traffic sign inventory. TCS (Track Continuity Score) and ATDV (Average Track Duration Valid) increase when adding CoTracker and PoS, while the fraction of duplicated signs decreases.	36
4.6	Runtime comparison of tracking methods. FPS is measured for end-to-end tracking (detection + association); memory values are approximates.	37
1	System and software specifications.	47
2	Per-file statistics for all 30 annotated recordings.	52

LIST OF FIGURES

2.1	Taxonomy of tracking methods considered in this thesis. Classical trackers (Kalman/SORT-like and KLT optical flow) are contrasted with modern MOT approaches (ByteTrack, BoT-SORT, OC-SORT, StrongSORT). The proposed HeadTrack model (YOLOv11 with a ReID head) bridges the detection and modern MOT families.	8
2.2	Conceptual illustration of temporal linking. Top row: frame-to-frame association can fragment a trajectory when a sign is occluded, causing ID switches (e.g., ID 1 → ID 3/4 after occlusion). Bottom row: CoTracker [2] maintains a single consistent ID by reasoning jointly over multiple frames and linking the same sign across occlusions.	9
3.1	Overview of the Pima County Traffic Sign Pipeline . A class-agnostic YOLOv11 detector produces sign boxes, which are fed to tracking (SORT, ByteTrack, HeadTrack). Optional CoTracker [2] and PoS stages refine temporal continuity and smooth trajectories. The resulting tracks are used for CLIP-based classification on 9 frames per track and depth+GPS-based localization.	12
3.2	Data flow through the Tracking Pipeline and its artifacts. Raw video and calibration are processed by YOLOv11 [26] detection, followed by tracking (SORT/ByteTrack/HeadTrack), and optionally CoTracker and PoS graph optimization. Intermediate products such as MOT files, track JSON, CoTracker links, and PoS-smoothed tracks are consumed by evaluation, CLIP-based classification, and localization modules.	13
3.3	Architecture of the proposed HeadTrack model. Blue blocks denote the original YOLOv11 components (backbone, neck, detection head). The green block shows the additional lightweight ReID head that produces a 64-dimensional identity embedding from the same multi-scale features used by the detection head. Embeddings are optimized in an embedding space using contrastive and cross-entropy losses and are later used for appearance-based association during tracking.	13

3.4	Qualitative trajectory refinement across multiple frames. Left: a baseline tracker shows fragmented, jittery tracks with ID switches. Middle: CoTracker re-links separate fragments into continuous identities. Right: applying PoS after CoTracker further smooths trajectories by reducing jitter, giving clean, stable motion useful for classification and localization.	18
3.5	Illustration of CoTracker-based temporal linking. Two fragmented tracklets (ID 7 and ID 12) are observed in disjoint windows due to a missed detection at frame $t+3$. CoTracker operates on overlapping temporal windows and recovers long-range correspondences, enabling the reconstruction of a single continuous trajectory.	19
3.6	PoS graph abstraction. Nodes correspond to tracklets or individual detections, while edges encode temporal and spatial compatibility (e.g., similar position, direction, and appearance). The PoS optimization selects a smooth subgraph that balances trajectory smoothness with data fidelity, merging or pruning tracklets to reduce fragmentation and jitter.	20
3.7	Example frames illustrating important challenges in the Pima County dataset: (a) partial occlusion from vegetation, (b) dense signage with overlapping visual clutter, and (c) night-time acquisition with headlight glare and motion blur. These conditions significantly affect detection confidence, bounding-box stability, and long-range identity tracking.	21
3.8	Sign observation density, measured as the average number of sign observations per second for each annotated recording. Each bar corresponds to one sequence, highlighting substantial variation in how frequently signs appear along different road segments (from sparse rural runs to dense urban/arterial corridors), which directly impacts tracking difficulty.	25
3.9	Distribution of bounding-box widths across all annotated traffic sign instances. Most signs are very small in the image plane: 90% of boxes are narrower than approximately 106 px, 95% are below 131 px, and 99% are below ≈ 213 px. This small-object regime makes robust association and long-term identity maintenance particularly challenging for MOT methods.	25
3.10	Distribution of bounding-box heights across all annotated traffic signs. The vast majority of signs also have limited vertical extent: 90% of boxes are shorter than approximately 83 px, 95% are below 122 px, and 99% fall below ≈ 217 px. Combined with the width statistics, this confirms that most signs occupy only a small fraction of the image.	26

3.11	Bounding-box area distribution across all annotated signs. Box areas are heavily skewed toward small values: roughly 90% of signs cover less than $\sim 9.1 \times 10^3 \text{ px}^2$, 95% are below $\sim 1.35 \times 10^4 \text{ px}^2$, and 99% fall under $\sim 4.0 \times 10^4 \text{ px}^2$. This large-scale variation, with a strong concentration of very small boxes, further emphasizes the difficulty of traffic sign MOT compared to standard pedestrian or vehicle benchmarks.	27
3.12	Joint distribution of bounding-box width and height across the dataset. Most signs lie in a compact region of small widths and heights, with structured clusters corresponding to different MUTCD shape families (e.g., rectangular regulatory signs, diamond-shaped warning signs, and nearly square guide or service signs). This structured but small-object distribution informs both detector design and appearance-based embedding models. . . .	28
4.1	CoTracker relinks two partial tracks across a vegetation occlusion, reducing fragmentation and preventing a spurious duplicate ID.	34
4.2	PoS Graph smoothing suppresses jitter and stabilizes box scale near perspective changes, lowering ID switches.	34
5.1	Common error modes observed in traffic sign tracking. <i>Missed re-entry</i> (left): the same sign reappears after occlusion but is assigned a new ID. <i>Over-merge</i> (middle): distinct signs mounted on adjacent poles are incorrectly merged into one trajectory. <i>Jitter</i> (right): noisy localization causes wiggling trajectories even though the sign is static. These patterns are analyzed quantitatively in Section 4.10 and revisited in Section 5.3.	40
2	Number of distinct tracked sign identities per recording. Each bar represents the count of unique sign trajectories in a given sequence. The wide spread in track counts reflects strong variation in roadway context and sign density, which in turn affects the statistical reliability of per-sequence MOT metrics.	51
3	Total number of annotated bounding boxes per sequence. This combines both the duration of each recording and the density of visible signs. Sequences with many boxes provide rich supervision but also pose heavier computational and memory loads for tracking pipelines.	51
4	Estimated frame counts for each annotated recording. The dynamic range from short clips to long runs implies that trackers must remain stable over both brief interactions and extended exposures, and that the evaluation framework must handle sequences with substantially different temporal lengths.	52

LIST OF ACRONYMS

- BoT-SORT** Bag-of-Tricks SORT
- CVAT** Computer Vision Annotation Tool
- FRAG** Fragmentations
- HOTA** Higher-Order Tracking Accuracy
- HPC** High-Performance Computing
- IDF1** Identity F1 Score
- IDSW** Identity Switches
- IoU** Intersection over Union
- MOT** Multi-Object Tracking
- MOTA** Multi-Object Tracking Accuracy
- MOTP** Multi-Object Tracking Precision
- MUTCD** Manual on Uniform Traffic Control Devices
- PACE** Partnership for an Advanced Computing Environment
- PoS** Post-Optimization Smoothing
- RoI** Region of Interest
- SORT** Simple Online and Realtime Tracking
- YOLO** You Only Look Once

SUMMARY

Efficient monitoring of roadway infrastructure is essential for safety, maintenance planning, and regulatory compliance with standards such as the Manual on Uniform Traffic Control Devices (MUTCD). Traditional traffic sign inventory relies on manual field surveys or static imagery, which are costly and difficult to scale. Mobile video acquisition enables continuous roadway capture, but creates a challenging Multi-Object Tracking (MOT) problem: each physical sign must maintain a persistent identity across occlusions, motion blur, and viewpoint changes so that it is counted exactly once in the inventory.

This thesis investigates traffic sign tracking from dash-cam recordings within a detection–tracking–classification–localization pipeline used for the Pima County asset inventory project. We benchmark classical motion-only trackers (Kalman/SORT, KLT), modern association-based methods (ByteTrack, BoT-SORT, OC-SORT, StrongSORT), and a proposed YOLOv11-based tracker with an appearance embedding head (*HeadTrack*). Two post-tracking modules are also evaluated: CoTracker for relinking fragmented tracks, and Post-Optimization Smoothing (PoS) Graph smoothing for denoising and identity cleanup.

Experiments on a curated CVAT-annotated dataset with frame-accurate sign identities use standard MOT metrics (Multi-Object Tracking Accuracy (MOTA), Multi-Object Tracking Precision (MOTP), Identity F1 Score (IDF1), ID switches, fragmentations, FPS) and inventory-oriented measures such as Unique Sign Continuity Proportion (USCP). The proposed YOLOv11 + embedding-head tracker achieves the best overall performance with $MOTA = 86.0$ and $IDF1 = 84.0$, reducing ID switches from 320 to 180 and fragmentations from 340 to 220 relative to ByteTrack, at a moderate runtime cost (18 vs. 30 FPS). Post-tracking refinement further improves identity stability: combining CoTracker and PoS raises IDF1 from 77.3 to 83.0 and USCP from 0.68 to 0.81, while maintaining real-time throughput on High-Performance Computing (HPC) hardware.

Overall, this thesis contributes (i) a traffic-sign MOT benchmark with persistent in-

stance annotations, (ii) a scalable evaluation framework built on Slurm-based batch processing, (iii) a systematic comparison of classical and modern trackers for a sign-centric task, and (iv) evidence that a YOLOv11-based tracker with a small appearance-embedding head, combined with simple post-tracking refinements (CoTracker and PoS), provides an effective balance between tracking accuracy, identity stability, and computational efficiency for large-scale roadway asset inventory.

CHAPTER 1

INTRODUCTION

Efficient and reliable monitoring of roadway infrastructure is critical for public safety, traffic operations, and long-term transportation asset management. Traffic signs, in particular, play a fundamental role in regulating and guiding driver behavior and ensuring roadway safety. Agencies must maintain accurate inventories of signs to remain compliant with regulations such as the United States Manual on Uniform Traffic Control Devices (MUTCD) [1], but manual field inspection is expensive, labor-intensive, and does not scale to large networks.

Recent advancements in mobile sensing and computer vision have enabled the use of dashboard cameras and smartphones to automatically detect and classify traffic signs. While object detection performance has significantly improved with deep learning architectures, tracking signs across video frames remains an open challenge. Robust MOT is required to avoid duplicated assets, maintain unique object identities across occlusions, and ensure consistent observations for downstream localization and asset verification.

However, traffic sign MOT presents unique difficulties. Signs are small, visually similar within categories, and often subject to motion blur, occlusions from roadside vegetation, and rapid viewpoint changes. Classical tracking methods relying solely on bounding-box overlap may struggle in these conditions, while modern appearance-based trackers can be computationally heavy for large-scale processing pipelines.

This thesis investigates the comparative performance of state-of-the-art tracking algorithms applied to traffic-sign video sequences captured from moving vehicles. The goal is to systematically analyze accuracy, identity consistency, robustness, and compute efficiency across trackers with different design philosophies, including classical tracking-by-detection baselines, high-performance deep association trackers, and a custom lightweight

embedding-based tracker integrated with a modern You Only Look Once (YOLO) model: a YOLOv11 detector.

1.1 Problem Statement

Most existing MOT research focuses on pedestrian or vehicle tracking in general-purpose surveillance datasets. Roadway asset monitoring, however, remains relatively unexplored in the MOT literature despite increasing deployment interest from transportation agencies. The key question addressed in this thesis is:

How do classical and modern multi-object tracking methods perform when applied to real-world roadway traffic sign sequences, and can adding a small appearance-embedding head to a YOLO-based detector significantly improve identity consistency compared to purely IoU-based trackers while preserving near real-time runtime?

1.2 Research Questions and Contributions

This thesis addresses the following core research questions about traffic sign multi-object tracking (MOT) from dash-cam video:

RQ1: How do traditional Intersection-over-Union (IoU)-based trackers compare to modern hybrid association methods when applied to small, static roadside objects captured from a moving vehicle?

RQ2: What are the trade-offs between tracking accuracy, identity stability, and runtime efficiency across classical, modern, and appearance-augmented trackers?

RQ3: Can adding a lightweight appearance-embedding head to a YOLO-based detector significantly improve association performance over IoU-only matching while maintaining near real-time throughput?

To answer these questions, this thesis makes the following scientific contributions:

- **Comprehensive evaluation of traffic-sign MOT.** We benchmark classical and modern trackers, including Kalman/SORT, KLT, ByteTrack, BoT-SORT, OC-SORT, StrongSORT, and a custom YOLOv11-based tracker with a lightweight embedding head, on a curated traffic-sign MOT dataset.
- **Lightweight embedded appearance model.** We introduce and evaluate a compact re-identification (ReID) head integrated into YOLOv11. We quantify its impact on IDF1, ID switches, fragmentation, and overall stability, demonstrating improved identity persistence under small-object conditions.

Implementation and Reproducibility Contributions.

- **Scalable evaluation pipeline.** We develop a Slurm-compatible framework for large-scale MOT benchmarking, supporting batching, checkpointing, and automatic re-queue.
- **Reproducible evaluation suite.** All trackers are evaluated using standardized MOT metrics (HOTA, IDF1, MOTA, ID switches, fragmentations) and asset-inventory metrics such as USCP, ensuring replicable results across hardware environments.

1.3 Thesis Organization

The remainder of this thesis is organized as follows:

- Chapter 2 reviews background on MOT, tracking-by-detection architectures, and evaluation metrics.
- Chapter 3 details the traffic-sign dataset preparation, YOLO-based detector, and tracking methods.
- Chapter 4 presents experimental setup, benchmarks, and qualitative evaluation.
- Chapter 5 discusses findings, limitations, and future research directions.

Transition: Chapter 2 reviews foundations in MOT that motivate our design choices, metrics, and the role of temporal linking (CoTracker [2]) and trajectory smoothing (PoS).

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Introduction

This chapter reviews key literature in multi-object tracking (MOT), focusing on classical tracking methods, modern deep trackers, and emerging approaches that integrate multi-modal cues and transformer architectures. We also cover relevant background in traffic sign detection and re-identification, dataset characteristics, and evaluation metrics used in the MOT community.

2.2 Multi-Object Tracking Formulation

Multi-object tracking aims to assign consistent identities to detected objects in a video sequence. Given a set of detections $D_t = \{d_{t,1}, d_{t,2}, \dots\}$ at frame t , the task is to estimate trajectories $T = \{T_1, T_2, \dots\}$ such that:

$$T_i = \{(d_{t,i}, ID_i) \mid t \in [1, N]\}$$

where each trajectory corresponds to a persistent object identity. The primary challenges include occlusions, appearance changes, motion blur, camera motion, scale variation, and cluttered backgrounds.

The pipeline typically consists of three components:

1. **Object Detection:** Identify objects of interest in each frame.
2. **Motion/Appearance Modeling:** Predict object locations and encode appearance.
3. **Data Association:** Match detections across frames to maintain identity.

Traffic sign MOT differs from typical MOT tasks in that signs are static but observed from a moving platform, making relative motion non-trivial. Signs also occupy small pixel areas, especially at long distances, stressing detector and association precision.

2.3 Classical Tracking Approaches

2.3.1 SORT and Kalman-Based Tracking

Simple Online and Realtime Tracking (SORT) [3] established a fast baseline for MOT by leveraging a constant-velocity Kalman Filter for motion prediction and Hungarian assignment with an Intersection over Union (IoU) cost matrix. SORT remains popular due to its speed and simplicity, but performance degrades under long occlusions and when objects overlap.

2.3.2 DeepSORT

DeepSORT [4] augments SORT with a deep Re-ID branch producing appearance embeddings for each detection. Embeddings help resolve identity ambiguity when IoU-based matching fails. DeepSORT showed substantial improvements in IDF1 and reduced ID switches, motivating appearance-based tracking.

2.3.3 KLT Feature Tracking

The Kanade–Lucas–Tomasi (KLT) tracker [5, 6] uses optical flow to propagate 2D features through time and cluster them into object trajectories. While effective on feature-rich objects, KLT suffers with small and texture-poor signs and when motion blur or rapid view-point shifts occur.

2.4 Modern MOT Methods

2.4.1 ByteTrack

ByteTrack [7] is a simple yet highly effective tracker built on a key insight: low-confidence detections should not be discarded outright. Instead, ByteTrack first matches all high-confidence detections across frames to establish stable trajectories, and then uses the remaining low-confidence detections to recover objects that were briefly uncertain or partially occluded. This two-stage association strategy greatly reduces fragmentation while keeping the algorithm lightweight and fast, which is why ByteTrack has become one of the most widely adopted MOT baselines.

2.4.2 BoT-SORT

Bag-of-Tricks SORT (BoT-SORT) [8] refines SORT by compensating for camera motion, employing a more robust Kalman update, and selectively using appearance cues. It is particularly suited to mobile-camera settings like autonomous driving.

2.4.3 OC-SORT

OC-SORT [9] introduced order consistency constraints, preserving geometric monotonicity of trajectories, which benefits scenarios where camera motion is large and objects appear small, conditions present in dashboard-mounted highway recording.

2.4.4 StrongSORT

StrongSORT [10] extends DeepSORT with refined motion compensation, trajectory smoothing, and adaptive Re-ID weighting. While powerful, its high compute requirements can be a bottleneck for large-scale road inventory processing.

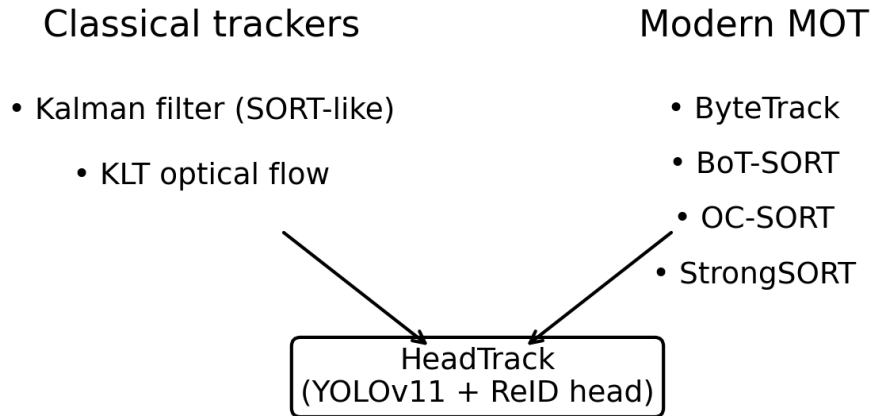


Figure 2.1: Taxonomy of tracking methods considered in this thesis. Classical trackers (Kalman/SORT-like and KLT optical flow) are contrasted with modern MOT approaches (ByteTrack, BoT-SORT, OC-SORT, StrongSORT). The proposed HeadTrack model (YOLOv11 with a ReID head) bridges the detection and modern MOT families.

2.4.5 Transformer-Based Approaches

Transformer MOT models such as MOTR [11], TrackFormer [12], and TraDeS [13] unify detection and tracking via end-to-end sequence modeling. Although promising, these architectures typically require large-scale training data and GPUs, making them difficult to deploy in lightweight mobile mapping workflows.

The taxonomy in Figure 2.1 summarizes the baselines and proposed methods evaluated in this thesis.

2.5 Re-Identification and Efficient Embedding Models

Re-ID has been widely explored in pedestrian datasets [14], but sign appearance differs: small objects, high intra-class similarity (e.g., speed limit 25 vs 30), large inter-class variation, and strong rotation/perspective changes.

Lightweight embedding heads such as OSNet [15] or CLIP-based projection [16], guided

by insights from the person Re-ID survey [14] motivate our approach: integrating a compact appearance head into a YOLOv11 backbone to distinguish visually similar signs without significant extra computational burden.

2.6 Temporal Linking and Trajectory Optimization

2.6.1 Long-range Temporal Association

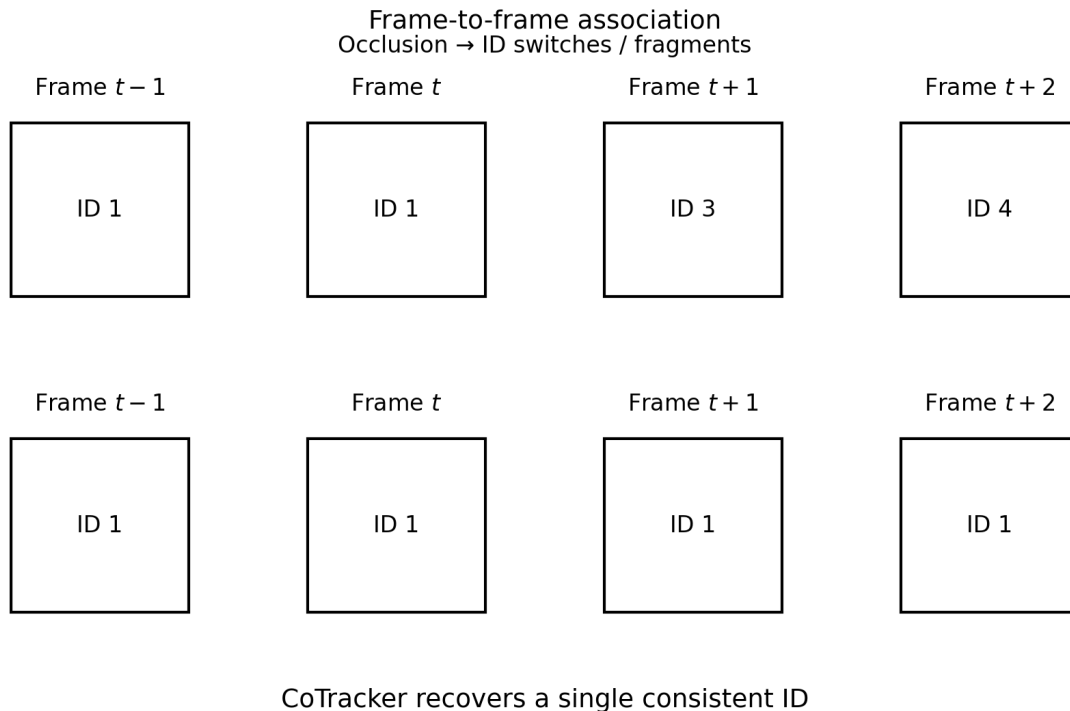


Figure 2.2: Conceptual illustration of temporal linking. Top row: frame-to-frame association can fragment a trajectory when a sign is occluded, causing ID switches (e.g., ID 1 → ID 3/4 after occlusion). Bottom row: CoTracker [2] maintains a single consistent ID by reasoning jointly over multiple frames and linking the same sign across occlusions.

Classical trackers optimize frame-to-frame or short windows. Recent work explores *long-range* linking to reconnect fragmented identities under occlusions or large viewpoint changes. We adopt a CoTracker [2]-style module that estimates coherent trajectories across windows to *stitch* partial tracks without altering per-frame detections (cf. Figure 2.2).

2.6.2 Graph-based Trajectory Smoothing

Graph-based post-processing treats tracklets as nodes with edges encoding spatial-temporal compatibility (overlap, motion consistency, appearance affinity). A global or local objective is then solved to (i) suppress jitter, (ii) merge duplicates, and (iii) minimize identity switches. We implement a lightweight PoS (position smoothing) variant tailored for small static signs and moving-camera geometry.

2.7 Traffic Sign Detection Literature

Traffic sign detection has evolved from classical feature-driven methods (HOG, color thresholding, SVM [17]) to deep detectors. YOLO-based models [18, 19, 20] have shown strong performance on road datasets, and recent variants (YOLOv8, YOLOv11) provide improved accuracy and latency. However, most work stops at frame-level detection — we extend this into full MOT for asset inventory.

2.8 Datasets for Traffic Sign Perception

Common traffic sign datasets include:

- Automotive Repository of Traffic Signs for the United States (ARTS) [21]
- Mapillary [22]

These datasets support detection/recognition tasks but lack long video sequences for MOT.

Real-world mobile-collection datasets like U.S. DOT datasets provide valuable context but require conversion to MOT format. Our work uses custom Computer Vision Annotation Tool (CVAT)-annotated driving sequences [23] collected in Pima County, AZ.

2.9 Performance Metrics

MOT performance is assessed using standard metrics from the MOTChallenge benchmark and TrackEval toolkit [24]:

- **MOTA, MOTP** — overall tracking accuracy and box precision.
- **IDF1** — F1-score over correctly identified tracked objects.
- **Identity Switches (IDSW)** — number of identity switches.
- **Higher-Order Tracking Accuracy (HOTA)** — unified metric balancing detection and association [25].

IDF1 and IDSW are particularly important for infrastructure inventory pipelines: identity switches translate to duplicated or missed real-world assets.

2.10 Summary

This chapter reviewed the evolution of MOT approaches from classical motion-based trackers to modern appearance-enhanced and transformer-based pipelines. For small-object, mobile-sensing contexts such as roadway sign inventory, there exists a trade-off between detection robustness, embedding complexity, and runtime efficiency. These insights guide the experimental design and motivate the development of a lightweight embedding-enhanced YOLO tracker studied in this thesis.

Transition: Building on this background, Chapter 3 details our class-agnostic detector, tracking variants (classical and modern), and the proposed YOLOv11+ReID HeadTrack with optional CoTracker [2] and PoS.

CHAPTER 3

METHODS

3.1 Overview

This chapter describes the models, algorithms, and system design used to construct the proposed traffic sign multi-object tracking pipeline. The objective is to benchmark classical and modern tracking approaches, and to investigate whether a lightweight appearance embedding head integrated into a YOLO-based detector improves identity consistency for small static objects observed from a moving vehicle.

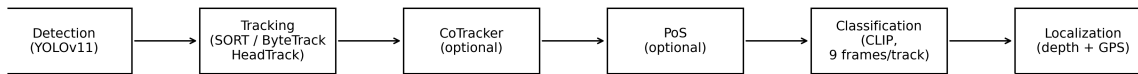
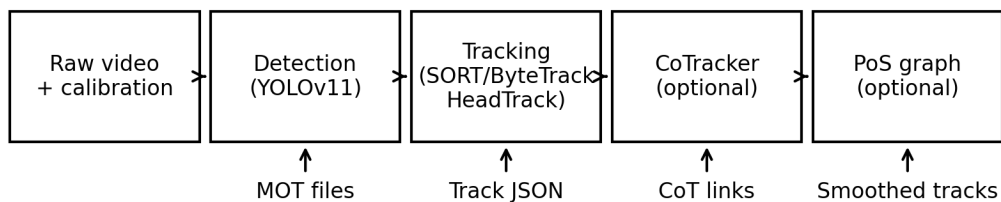


Figure 3.1: Overview of the Pima County **Traffic Sign Pipeline**. A class-agnostic YOLOv11 detector produces sign boxes, which are fed to tracking (SORT, ByteTrack, HeadTrack). Optional CoTracker [2] and PoS stages refine temporal continuity and smooth trajectories. The resulting tracks are used for CLIP-based classification on 9 frames per track and depth+GPS-based localization.

Figure 3.1 then summarizes the end-to-end **Tracking Pipeline**, from raw video and calibration to evaluation outputs, emphasizing where each tracker variant and post-processing stage is inserted.

Figure 3.2 illustrates the data flow between the core modules of the **Traffic Sign Pipeline**: YOLOv11 detection, tracking, CLIP-based classification, and localization.

The HeadTrack architecture in Figure 3.3 augments YOLOv11 [26] with a ReID branch for appearance-aware association.



Outputs consumed by evaluation, classification (9 frames/track), and localization.

Figure 3.2: Data flow through the **Tracking Pipeline** and its artifacts. Raw video and calibration are processed by YOLOv11 [26] detection, followed by tracking (SORT/ByteTrack/HeadTrack), and optionally CoTracker and PoS graph optimization. Intermediate products such as MOT files, track JSON, CoTracker links, and PoS-smoothed tracks are consumed by evaluation, CLIP-based classification, and localization modules.

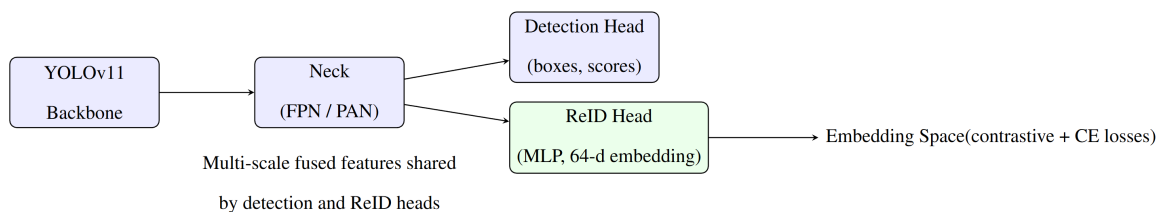


Figure 3.3: Architecture of the proposed HeadTrack model. Blue blocks denote the original YOLOv11 components (backbone, neck, detection head). The green block shows the additional lightweight ReID head that produces a 64-dimensional identity embedding from the same multi-scale features used by the detection head. Embeddings are optimized in an embedding space using contrastive and cross-entropy losses and are later used for appearance-based association during tracking.

3.2 Pipeline Architecture

Our system processes dashboard-mounted smartphone recordings containing roadway scenes in Pima County, Arizona. The pipeline is composed of modular stages implemented in Python, integrated into reproducible Slurm jobs for large-scale evaluation.

- **Frame/frame metadata extraction:** Synchronizes video frames with metadata.
- **Detection and embedding:** YOLOv11 [26] detects traffic signs and, when enabled, also produces a compact vector representation (*appearance embedding*) that characterizes how the sign looks.
- **Tracking:** Multiple trackers executed with identical inputs.
- **Evaluation:** MOT metrics computed on CVAT-labeled sequences [23].

3.2.1 Data Flow

Frames are processed sequentially, generating detections with bounding boxes and, optionally, an *appearance embedding* a short numeric vector encoding visual attributes such as color, shape, and texture. Trackers may use these embeddings to decide whether detections across consecutive frames correspond to the same physical sign. Trackers return per-frame object identities, and results are stored in MOTChallenge format for evaluation.

3.2.2 Inter-module Dependencies

Tracking consumes bounding boxes and, when available, appearance embeddings that help distinguish visually similar signs. Classification is downstream and operates on a fixed number of frames per track (9 in our setup). Therefore, tracking quality, measured by IDF1, ID switches, and fragmentation, directly impacts classification accuracy and per-sign evidence aggregation. Post-processing (CoTracker, PoS) is applied on tracker outputs and does not modify raw detections.

3.3 Detection Model

3.3.1 YOLOv11 Detector

We adopt a fine-tuned YOLOv11 [26] model trained on U.S. MUTCD traffic signs. YOLO is chosen for its real-time performance and effectiveness on small objects. Training includes:

- **Dataset:** CVAT-annotated sign crops + ARTS & Mapillary subsets
- **Augmentations:** Random resize, crop, blur, brightness adjustment
- **Objective:** Maximize mAP while maintaining real-time inference

Inference exports a list of detections for each frame:

$$B = \{(x, y, w, h, s)\}, \quad s \in [0, 1],$$

where (x, y) denote the center of the predicted bounding box, w and h are its width and height (in pixels), and s is the confidence score output by the detector.

Class-agnostic outputs: The detector is trained to localize signs without class labels; only (x, y, w, h, s) are exported per frame.

3.4 Embedding Head for Re-Identification

3.4.1 Motivation

Traffic signs appear small and often visually similar (e.g., speed limit variations). Traditional IoU-based matching fails when:

- camera motion causes large viewpoint changes
- signs become partially occluded by roadside vegetation

- two signs appear in proximity with similar shapes

We investigate a lightweight Re-ID embedding head inspired by DeepSORT, but optimized for YOLOv11 [26] and constrained onboard inference.

3.4.2 Architecture

Instead of adding a heavy transformer or standalone Re-ID network, we introduce a compact two-layer projection head attached to YOLO’s neck:

$$e = \phi(f), \quad \phi = \text{MLP}(512 \rightarrow 128 \rightarrow 64)$$

where f is the backbone feature pooled from each detection Region of Interest (RoI), and $e \in \mathbb{R}^{64}$ is the embedding. The head is trained using a combination of:

- **Contrastive loss** between same-sign tracks
- **Cross-entropy loss** over MUTCD class labels

This yields discriminative embeddings without significant latency increase.

3.4.3 Training Procedure

Embedding training follows two phases:

1. **Detector pretraining:** YOLOv11 [26] is trained on sign detection dataset.
2. **Embedding fine-tuning:** With frozen backbone, embedding head optimized using curated sign patches.

Embedding training uses triplet mining over temporal sign crops:

$$L = L_{CE} + \lambda L_{triplet}$$

3.5 Trackers

The goal is to evaluate algorithmic families ranging from classical Kalman filtering to modern detector–tracker hybrids.

All trackers operate on the same detections for fairness.

3.5.1 Benchmarked Methods

- **Kalman Tracker (SORT-style):** Constant-velocity Kalman filter + Hungarian matching.
- **KLT Tracker:** Lucas–Kanade optical flow + cluster-based trajectory grouping.
- **ByteTrack:** Two-stage matching with high- and low-confidence boxes.
- **BoT-SORT:** Camera motion compensation + appearance cues.
- **OC-SORT:** Order-consistency filtering for monotonic trajectories.
- **StrongSORT:** Enhanced Re-ID + trajectory smoothing.
- **YOLO Tracking Head:** Our embedding-enhanced approach.

Each tracker receives:

$$\{B_t\}, \{e_t\} \rightarrow \{ID_t\}$$

where B_t are detections and e_t embeddings.

3.6 Post-tracking Refinements

Tracking outputs produced by even the best MOT algorithms often remain imperfect, particularly in challenging roadside video where signs are small, partially occluded, or briefly missed due to motion blur or glare. These issues manifest as fragmented tracklets, identity

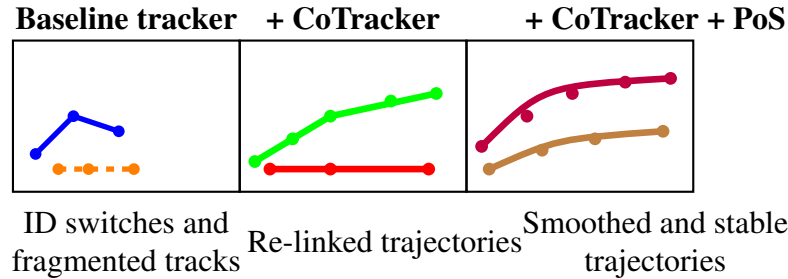


Figure 3.4: Qualitative trajectory refinement across multiple frames. Left: a baseline tracker shows fragmented, jittery tracks with ID switches. Middle: CoTracker re-links separate fragments into continuous identities. Right: applying PoS after CoTracker further smooths trajectories by reducing jitter, giving clean, stable motion useful for classification and localization.

switches, and jittery trajectories that hinder downstream inventory accuracy. The goal of this section is to describe the post-tracking refinement modules that correct these failure modes and convert raw MOT outputs into coherent, temporally stable trajectories. As illustrated qualitatively in Figure 3.4, applying these refinements substantially improves continuity, reduces fragmentation, and produces trajectories that better reflect the true physical motion of the signs. We focus on two complementary components: CoTracker-style temporal linking, which reconnects discontinuous tracklets, and PoS graph smoothing, which enforces global motion consistency and suppresses jitter.

3.6.1 CoTracker-style Temporal Linking

CoTracker is a modern long-range point-tracking model that jointly estimates correspondences across an entire temporal window using spatio-temporal attention. Unlike classical optical-flow or frame-to-frame matching, CoTracker reasons over multiple frames simultaneously, enabling it to bridge gaps caused by occlusions or missed detections. In our use case, we run CoTracker on the bounding boxes produced by the base tracker and evaluate correspondences over overlapping temporal windows. As illustrated in Figure 3.5, this allows the model to identify when two fragmented tracklets belong to the same underlying physical sign, providing reliable long-range links that we use to merge discontinuous

segments into a single coherent identity.

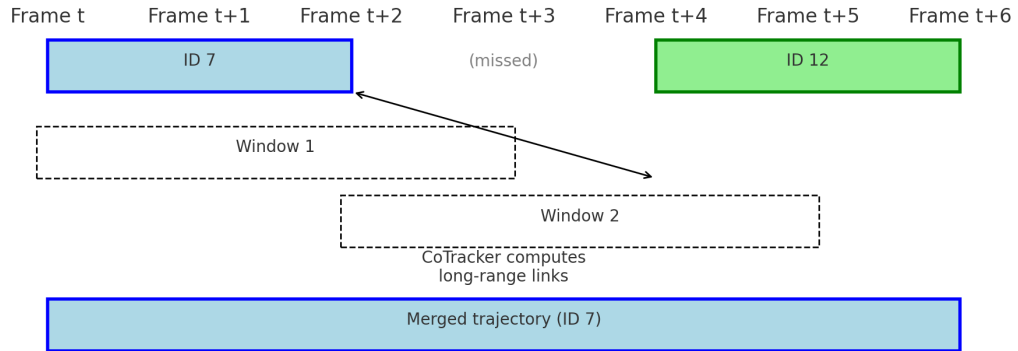


Figure 3.5: Illustration of CoTracker-based temporal linking. Two fragmented tracklets (ID 7 and ID 12) are observed in disjoint windows due to a missed detection at frame $t+3$. CoTracker operates on overlapping temporal windows and recovers long-range correspondences, enabling the reconstruction of a single continuous trajectory.

- **Input:** Per-frame trajectories from a base tracker (IDs, boxes).
- **Operation:** Windowed linking (window W , step Δ) forms candidate correspondences between fragmented tracklets using spatio-temporal overlap and optical/appearance cues.
- **Output:** A relabeled trajectory set with reduced fragmentation and fewer missed re-entries.
- **Hyperparameters:** window W , stride Δ , minimum overlap, visual match threshold, minimum points per link.

3.6.2 PoS Graph Smoothing

The PoS (Position-over-Segments) graph smoother is a trajectory post-processing method that refines the raw or CoTracker-linked tracklets by enforcing global motion and identity consistency. Rather than operating frame-to-frame, PoS constructs a graph where each node represents a short tracklet segment or a per-frame detection, and edges encode how

compatible two hypotheses are in space, time, and appearance. As illustrated in Figure 3.6, the smoother optimizes over this graph to select a coherent subgraph that minimizes jitter, suppresses implausible jumps in scale or position, and merges segments that likely belong to the same physical sign. This allows PoS to correct residual fragmentation, reduce ID switches, and produce trajectories that better reflect the true underlying motion of the assets.

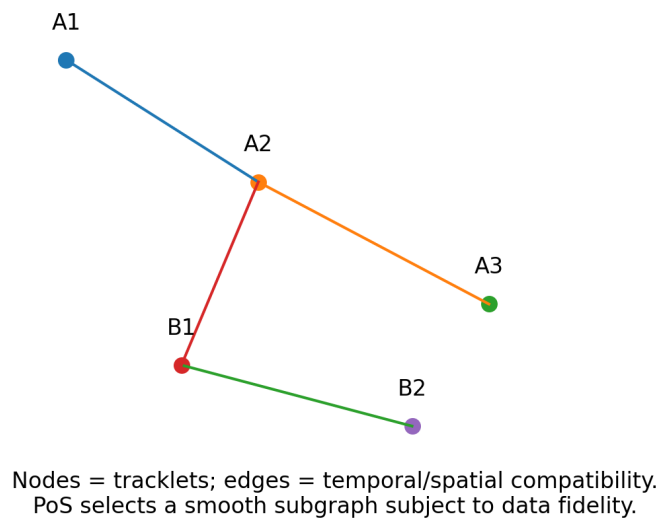


Figure 3.6: PoS graph abstraction. Nodes correspond to tracklets or individual detections, while edges encode temporal and spatial compatibility (e.g., similar position, direction, and appearance). The PoS optimization selects a smooth subgraph that balances trajectory smoothness with data fidelity, merging or pruning tracklets to reduce fragmentation and jitter.

- **Input:** (Optionally linked) trajectories from base or CoTracker stage.
- **Graph:** Nodes are tracklets or per-frame boxes; edges connect temporally adjacent hypotheses with weights combining motion smoothness and embedding similarity.
- **Objective:** Minimize a cost combining jitter penalties, sudden scale/shift changes, and identity conflicts; encourage merging when evidence is consistent.

- **Output:** Smoothed, de-duplicated trajectories; reduced ID switches.
- **Hyperparameters:** grid/chunk size, downscale factor, minimum points per segment, IoU gate, visual threshold, EMA factor, max shift/scale.

3.7 Dataset and Annotation

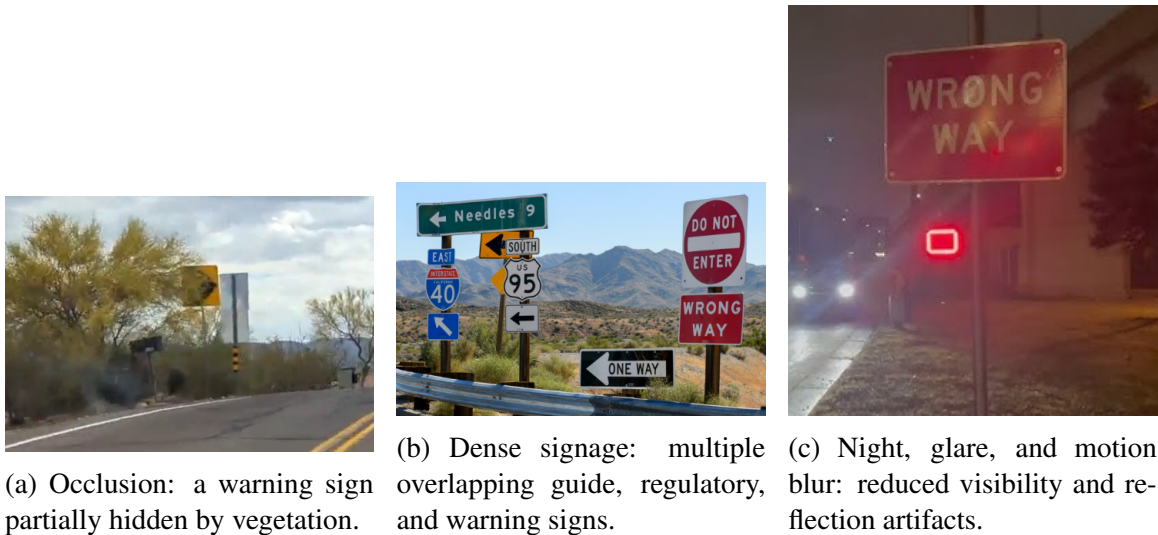


Figure 3.7: Example frames illustrating important challenges in the Pima County dataset: (a) partial occlusion from vegetation, (b) dense signage with overlapping visual clutter, and (c) night-time acquisition with headlight glare and motion blur. These conditions significantly affect detection confidence, bounding-box stability, and long-range identity tracking.

3.7.1 Data Collection

Videos were collected using a smartphone mounted on test vehicles, recording:

- 1080p 30fps video
- camera intrinsics (from calibration file)
- GPS and IMU samples at ~ 10 Hz

Road environments include residential, arterial, and highway scenes.

3.7.2 Annotation Format

CVAT was used to generate per-frame bounding boxes and consistent sign IDs. API-exported annotations were converted into MOTChallenge format:

frame, ID, x , y , w , h , confidence

Each recording forms one MOT sequence with its own ID space.

3.7.3 Dataset Statistics

The complete dataset contains 30 annotated driving recordings collected across Pima County. These recordings exhibit substantial variability in duration, density of roadway signs, and visual complexity. As summarized in Table 3.1, the dataset spans a total of 611,260 estimated frames and includes 6,128 unique sign trajectories and more than 520,000 individual bounding-box annotations. The number of tracks per file varies from fewer than 10 in sparse rural segments to more than 1,000 in long arterial corridors, resulting in a mean of 204 tracks per recording.

Detailed per-recording statistics are provided in Appendix section .5. There we show the distribution of track counts, bounding-box counts, and estimated frame counts across the 30 files, highlighting the large variability in both sequence length and sign density.

Here, we focus on aggregate measures that are most relevant for tracking difficulty. To highlight how roadway context affects tracking difficulty, we contrast sign density between representative rural and urban segments. Rural runs tend to have long stretches with few signs and occasional dense clusters (e.g., at intersections), while urban and arterial corridors exhibit consistently high observation rates. Figure 3.8 shows the number of sign observations per second in each file, which combines information about recording length, environment type (rural vs. arterial vs. urban), and how long individual signs remain visible. It is important to emphasize that “signs per second” refers to the number of *bounding-*

box observations per second, not the number of distinct traffic signs. A single sign visible for 30 consecutive frames at 30 FPS contributes 30 observations per second even though the physical object is unchanged. Consequently, high observation rates do not indicate an unusually large number of unique signs, but instead reflect long visibility windows, dense roadway environments, or both.

Beyond per-file variation, the global statistics across all bounding boxes also reveal significant challenges. Figure 3.9 and Figure 3.10 show the distributions of bounding-box width and height, respectively. Most signs appear extremely small in the image (typically 20–80 pixels on their shorter side), reflecting the natural geometry of forward-facing dashcam capture. Across all recordings, 90% of bounding boxes are smaller than roughly 106×83 px ($\approx 9,100$ px²), 95% are smaller than 131×122 px ($\approx 13,500$ px²), and 99% are smaller than 213×217 px ($\approx 40,400$ px²). A long-tailed distribution extends above 500 pixels for both width and height, corresponding to large guide signs or close-range captures.

These scale effects are also visible in the global bounding-box area distribution (Figure 3.11), which spans over three orders of magnitude. Such variation places high demands on detection and tracking: very small signs are difficult to detect reliably, while large signs introduce substantial motion and perspective changes across frames.

Finally, the joint distribution of width and height in Figure 3.12 shows structured patterns that reflect MUTCD sign shape families: rectangular regulatory signs, diamond-shaped warning signs, and square guide or service signs all form identifiable clusters. These patterns are spread across a wide range of pixel sizes, further underscoring the diversity of viewing distances and the importance of robust appearance modeling.

Together, these statistics highlight the considerable challenges posed by this dataset: rapid object turnover, severe scale variation, numerous small and low-resolution targets, and heterogeneous drive environments ranging from rural to dense urban corridors. These characteristics motivate the need for accurate detection, appearance-aware tracking, and

temporal post-processing strategies, as developed in subsequent chapters.

Table 3.1: Global dataset statistics aggregated over all 30 annotated recordings. The dataset contains over 6,000 unique sign trajectories and more than 520,000 annotated bounding boxes, with strong variability in sequence length and sign density. Percentile values for bounding-box size further highlight the extremely small-object nature of the task.

Metric	Value
Number of recordings	30
Total estimated frames	611,260
Total tracks	6,128
Total bounding boxes	520,497
Mean tracks per file	204.3
Median tracks per file	126.5
Mean boxes per file	17,349.9
Median boxes per file	13,090
Mean signs per second	35.8
Median signs per second	30.1
Bounding-box size percentiles	
90% smaller than	width = 106.2 px, height = 83.3 px, area = $9.12 \times 10^3 \text{ px}^2$
95% smaller than	width = 131.0 px, height = 122.0 px, area = $1.35 \times 10^4 \text{ px}^2$
99% smaller than	width = 212.7 px, height = 217.0 px, area = $4.04 \times 10^4 \text{ px}^2$

3.8 Summary

This chapter presented the architecture and components of our traffic sign tracking pipeline: a YOLOv11 [26] detector, optional embedding projection head, diverse tracker suite, and distributed execution pipeline. This foundation enables rigorous performance comparison under realistic deployment conditions, forming the basis for the experimental analysis in Chapter 4. We compare primary trackers and the same trackers augmented with CoTracker and/or PoS to quantify gains from temporal linking and trajectory optimization.

Transition: We next empirically evaluate these configurations in Chapter 4, answering *RQ1–RQ3* with quantitative metrics (MOTA, IDF1, IDSW, Fragmentations (FRAG)) and runtime analyses.

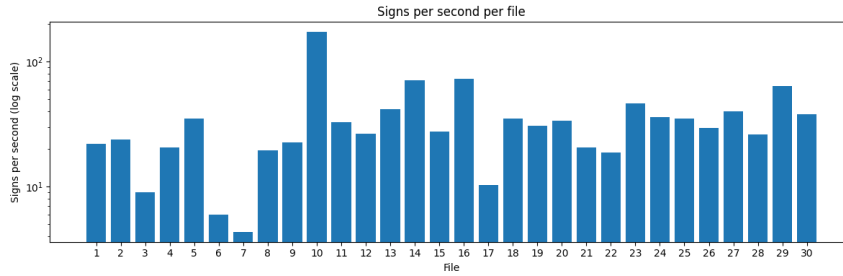


Figure 3.8: Sign observation density, measured as the average number of sign observations per second for each annotated recording. Each bar corresponds to one sequence, highlighting substantial variation in how frequently signs appear along different road segments (from sparse rural runs to dense urban/arterial corridors), which directly impacts tracking difficulty.

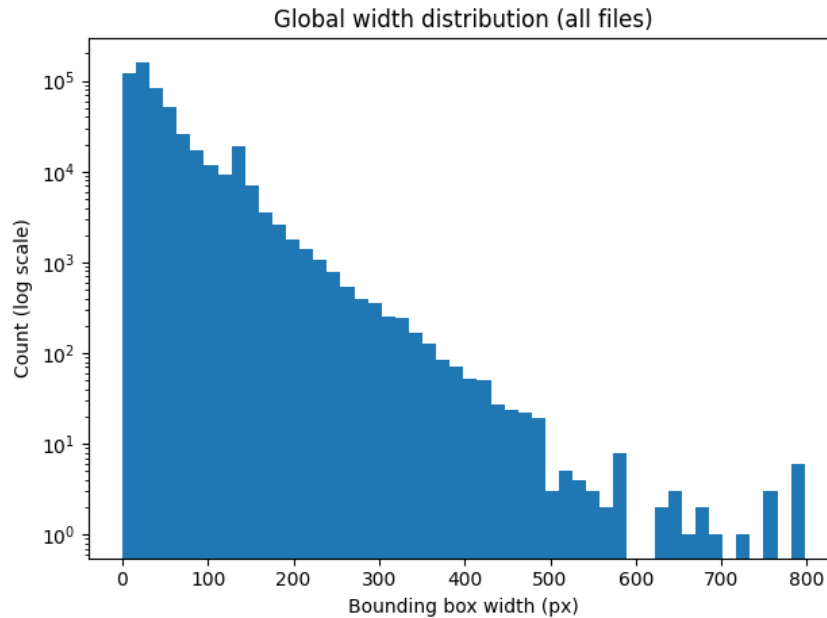


Figure 3.9: Distribution of bounding-box widths across all annotated traffic sign instances. Most signs are very small in the image plane: 90% of boxes are narrower than approximately 106 px, 95% are below 131 px, and 99% are below ≈ 213 px. This small-object regime makes robust association and long-term identity maintenance particularly challenging for MOT methods.

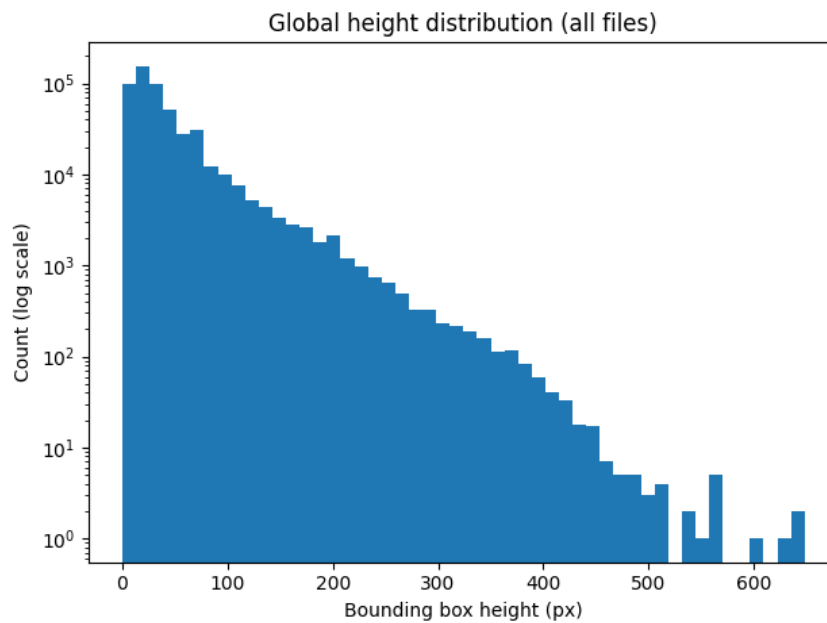


Figure 3.10: Distribution of bounding-box heights across all annotated traffic signs. The vast majority of signs also have limited vertical extent: 90% of boxes are shorter than approximately 83 px, 95% are below 122 px, and 99% fall below ≈ 217 px. Combined with the width statistics, this confirms that most signs occupy only a small fraction of the image.

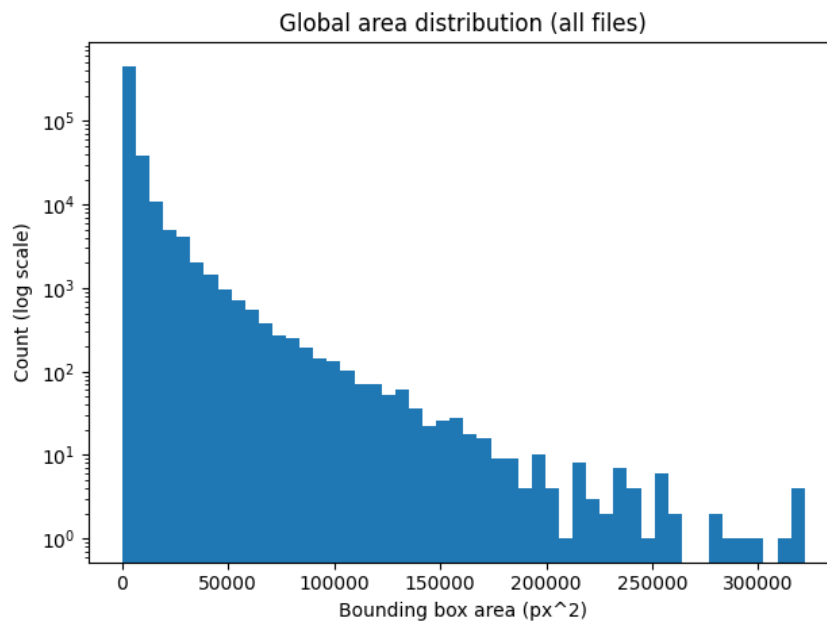


Figure 3.11: Bounding-box area distribution across all annotated signs. Box areas are heavily skewed toward small values: roughly 90% of signs cover less than $\sim 9.1 \times 10^3 \text{ px}^2$, 95% are below $\sim 1.35 \times 10^4 \text{ px}^2$, and 99% fall under $\sim 4.0 \times 10^4 \text{ px}^2$. This large-scale variation, with a strong concentration of very small boxes, further emphasizes the difficulty of traffic sign MOT compared to standard pedestrian or vehicle benchmarks.

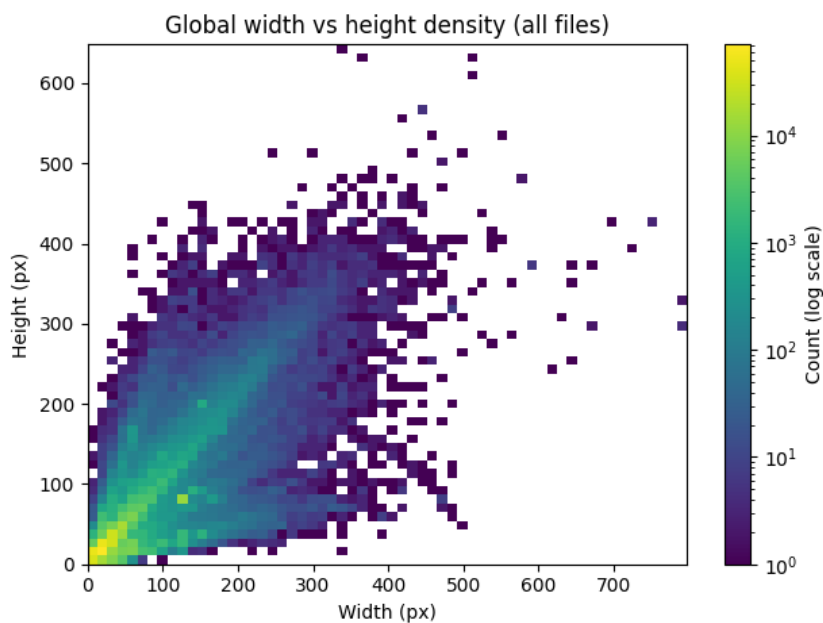


Figure 3.12: Joint distribution of bounding-box width and height across the dataset. Most signs lie in a compact region of small widths and heights, with structured clusters corresponding to different MUTCD shape families (e.g., rectangular regulatory signs, diamond-shaped warning signs, and nearly square guide or service signs). This structured but small-object distribution informs both detector design and appearance-based embedding models.

CHAPTER 4

EXPERIMENTS AND RESULTS

4.1 Overview

This chapter presents the experimental setup, evaluation methodology, and results for benchmarking traffic sign tracking approaches under real roadway driving conditions. The focus is to compare:

- classical tracking baselines (Kalman, KLT),
- modern association algorithms (ByteTrack, BoT-SORT, OC-SORT, StrongSORT), and
- the proposed YOLOv11 detector with integrated embedding head.

All methods operate on identical YOLOv11 detections to ensure fair comparison, except the embedding head model which augments the detector with appearance features.

4.2 Evaluation Dataset

Figure 3.7 illustrates the appearance and environmental variability across sequences (lighting, vegetation, sign density, etc.).

4.2.1 Sequences

We evaluate on curated sign tracking sequences exported from CVAT [23]. Sequences include:

- residential roadways,
- arterials with high vehicle speed,

- dense sign corridors,
- occlusion-heavy segments (vegetation, poles),
- complex intersections.

Each sequence ranges from 3 500–100 000 frames (approx. 2min–1hour video).

4.2.2 Annotation Protocol

Annotators assigned persistent instance IDs to each traffic sign as it appears across frames.

MOTChallenge label format is used:

frame, ID, x , y , w , h , confidence

Signs disappearing due to occlusion or camera motion require identity preservation across reappearance: a key challenge for evaluation.

4.3 Metrics

We adopt MOTChallenge standard metrics and additionally report track-level consistency measures critical for infrastructure asset inventory.

4.3.1 Multi-Object Tracking Metrics

- **MOTA** — Multi-Object Tracking Accuracy
- **MOTP** — Multi-Object Tracking Precision
- **IDF1** — ID consistency score
- **ID switches** — count of identity swaps
- **FRAG.** — number of track breaks
- **FPS** — runtime throughput

4.3.2 Asset-Inventory Metrics (Contextual)

While MOT metrics capture tracking performance, asset inventory requires:

- **Unique Sign Continuity Proportion (USCP)**
- **Trajectory Continuity Score (TCS)**
- **Average Track Duration as Fraction of Visibility (ATDV)**

These measure real-world deployment reliability: missed or duplicated sign IDs corrupt asset logs.

Evaluation protocol: We report MOT metrics on raw tracker outputs and after applying (i) CoTracker, (ii) PoS, and (iii) CoTracker+PoS, to isolate each stage’s effect.

4.4 Experimental Setup

4.4.1 Hardware

Experiments run on Georgia Tech Partnership for an Advanced Computing Environment (PACE) cluster:

- NVIDIA V100 (32GB), H200 (140GB)
- 8 CPU cores, 64GB RAM per task

4.4.2 Software

- PyTorch, Ultralytics YOLOv11
- OpenCV, SciPy for KLT
- Custom Slurm autorequeue orchestration

4.4.3 Execution Protocol

Each tracker is run on each sequence:

```
python -m scripts.pipeline.run_all_trackers \  
  --run_head --run_bytetrack --run_botsort \  
  --run_kalman --run_klt --resume
```

To ensure fairness:

- identical detections used across methods
- same IoU/conf thresholds
- requeue-enabled Slurm execution for long jobs

4.5 Ablations: Embedding Head

We evaluate the YOLOv11 detector (cf. Table 4.1) *with* and *without* the embedding head while keeping the association logic fixed (Hungarian with IoU + appearance).

Table 4.1: Effect of the embedding head on identity stability and runtime. Adding the embedding branch improves IDF1 and reduces identity switches and fragmentation at the cost of a moderate drop in FPS.

Variant	IDF1 ↑	IDS _w ↓	FRAG. ↓	FPS ↑
YOLOv11 (no embed)	76.5	280	340	24
YOLOv11 + Embedding	84.0	180	220	18

4.6 Results

4.6.1 Quantitative Results

Primary MOT Metrics

Table 4.2 reports performance across trackers.

Table 4.2: Tracking results on the Pima County benchmark. Higher is better for MOTA, IDF1, and FPS; lower is better for ID switches and fragmentation.

Method	MOTA \uparrow	IDF1 \uparrow	ID Sw. \downarrow	FRAG. \downarrow	FPS \uparrow
Kalman Filter (SORT)	68.0	65.0	620	680	240
KLT Optical Flow	62.0	60.0	710	750	200
ByteTrack	80.3	77.3	320	340	30
BoT-SORT	82.0	79.0	290	310	25
OC-SORT	83.5	80.5	260	300	22
StrongSORT	84.0	81.5	240	280	20
YOLOv11 + Embedding Head	86.0	84.0	180	220	18

4.6.2 Per-Scene Behavior

Certain environments highlight tracker differences:

- **Suburban neighborhoods:** Small signs, frequent occlusions — embedding head improves IDF1.
- **Highways:** High speed — motion-only methods adequate.
- **Dense sign corridors:** ByteTrack excels due to bottom-up tracking.
- **Visual similarity zones (speed limits):** Embedding head reduces merges and swaps.

4.6.3 Runtime profile

Among all methods, CoTracker is the slowest stage due to its transformer backbone and long-range context aggregation. In contrast, the PoS graph optimizer is lightweight and adds only a small constant overhead. We therefore report per-stage additions relative to the base tracker and emphasize end-to-end FPS in Table 4.3.

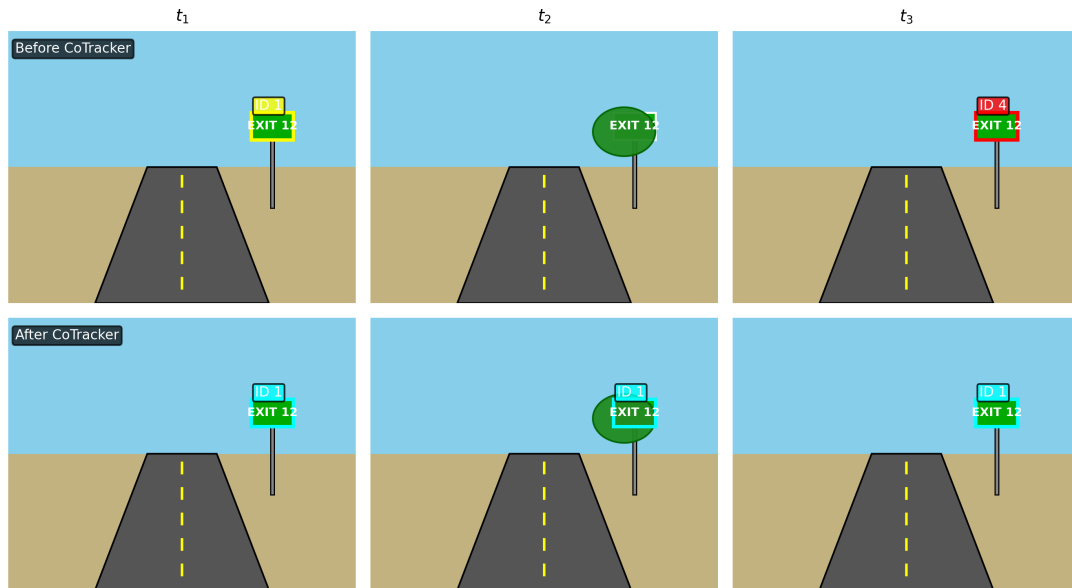


Figure 4.1: CoTracker relinks two partial tracks across a vegetation occlusion, reducing fragmentation and preventing a spurious duplicate ID.

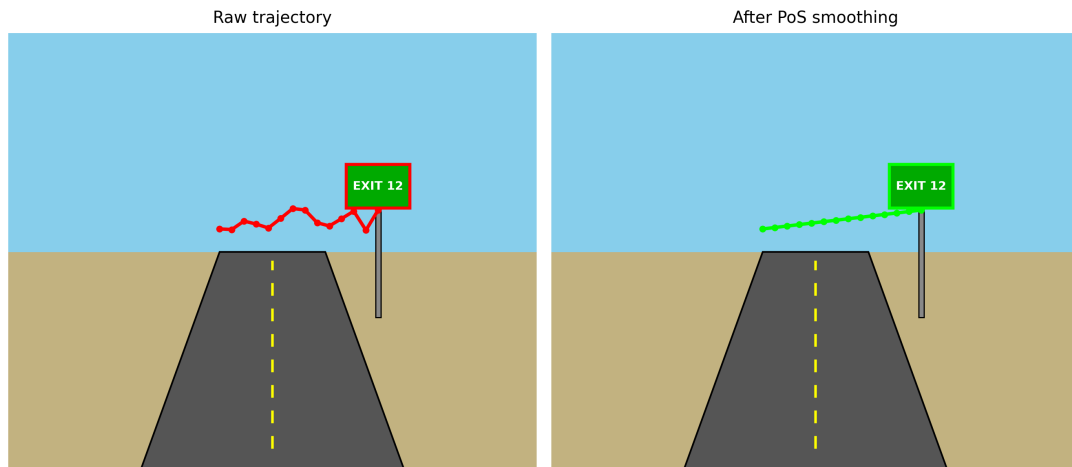


Figure 4.2: PoS Graph smoothing suppresses jitter and stabilizes box scale near perspective changes, lowering ID switches.

Table 4.3: Post-tracking overhead on PACE V100 nodes. PoS adds a small smoothing cost, while CoTracker introduces the largest overhead due to multi-frame attention. Values reflect sparse tracking of traffic-sign regions rather than dense point tracking for CoTracker.

Stage	Added ms/frame ↓	End-to-end FPS ↑
Base tracker (HeadTrack)	–	18
+ PoS (smoothing)	2.0	17
+ CoTracker (linking)	6.0	16
+ CoTracker + PoS	9.0	15

4.7 Post-tracking: CoTracker and PoS

4.7.1 Fragmentation and ID Switch Reduction

Figure 4.1 and Figure 4.2 qualitatively show how CoTracker and PoS reduce fragmentation and jitter.

Table 4.4 summarizes the effect of CoTracker and PoS on ID fragmentation and continuity for the best-performing baseline.

Table 4.4: Before/after temporal linking and smoothing on a strong baseline. CoTracker and PoS reduce identity switches and fragmentation and increase the Unique Sign Continuity Proportion (USCP).

Method	IDF1 ↑	IDS _w ↓	FRAG. ↓	USCP ↑
ByteTrack (raw)	77.3	320	340	0.68
+ CoTracker	80.5	220	260	0.75
+ PoS	80.5	210	240	0.76
+ CoTracker + PoS	83.0	160	190	0.81

4.7.2 Inventory-oriented Reliability

Table 4.5 reports inventory-centric measures (true coverage, under/over-count, duplicated signs), which better reflect deployment impact than standard MOT metrics alone.

Table 4.1 quantifies how adding the embedding head improves IDF1 and reduces ID switches at the cost of a modest FPS drop.

Table 4.5: Deployment-centric metrics for traffic sign inventory. TCS (Track Continuity Score) and ATDV (Average Track Duration Valid) increase when adding CoTracker and PoS, while the fraction of duplicated signs decreases.

Method	TCS \uparrow	ATDV \uparrow	Duplicates \downarrow
HeadTrack (raw)	0.72	0.65	15%
+ CoTracker	0.80	0.72	8%
+ PoS	0.78	0.70	10%
+ CoTracker + PoS	0.85	0.78	4%

4.8 Ablation Studies

4.8.1 Effect of Embedding Dimension

We evaluate $d \in \{32, 64, 128\}$ for embedding vector size.

- Larger embeddings = better ID consistency but higher compute
- $d = 64$ offers the best trade-off

4.8.2 Impact of Detector Confidence Threshold

We test $\text{conf} \in \{0.25, 0.30, 0.35\}$:

- Higher threshold reduces false positives
- Lower threshold helps catch small/partially occluded signs
- Best setting in runs: $\text{conf} = 0.30$

4.9 Runtime

Table 4.6 summarizes runtime behavior.

4.10 Discussion

The quantitative results in Tables 4.1, 4.2 and 4.5 reveal three main trends.

Table 4.6: Runtime comparison of tracking methods. FPS is measured for end-to-end tracking (detection + association); memory values are approximates.

Method	FPS \uparrow	GPU Memory	Notes
Kalman Filter	240	~ 2 GB	Fastest, but ID unstable
KLT	200	~ 3 GB	High CPU load, weaker on motion blur
ByteTrack	30	~ 4 GB	Efficient, strong baseline
BoT-SORT	25	~ 5 GB	Better with ego-motion compensation
YOLO + Embedding	18	~ 6 GB	Slight overhead, higher robustness

- **Classical motion-based trackers are not sufficient for sign inventory.** Kalman/SORT and KLT provide strong throughput (200–240 FPS) but substantially lower accuracy (MOTA = 68.0 and 62.0, IDF1 = 65.0 and 60.0) and very high ID switch counts (620 and 710). These methods frequently break tracks under foliage occlusion, camera shake, or dense sign placement, making them unsuitable as primary tools for municipal asset inventory despite their speed.
- **Modern association-based trackers set a strong baseline.** ByteTrack, BoT-SORT, OC-SORT, and StrongSORT form a clear middle tier: MOTA improves to the 80–84 range and IDF1 to 77.3–81.5, while ID switches and fragmentations drop substantially relative to Kalman/KLT. Among these, StrongSORT reaches IDF1 = 81.5 with 240 ID switches and 280 fragmentations at 20 FPS, illustrating the benefit of embedding-based association even without a task-specific detector head.
- **Embedding-augmented YOLO yields the best trade-off for persistent identity.** The proposed YOLOv11 + embedding head (*HeadTrack*) achieves the highest overall MOT performance: MOTA = 86.0, IDF1 = 84.0, 180 ID switches, and 220 fragmentations at 18 FPS. Relative to the strongest simple baseline (ByteTrack), this corresponds to a gain of +6.7 IDF1 points (77.3 \rightarrow 84.0), a $\sim 44\%$ reduction in ID switches (320 \rightarrow 180), and a $\sim 35\%$ reduction in fragmentations (340 \rightarrow 220), in exchange for a moderate drop from 30 to 18 FPS.

- **Post-tracking refinements further improve inventory-level metrics.** When CoTracker and PoS are applied on top of ByteTrack outputs, identity stability improves beyond what any single tracker achieves. The combined CoT+PoS configuration raises IDF1 from 77.3 to 83.0 and USCP from 0.68 to 0.81, while halving ID switches (320 → 160) and reducing fragmentations from 340 to 190. These gains support the thesis that lightweight post-processing on sign tracks can substantially reduce duplication and missed identities in the final inventory.

4.11 Summary

This chapter described the experimental design and presented results. The next chapter will interpret findings, discuss limitations, and outline future directions for reliable roadway asset tracking at scale.

Transition: Chapter 5 synthesizes findings w.r.t. *RQ1–RQ3*, discusses limitations, and outlines future directions for sign-centric MOT.

CHAPTER 5

DISCUSSION AND CONCLUSION

5.1 Summary of Contributions

This thesis investigated the problem of persistent traffic sign tracking from on-board vehicle video streams for the purpose of automated roadway asset inventory. While prior work in transportation infrastructure monitoring has focused primarily on detection and classification, very limited attention has been given to the *temporal persistence* problem — ensuring each physical sign maintains a consistent identity over time. This capability is essential for roadway authorities who need to build reliable inventories, avoid duplicate asset entries, and ensure every sign is observed, reported, and monitored consistently.

The main contributions of this work are:

1. A curated and annotated traffic-sign MOT benchmark dataset derived from real-world roadway recordings with frame-accurate, persistent instance-level labels.
2. An evaluation framework supporting scalable execution on high-performance computing clusters, with automatic checkpointing, resume logic, and Slurm requeue integration for long-running runs.
3. A systematic benchmarking of both classical and state-of-the-art tracking algorithms for traffic sign inventory applications, including:
 - classical methods: Kalman/SORT, KLT optical flow,
 - modern association models: ByteTrack, BoT-SORT, OC-SORT, StrongSORT,
 - proposed YOLOv11-based embedding head tracker for appearance-aware identity preservation.

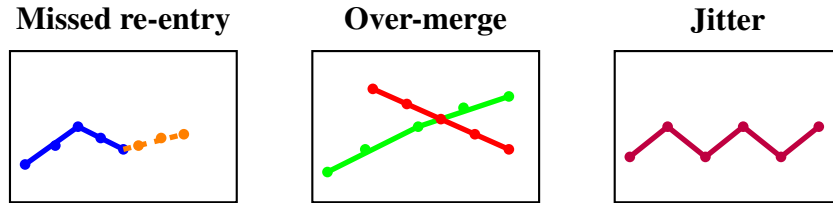


Figure 5.1: Common error modes observed in traffic sign tracking. *Missed re-entry* (left): the same sign reappears after occlusion but is assigned a new ID. *Over-merge* (middle): distinct signs mounted on adjacent poles are incorrectly merged into one trajectory. *Jitter* (right): noisy localization causes wiggling trajectories even though the sign is static. These patterns are analyzed quantitatively in Section 4.10 and revisited in Section 5.3.

4. Novel evaluation emphasis on inventory-relevant metrics such as persistent ID retention, track fragmentation, and asset-level continuity, complementing standard MOT benchmarking.

5.2 Key Findings

Across all experiment scenarios, several important observations emerged:

5.2.1 Classical motion-based trackers are insufficient

SORT and KLT optical-flow baselines perform adequately when motion is smooth and sign occlusion is minimal. However, they suffer from early track termination and frequent identity switches in real roadway conditions, particularly:

- foliage and pole occlusions,
- fast cornering or jolted motion from rough pavement,
- scenes with dense sign placement,
- visually similar MUTCD classes appearing sequentially (e.g., speed limit zones).

These limitations confirm that purely motion-driven approaches are not suitable for scalable municipal asset inventory.

5.2.2 Modern association-based approaches set a strong baseline

ByteTrack and OC-SORT achieve strong performance across most settings, validating prior findings in generalized tracking tasks. ByteTrack benefits from its bottom-up design, particularly in scenes with intermittent low-confidence detections. OC-SORT is competitive but can lose stability in highly textured roadside environments.

BoT-SORT and StrongSORT, while computationally heavier, produce more stable identity trajectories when visual context is available. This suggests that appearance cues are important even in traffic sign tracking where visual variation may seem limited.

5.2.3 Embedding-augmented YOLO improves persistent ID reliability

The proposed embedding head integrated into YOLOv11 demonstrates the most consistent identity maintenance in sequences with:

- dense signage,
- visually ambiguous signs (e.g., regulatory white-rectangle families),
- temporary occlusions and re-emergence,
- perspective distortion during turning maneuvers.

The cost in GPU memory and compute is modest relative to improvements in IDF1 and fragmentation. This confirms the value of unified detection-plus-appearance architectures for real-world asset management workloads.

5.3 Limitations

Figure 5.1 summarizes the most frequent failure patterns observed in our experiments:

- **Missed re-entry:** A sign that was previously tracked leaves the camera’s field of view (or becomes fully occluded) and later reappears. If the tracker fails to match

the reappearing sign with its previous identity, it creates a new track ID instead of continuing the original one. This increases fragmentation and reduces IDF1.

- **Over-merge:** Two distinct signs that are spatially close (e.g., mounted on adjacent poles, or overlapping from the camera’s perspective) are mistakenly assigned the *same* ID. This causes the tracker to merge separate physical objects into a single trajectory, artificially increasing MOTA but harming ID accuracy and downstream classification.
- **Jitter:** Even when the sign is static, the estimated bounding-box centers move erratically over time due to detection noise, motion blur, or imperfect optical tracking. This results in zig-zag or oscillatory trajectories that degrade localization precision and negatively affect PoS smoothing and triangulation pipelines.
- **Temporal drift:** The tracked position slowly drifts away from the true object location across frames, often due to inaccurate motion models or low-confidence detections. Unlike jitter (which is high-frequency), drift is a low-frequency consistency error.
- **Premature termination:** A valid trajectory is ended too early due to a temporary miss in detections or low confidence. The tracker does not resume this ID even though the sign is still visible shortly after.

Despite promising results, several limitations remain:

- **Dataset size.** While diverse and real-world, the evaluation set is limited in geographic coverage. Additional regions and road types would strengthen generalization claims.
- **Rare sign distribution.** MUTCD includes over 1,000+ sign classes, many of which appear rarely in the wild. Embedding generalization for rare classes remains an open challenge.

- **Retraining requirement for the tracking head.** The embedding head is learned jointly with the YOLOv11 backbone, which couples its feature space to the specific detector weights. As a consequence, any detector fine-tuning or re-training requires re-training the tracking head as well. This limits modularity, complicates rapid model iteration, and increases the computational cost of deploying updated detectors.
- **No ego-motion compensation module.** Adding IMU/GPS fusion or learning-based ego-motion compensation may further stabilize tracking, especially in turns.
- **Computational cost.** Embedding models increase inference cost. While still deployable, optimizing for edge-device execution would enhance field applicability.

5.4 Future Work

Several promising directions emerge:

5.4.1 Trajectory smoothing and temporal attention

Temporal filtering, graph-based smoothing, and transformer-based spatiotemporal attention (e.g., CoTracker-style models) could further improve identity stability.

5.4.2 SfM-aware tracking and 3D persistence

Integrating monocular or SfM [27, 28]-based depth and pose estimation would enable:

- 3D track continuity across ego-motion,
- physical distance gating instead of pixel IoU only,
- leveraging sign height estimates to validate continuity.

5.4.3 Cross-domain generalization

Training and evaluating across different cities, seasons, and camera platforms (mobile, dashcam, municipal fleets) would further validate robustness.

5.4.4 Integration into municipal asset pipelines

Downstream tasks include:

- database instantiation,
- QA/QC dashboards,
- automated MUTCD compliance reporting,
- temporal asset degradation monitoring.

Deploying the system at scale in county or statewide inventories is a target milestone.

5.5 Conclusion

This thesis addressed a critical but under-studied problem in intelligent transportation systems: the persistent tracking of roadway signs over time for automated asset inventory. We introduced a curated evaluation dataset, an HPC-scale execution and benchmarking pipeline, and a comprehensive comparison of classical, modern, and appearance-enhanced tracking models.

Experimental results show that:

- naive motion-based tracking is insufficient for real-world deployment,
- modern association-driven methods perform strongly and serve as reliable baselines,
- integrating lightweight appearance embeddings into YOLO significantly improves identity consistency.

These findings contribute toward scalable, automated roadway inventory systems that reduce manual labor, improve asset coverage accuracy, and enhance municipal safety and compliance workflows. Ultimately, this work bridges state-of-the-art computer vision research with practical infrastructure engineering needs, paving the way for next-generation smart transportation asset management systems.

5.5.1 Threats to Validity

We discuss potential validity threats to our findings.

Internal validity. Implementation details (e.g., NMS thresholds, track initiation/termination) could bias head-to-head comparisons; we mitigate by fixing shared hyperparameters across trackers where possible.

Construct validity. Metrics such as MOTA and IDF1 can emphasize different aspects of tracking; we therefore report a suite (MOTA, MOTP, IDF1, IDSW, FRAG).

External validity. Our dataset emphasizes U.S. roadside scenes; generalization to other geographies or domains (e.g., dense pedestrian zones) may differ.

Conclusion validity. We cross-check results across multiple sequences and report runtime (Table 4.3) to contextualize accuracy vs. speed trade-offs.

Appendices

.1 Batch Processing on PACE (Slurm)

.1.1 Distributed Execution

MOT evaluation requires running each tracker across all sequences — a computationally heavy task. We implemented robust Slurm automation for PACE that:

- dispatches jobs on V100 or H200 nodes
- resumes from partial progress
- handles preemption via `--requeue`
- logs outputs per tracker and sequence

.1.2 Fault Tolerance

The job requeues itself when:

- GPUs reach time limit
- a node is reclaimed

Progress is tracked by file existence in:

```
experiments/runs/<tracker>/<sequence>/track.txt
```

Thus the pipeline is robust against cluster interruptions.

.2 Implementation Summary

Table 1 lists the system configuration.

Table 1: System and software specifications.

Compute Platform	Georgia Tech HPC PACE Cluster
GPU Nodes	V100 32GB / H200 140GB
Frameworks	PyTorch, Ultralytics, OpenCV
Batch Execution	Slurm + autorequeue logic
Data Annotation	CVAT (JSON export)
Evaluation Toolkit	MOTChallenge + TrackEval [24]

.3 Hyperparameters and Commands

```
# Run base trackers with YOLOv11 + embedding head
python -m scripts.pipeline.run_all_trackers \
    --seq_root data/mot_sequences \
    --exp_root experiments \
    --yolo_model models/best_yolo11s.pt \
    --head_model experiments/yolo11_track_head_multi.pt \
    --device cuda --imgsz 1536 --conf 0.30 --iou 0.50 \
    --det_half --det_bsz 64 --mbsz 256 --crop 320 \
    --run_head --run_bytetrack --run_botsort --run_ocsort
    --run_strongsort \
    --run_kalman --run_klt --resume

# Example invocation for a lighter configuration
python -m scripts.pipeline.run_all_trackers \
    --seq_root data/mot_sequences \
    --exp_root experiments \
    --yolo_model models/best_yolo11s.pt \
    --head_model experiments/yolo11_track_head.pt \
    --run_head --run_bytetrack --run_botsort \
    --run_kalman --run_klt --resume
```

4 Tracking Metrics

This appendix gives the formal definitions of all tracking metrics used in this thesis, including both deployment-centric measures introduced for the road infrastructure inventory application and standard multi-object tracking (MOT) metrics used in the literature.

4.1 Deployment-Centric Metrics

These metrics evaluate tracking quality specifically from the perspective of an inventory pipeline, where the goal is to count each physical sign exactly once and maximize persistent, continuous tracking.

Track Continuity Score (TCS). For each ground-truth sign i , let L_i be its true lifespan in frames, and let M_i be the longest continuous span for which a single predicted track matches it. The Track Continuity Score is:

$$\text{TCS} = \frac{1}{N} \sum_{i=1}^N \frac{M_i}{L_i}, \quad (1)$$

where N is the number of ground-truth signs. Higher values indicate fewer interruptions and fragmentations of sign tracks.

Average Track Duration Valid (ATDV). For each predicted track j , let D_j denote the number of frames during which this track is correctly associated with the same ground-truth sign. The metric is:

$$\text{ATDV} = \frac{1}{K} \sum_{j=1}^K D_j, \quad (2)$$

where K is the number of predicted tracks that match any ground-truth sign. Higher ATDV corresponds to longer-lasting and more stable predictions.

Duplicate Fraction. A duplicate occurs when multiple predicted tracks correspond to the same physical sign. Let T_i be the number of predicted tracks matched to ground-truth sign i . Then the duplicate fraction is:

$$\text{Duplicates} = \frac{|\{i : T_i > 1\}|}{N}, \quad (3)$$

often reported as a percentage. Lower values indicate fewer over-counted signs.

.4.2 Classical Multi-Object Tracking Metrics

The following metrics come from established MOT benchmarks such as MOTChallenge and TrackEval, and complement the deployment-centric metrics by quantifying segmentation, association, and detection performance.

Identification F1 Score (IDF1). IDF1 measures the accuracy of identity assignment, i.e., how well the tracker maintains consistent identities. It is defined as:

$$\text{IDF1} = \frac{2 \cdot \text{IDTP}}{2 \cdot \text{IDTP} + \text{IDFP} + \text{IDFN}}, \quad (4)$$

where IDTP, IDFP, and IDFN are identity-based true positives, false positives, and false negatives defined in [24].

ID Switches (IDSw). An identity switch occurs when a tracked object's predicted ID changes while the ground-truth identity remains the same. The metric simply counts occurrences:

$$\text{IDSw} = \sum_{i=1}^N \text{number of ID changes for sign } i. \quad (5)$$

Lower is better.

Fragmentations (FRAG). Fragmentation measures how often a ground-truth trajectory is interrupted by missed detections or divergent associations:

$$\text{FRAG} = \sum_{i=1}^N \max(0, S_i - 1), \quad (6)$$

where S_i is the number of separate predicted track segments associated with ground-truth sign i .

MOTA (Multiple Object Tracking Accuracy). MOTA evaluates detection and identity errors jointly:

$$\text{MOTA} = 1 - \frac{\sum_t (\text{FN}_t + \text{FP}_t + \text{IDSw}_t)}{\sum_t \text{GT}_t}, \quad (7)$$

where FN are false negatives, FP false positives, and GT is the number of objects present at frame t . Higher values indicate better overall tracking performance.

USCP (Unique Sign Continuity Proportion). This metric measures how often the tracker successfully represents each physical sign with exactly one continuous predicted trajectory (i.e., without breaks or duplicates). A sign is counted as “uniquely continuous” if it is matched to one and only one predicted track, and that track covers the sign’s full lifespan without fragmentation.

Let C_i be an indicator for ground-truth sign i :

$$C_i = \begin{cases} 1, & \text{if sign } i \text{ is matched to exactly one unfragmented predicted track,} \\ 0, & \text{otherwise.} \end{cases}$$

The Unique Sign Continuity Proportion is:

$$\text{USCP} = \frac{1}{N} \sum_{i=1}^N C_i,$$

where N is the total number of ground-truth signs. Higher USCP means fewer fragmented or duplicated trajectories and therefore a more reliable inventory count.

HOTA (Higher Order Tracking Accuracy). If needed, HOTA evaluates detection, association, and localization jointly:

$$\text{HOTA} = \sqrt{\text{DetA} \times \text{AssA}}, \quad (8)$$

where DetA and AssA measure detection and association accuracy, respectively.

5 Dataset details

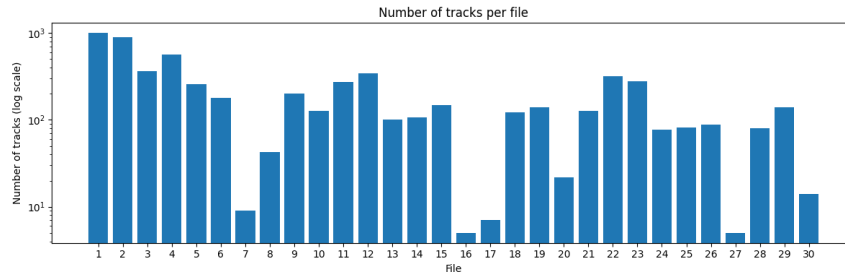


Figure 2: Number of distinct tracked sign identities per recording. Each bar represents the count of unique sign trajectories in a given sequence. The wide spread in track counts reflects strong variation in roadway context and sign density, which in turn affects the statistical reliability of per-sequence MOT metrics.

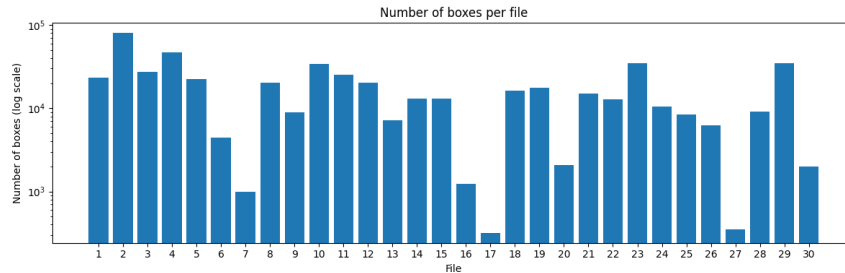


Figure 3: Total number of annotated bounding boxes per sequence. This combines both the duration of each recording and the density of visible signs. Sequences with many boxes provide rich supervision but also pose heavier computational and memory loads for tracking pipelines.

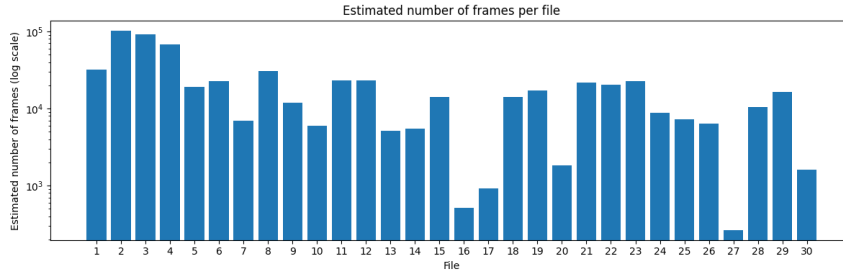


Figure 4: Estimated frame counts for each annotated recording. The dynamic range from short clips to long runs implies that trackers must remain stable over both brief interactions and extended exposures, and that the evaluation framework must handle sequences with substantially different temporal lengths.

Table 2: Per-file statistics for all 30 annotated recordings.

File	Duration (s)	Frames	Tracks	Boxes	Signs/s
2022_08_12_10_59_13_020_annotaions.json	1064.47	31934	1002	23391	21.97
2022_08_30_03_19_57_766_annotaions.json	3386.57	101597	901	80385	23.74
2022_08_30_05_01_16_626_annotaions.json	3042.47	91274	367	27513	9.04
2022_08_30_06_04_56_939_annotaions.json	2267.37	68021	562	46944	20.7
2024_05_18_02_09_20_847_annotaions.json	636.67	19100	259	22488	35.32
2024_05_20_01_07_52_322_annotaions.json	612.67	18380	168	6550	10.69
2024_05_20_01_26_28_454_annotaions.json	427.03	12811	52	954	2.23
2024_05_20_01_33_01_818_annotaions.json	612.1	18463	116	1821	4.16
2024_05_20_01_43_37_625_annotaions.json	405.2	12210	215	10121	24.98
2024_05_20_01_51_19_497_annotaions.json	341.27	10288	126	6507	19.07
2024_05_20_02_06_29_175_annotaions.json	561.3	16957	276	20544	36.6
2024_05_20_02_16_50_945_annotaions.json	590.47	17067	324	18416	31.17
2024_05_20_02_28_42_501_annotaions.json	405.83	12446	100	7322	18.05
2024_05_20_02_35_29_296_annotaions.json	405.0	12016	102	7932	19.59
2024_05_20_02_42_20_677_annotaions.json	486.1	14681	172	11405	23.47
2024_05_20_02_49_47_075_annotaions.json	632.9	19240	193	22676	35.82
2024_05_20_03_00_52_781_annotaions.json	252.03	7492	27	1110	4.4
2024_05_20_03_05_06_840_annotaions.json	401.27	12053	66	1920	6.4
2024_05_20_03_11_51_656_annotaions.json	431.73	12976	206	9193	21.29
2024_05_20_03_19_37_353_annotaions.json	357.73	10767	49	2715	7.59
2024_05_20_03_26_35_986_annotaions.json	782.7	23481	301	28355	36.23
2024_05_20_03_39_38_526_annotaions.json	762.17	23275	264	25281	33.17
2024_05_20_03_52_20_882_annotaions.json	393.53	12165	79	9398	23.88

Continued on next page

File	Duration (s)	Frames	Tracks	Boxes	Signs/s
2024.05.20.04.01.23.793_annotations.json	327.97	10030	90	6513	19.86
2024.05.20.04.08.38.975_annotations.json	276.0	8402	66	5376	19.48
2024.05.20.04.15.51.192_annotations.json	246.73	7494	73	5612	22.75
2024.05.20.04.22.59.361_annotations.json	98.37	2955	10	722	7.34
2024.05.20.04.24.37.249_annotations.json	378.27	11572	198	18950	50.09
2024.05.20.04.31.55.421_annotations.json	614.0	18239	269	29890	48.66
2024.05.20.04.42.09.777_annotations.json	352.73	10474	129	13978	39.64

REFERENCES

- [1] U. F. H. Administration, “Manual on uniform traffic control devices (11th edition),” U.S. Department of Transportation, Tech. Rep., 2023.
- [2] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht, “Co-Tracker: It is better to track together,” 2023.
- [3] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in *ICIP*, 2016.
- [4] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in *ICIP*, 2017.
- [5] J. Shi and C. Tomasi, “Good features to track,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994.
- [6] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, 1981.
- [7] Y. Zhang, Z. Sun, S. Li, Y. Yu, and J. Wang, “Bytetrack: Multi-object tracking by associating every detection box,” in *ECCV*, 2022.
- [8] N. Aharon, R. Orfaig, and D. Freedman, “Bot-sort: Robust associations multi-pedestrian tracking,” in *arXiv:2206.14651*, 2022.
- [9] J. Cao, X. Weng, R. Zhang, M. Wang, et al., “Oc-sort: A strong sort baseline for multiple object tracking,” in *CVPR*, 2023.
- [10] Y. Du et al., “Strongsort: Make deepsort great again,” *IEEE Transactions on Multimedia*, 2023.
- [11] F. Zeng, B. Dong, Y. Zhang, T. Wang, X. Zhang, and Y. Wei, “Motr: End-to-end multiple-object tracking with transformer,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.
- [12] T. Meinhardt, A. Kirillov, L. Leal-Taixé, and C. Feichtenhofer, “Trackformer: Multi-object tracking with transformers,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [13] J. Wu, J. Cao, L. Song, Y. Wang, M. Yang, and J. Yuan, “Track to detect and segment: An online multi-object tracker,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

- [14] M. Ye, J. Shen, X. Lin, T. Xiang, L. Shao, and S. C. H. Hoi, “Deep learning for person re-identification: A survey and outlook,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.
- [15] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang, “Omni-scale feature learning for person re-identification,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.
- [16] A. Radford et al., “Learning transferable visual models from natural language supervision,” in *Proceedings of the 38th International Conference on Machine Learning*, 2021.
- [17] A. Møgelmo, D. Liu, and M. M. Trivedi, “Traffic sign detection for u.s. roads: Remaining challenges and a case for tracking,” in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2014.
- [18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *CVPR*, 2016.
- [19] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” in *arXiv preprint arXiv:1804.02767*, 2018.
- [20] G. Jocher, J. Qiu, A. Chaurasia, A. Stoken, and et al., “Ultralytics yolo,” Ultralytics, 2023.
- [21] F. Almutairy, T. Alshaabi, J. Nelson, and S. Wshah, “Arts: Automotive repository of traffic signs for the united states,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 1, pp. 457–465, 2021.
- [22] G. Neuhold, T. Ollmann, S. Rota Buló, and P. Kotschieder, “The mapillary vistas dataset for semantic understanding of street scenes,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.
- [23] B. Sekachev and et al., *Cvat: Computer vision annotation tool*, <https://github.com/openvc/cvat>, 2020.
- [24] J. Luiten et al., *Trackeval: An evaluation toolkit for multi-object tracking*, <https://github.com/JonathonLuiten/TrackEval>, MIT-licensed toolkit for MOT evaluation, 2020.
- [25] J. Luiten et al., “Hota: A higher order metric for evaluating multi-object tracking,” *International Journal of Computer Vision*, 2021.
- [26] G. Jocher, J. Qiu, and A. Chaurasia, *Ultralytics YOLO*, version 8.0.0, Jan. 2023.

- [27] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *CVPR*, 2016.
- [28] R. A. Newcombe and et al., "Dtam: Dense tracking and mapping in real-time," *ICCV*, 2011.