Master in
Computer Vision
Barcelona

Project
Module 6
Coordination

Video Surveillance for Road Traffic Monitoring

Javier Ruiz / Ramon Morros

Week 4: Final Presentation

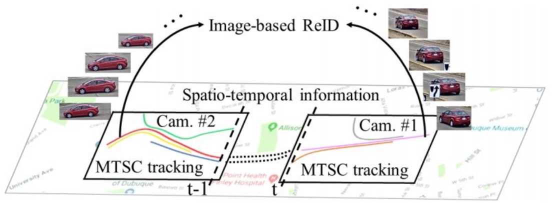
Andrea Sánchez
Daniel Pardo
Merlès Subirà
Francisco de Paula Urmeneta

Master in
Computer Vision
Barcelona

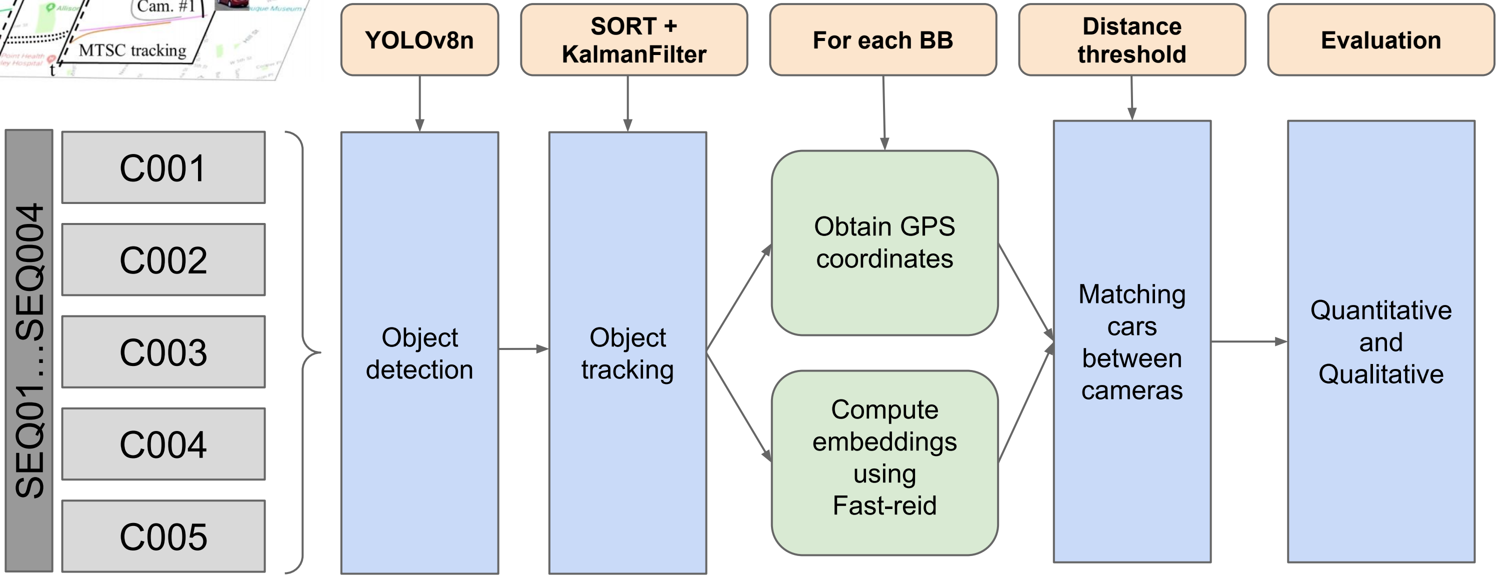
Index

1.a. Multi-camera tracking: Proposed pipeline

Dataset: CityFlow[1], S01, S03 and S04



Algorithm:

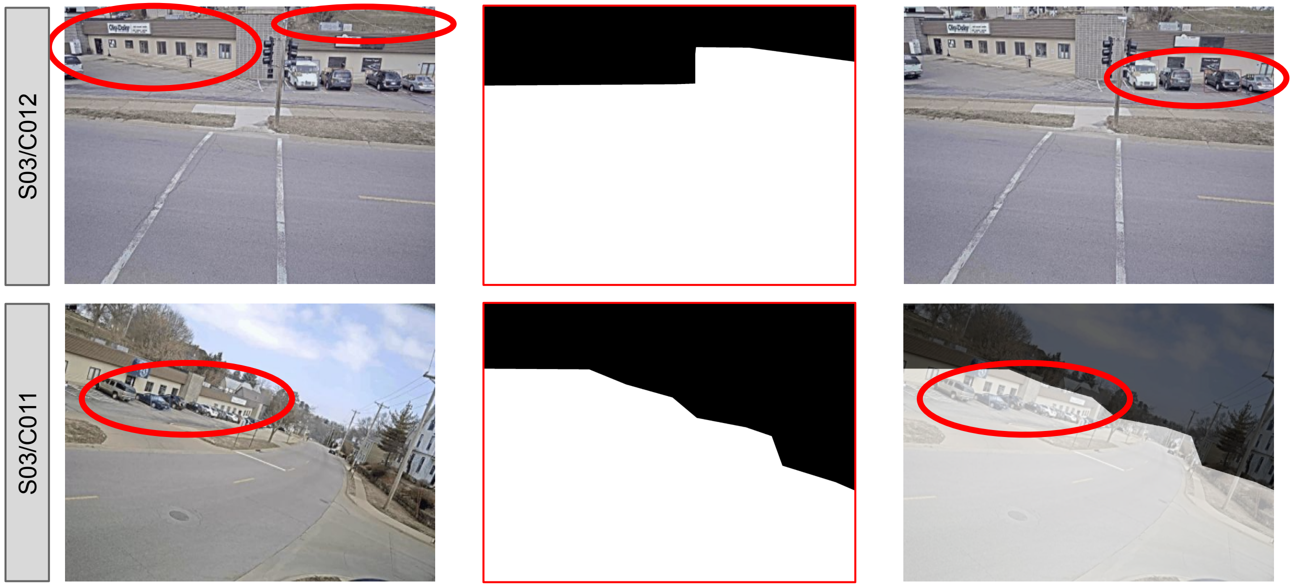


[1]Zheng Tang, et al., "CityFlow: A City-Scale Benchmark for Multi-Target Multi-Camera Vehicle Tracking and Re-Identification" CVPR 2019.

1.a. Multi-camera tracking: Object detection and tracking

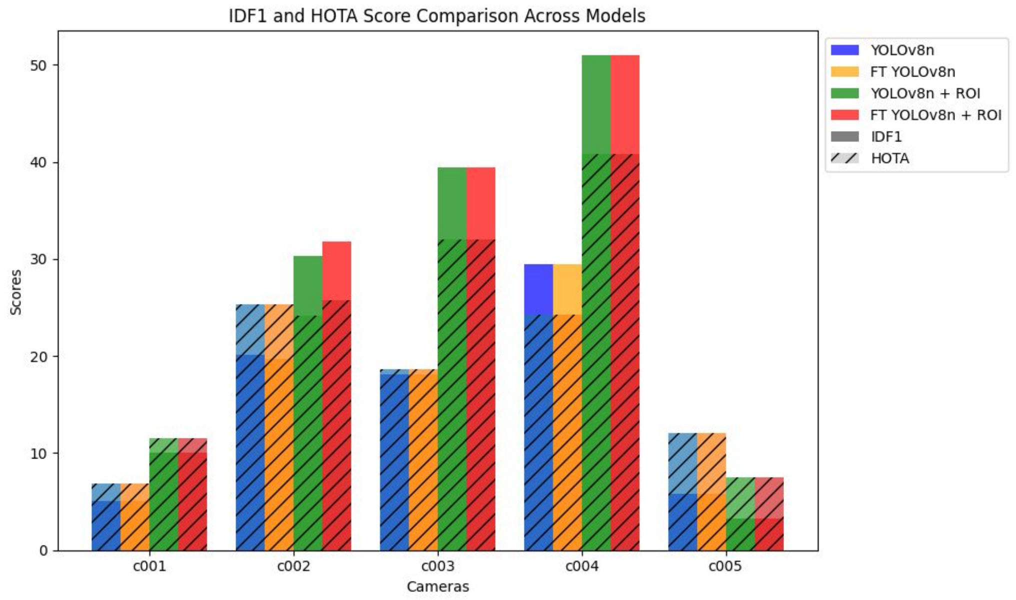
Improvement compared to recent weeks: use the region of interest (ROI) of the image provided in the dataset to filter detections.

1. Compute center of the bbox detection
2. Check whether it is inside or outside ROI mask
3. If it is outside, we eliminate the detection



1.a. Multi-camera tracking: Object detection and tracking

Testing quantitatively the effect of ROI mask:
no changes for the sequences studied, except for c002, where fine-tuned version improves the results with ROI mask, while original version worsens.

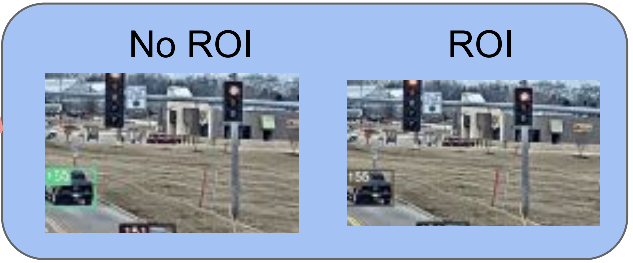


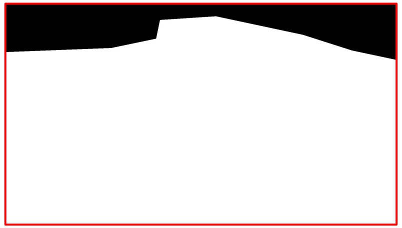
1.a. Multi-camera tracking: Object detection and tracking

Why is it an improvement then? We can filter out false positives in the YOLO detection, which prevents these errors from spreading to the tracking.

Fine-tuned version detects a greater number of cars, especially stationary ones, compared to the original pre-trained version.

	NO ROI	ROI
YOLOv8n		
YOLOv8n fine-tuned		



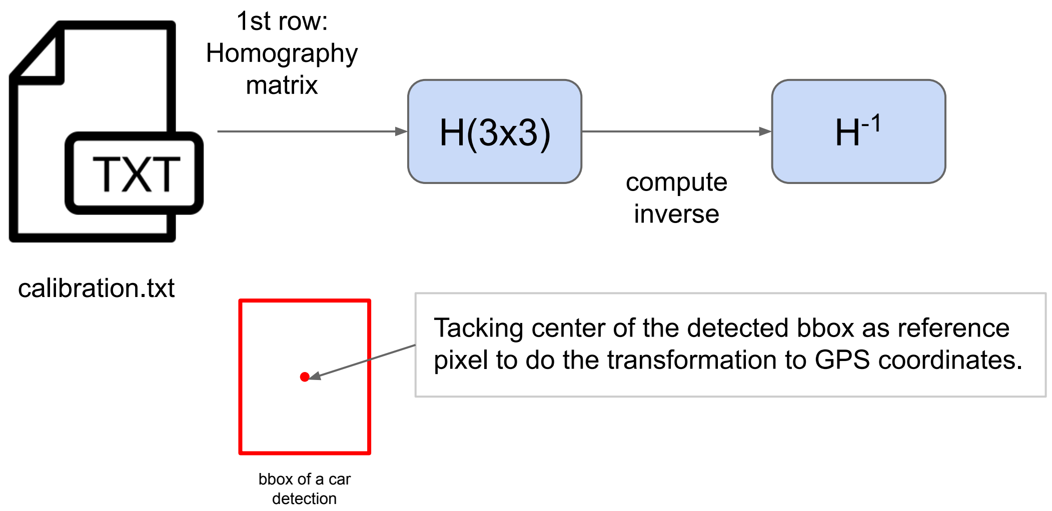

ROI c002

1.a. Multi-camera tracking: Obtaining GPS coordinates

Adjusting videos: we skipped first frames from the firsts videos so they are all aligned with the last (based on the timestamp information provided in the dataset).

Note: Video transmission noise causes skipped frames, leading to misalignment in videos. Most videos run at 10 FPS, except for c015 in S03, which runs at 8 FPS.

1.a. Multi-camera tracking: Obtaining GPS coordinates



Tacking center of the detected bbox as reference pixel to do the transformation to GPS coordinates.

Homography matrix: *projects GPS coordinates to 2D image coordinates (pixels)*

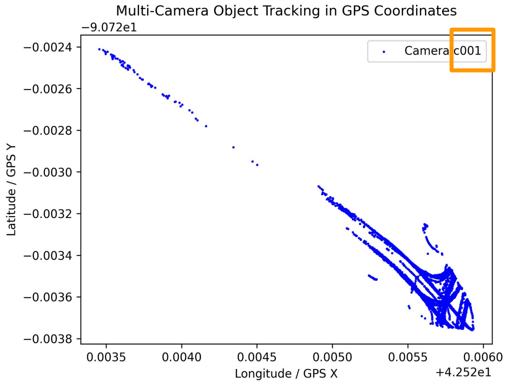
Inverse homography matrix: converts image coordinates to global GPS coordinates.

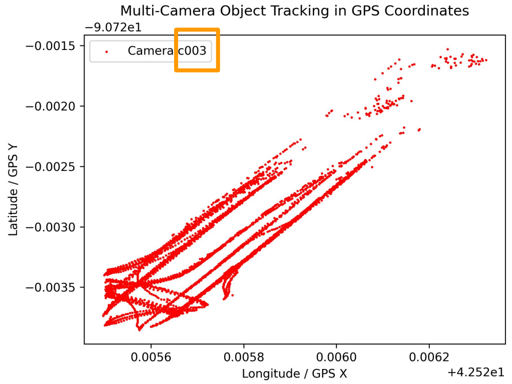
✓ **We have pixels from all cameras in the same reference space.**

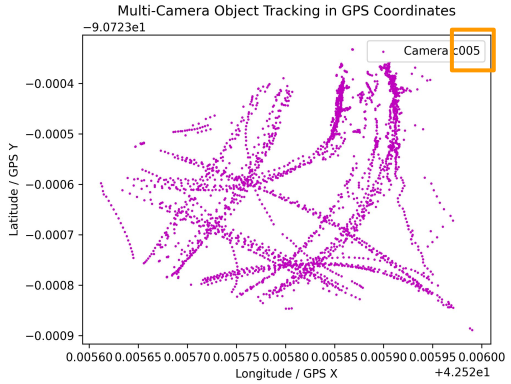
Problem: we lack the camera's calibration parameters and distortion coefficients. Without these, we cannot properly correct pixel distortion before applying H^{-1} to accurately convert pixels to GPS coordinates.

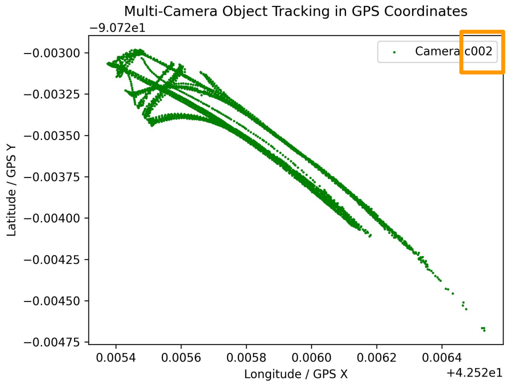
1.a. Multi-camera tracking: Obtaining GPS coordinates

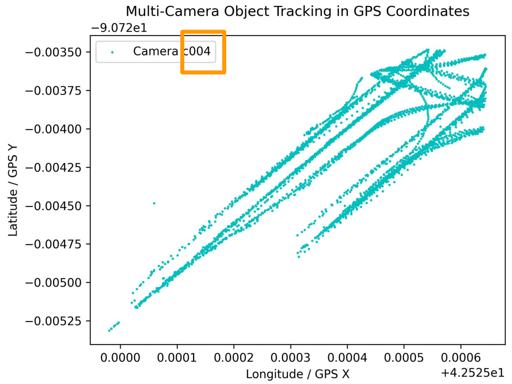
Here we show for every camera of S01 its tracking in global coordinates:

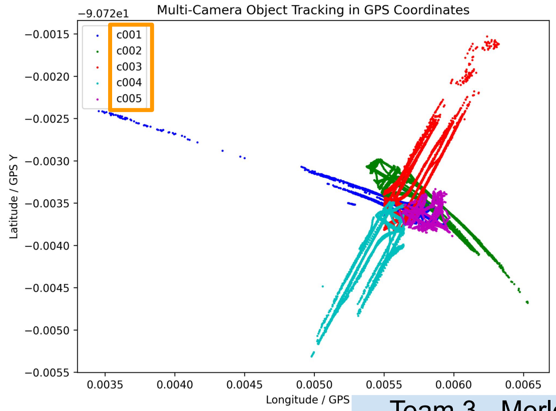








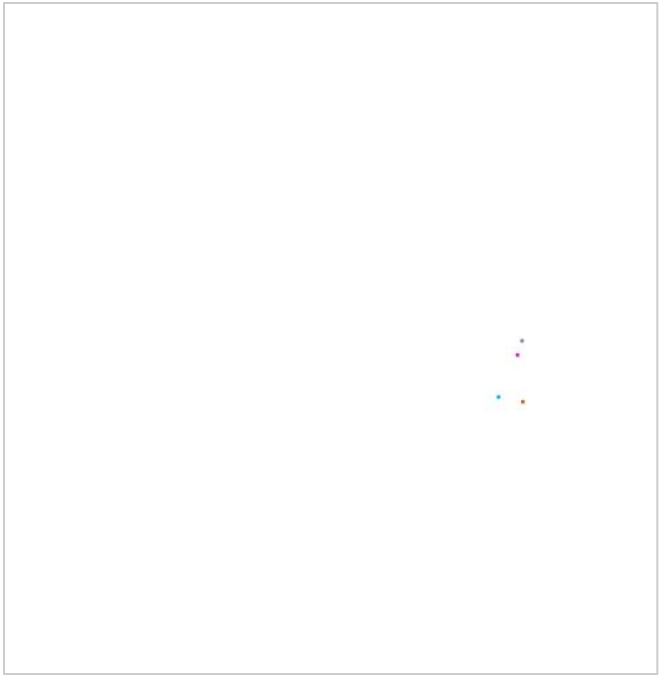




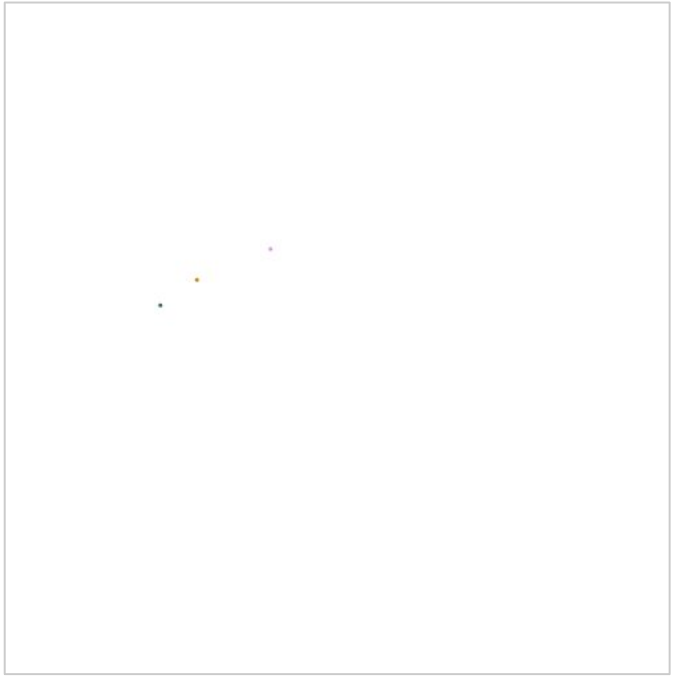
1.a. Multi-camera tracking: Obtaining GPS coordinates

Distorsion problem:

Watch tracking in 5 cameras:



Watch tracking in 2 of the cameras:



1.a. Multi-camera tracking: Computing embeddings

Embeddings are used for Re-Identification in a multi-camera tracking system.

- Framework: pre-trained **FastReID**[2].
- Feature extractor: ResNet-50 processes the cropped object within a bbox and outputs a 2048-dimensional embedding vector.

Pre-training dataset: **VeRi**

- large-scale vehicle Re-ID dataset
- 776 vehicle IDs across multiple camera views, captured under different lighting conditions, viewpoints, and occlusions.

Why pre-trained FastReID on VeRi-776?

- It has learned to capture distinct visual features for different vehicles, generating meaningful embeddings for different vehicles.
- Metric Learning: objects from the same class are closer in the feature space, while objects from different classes (different vehicles) are far apart.
- Compute distances between embeddings (cosine similarity) to track objects across multiple cameras or frames.

[2]FastReID: https://github.com/JDAI-CV/fast-reid/blob/master/MODEL_ZOO.md#veri-baseline

1.a. Multi-camera tracking: Matching between cameras

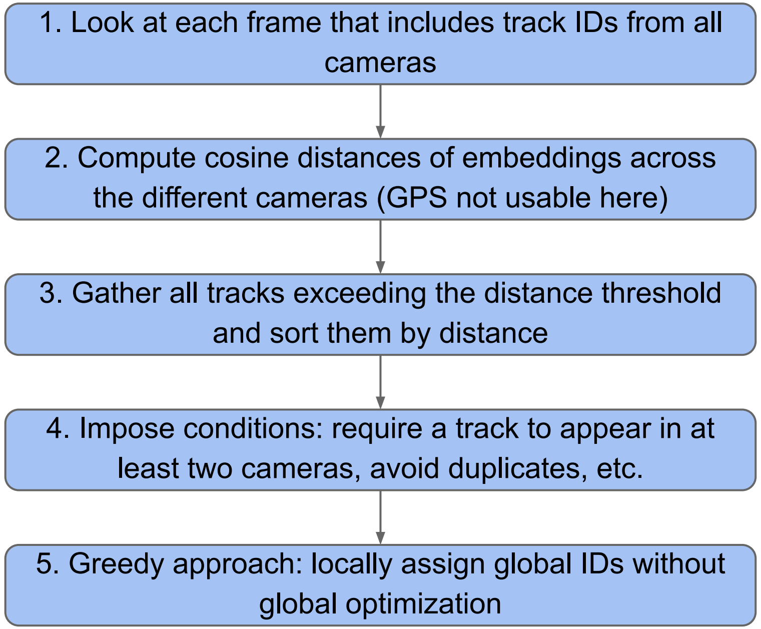
1. **Refining Intra-camera ID Switches (Using GPS + Embeddings)**

We leverage GPS information to refine ID switches that arise from SORT + Kalman tracking. If the cameras' GPS coordinates are available, we can combine this positional information with extracted embeddings to further adjust the tracking.

- For each frame, we consider a sliding window of the previous three frames and calculate the Haversine distance in GPS space between the current track ID and its previous positions, thus obtaining a distance in kilometers.
- We also compute the cosine distance of the embeddings.
- These two distances are then weighted, and if the combined metric exceeds a certain threshold (indicating an ID switch), we update the ID accordingly.

2. **Multi-camera Matching**

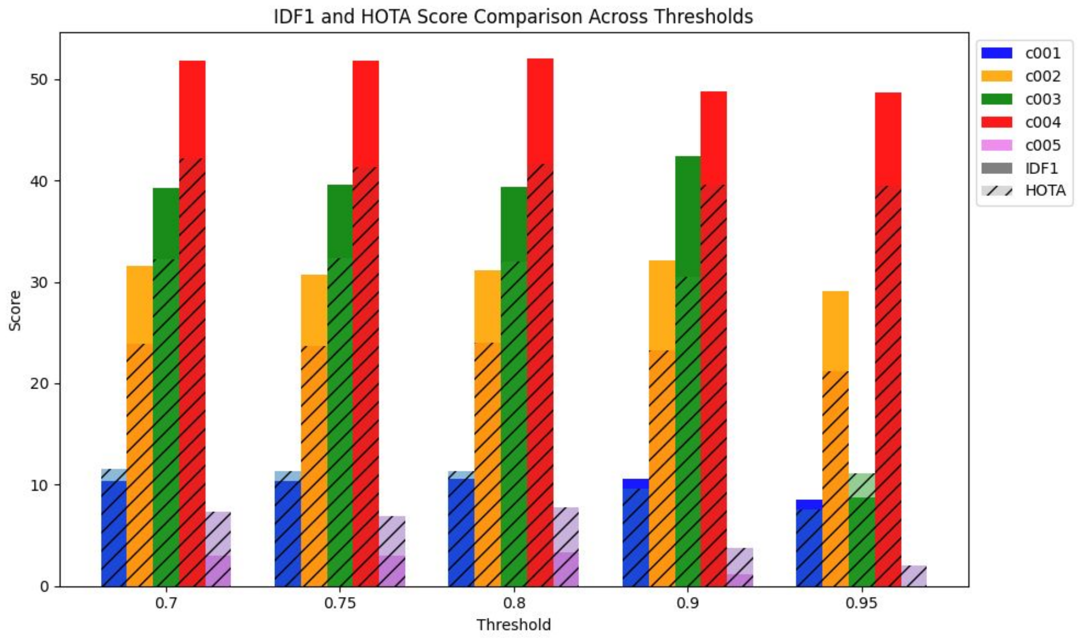
At this stage, the videos already have intra-camera tracking (from last week's Kalman + SORT). Now we need to handle inter-camera matching.



1. Look at each frame that includes track IDs from all cameras
2. Compute cosine distances of embeddings across the different cameras (GPS not usable here)
3. Gather all tracks exceeding the distance threshold and sort them by distance
4. Impose conditions: require a track to appear in at least two cameras, avoid duplicates, etc.
5. Greedy approach: locally assign global IDs without global optimization

1.b. Multi-camera tracking: Quantitative results

For SEQ1, we tested different cosine-similarity distance thresholds to determine the best matching configuration. A threshold of 0.8 provided a good trade-off, so we used it to evaluate the remaining sequences.



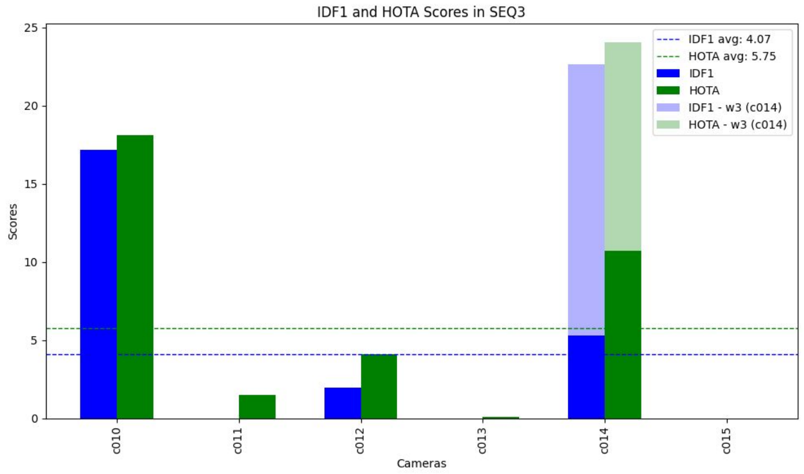
Metric	IDF1
SEQ 1	27.276
SEQ 3	6.7943
SEQ 4	26.7628
AVERAGE	20.2777

1.b. Multi-camera tracking: Quantitative results

Metrics are highly detector-dependent: evaluate both quality of detections and consistency of identities.

- Last week: with parked cars, detector found more objects, tracker maintained ID => evaluated as stable => increased IDF1 and HOTA.
- This week: removing parked cars => fewer detections => fewer identity associations; IDF1 and HOTA decrease.

Although tracking is better (because we are not tracking irrelevant items), the metrics interpret this as a loss of information. Therefore, if it is not taken into account, results may be overestimated.



1.b. Multi-camera tracking: Quantitative results

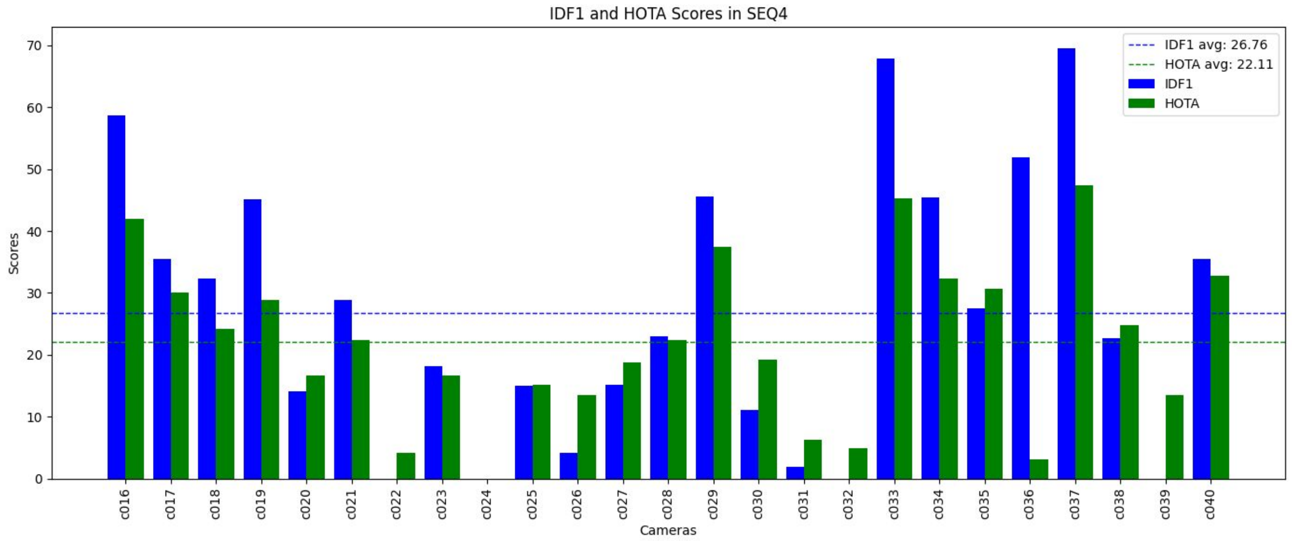
SEQ4 Challenges

Timing & Synchronization:
Large timestamp offsets and unsynchronized cameras require manual adjustment.

Scene Overlap:
Cameras capture different scenes with only partial trajectory intersections.

SEQ4 Potential Solution

Multi-Camera View:
Combining all 25 cameras is impractical; instead, focus on pairing cameras that share the same scene for matching.

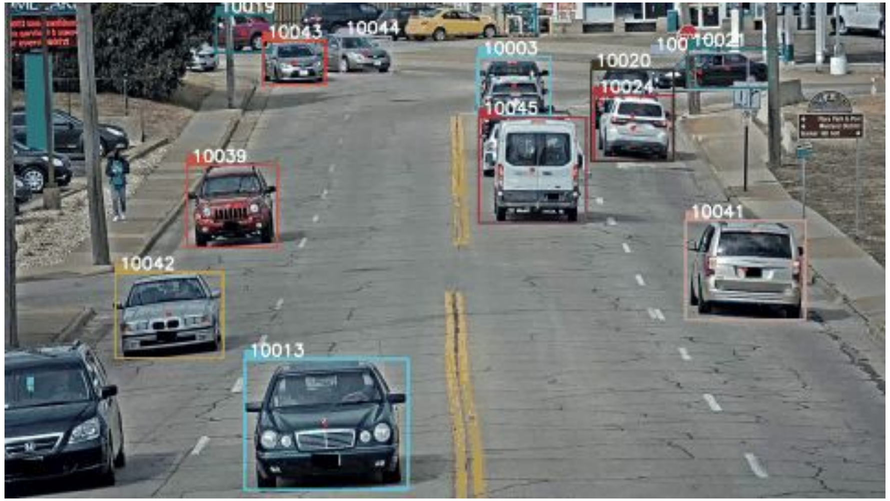


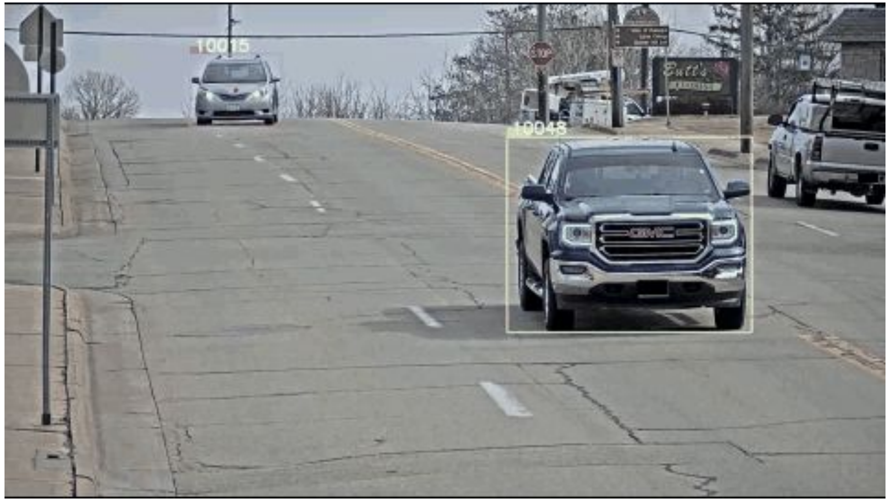
A closer look…

Frontal position with fewer occlusions, key for a good performance.

C033

C037

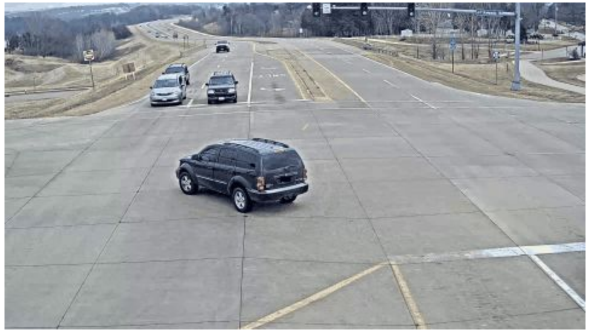
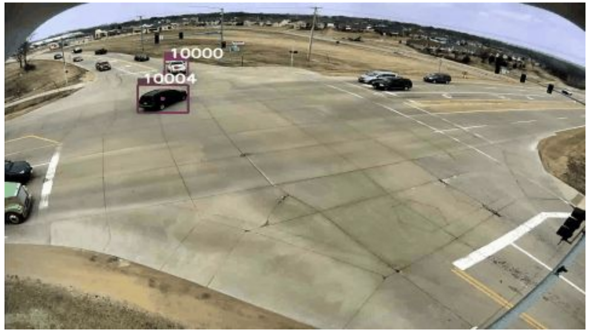


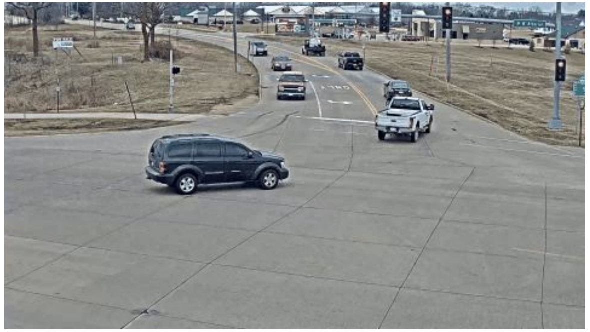


1.b. Multi-camera tracking: Qualitative results

Visual examples:

Unique track id and color across cameras.



2. Conclusions

- **Pipeline Complexity & Dependencies**: The entire process is highly dependent on previous stages. For example: if object detector fails -> entire pipeline fails; if intra-camera tracking is not well-performed -> matching objects across different cameras becomes even more challenging.
- **Challenges with GPS Coordinates**: The absence of camera calibration matrices and distortion parameters made it difficult to use GPS coordinates effectively.
- **Synchronization & Lost Frames**: The need to synchronize multiple cameras and account for lost frames during transmission adds another layer of complexity to the tracking process. Manually mapping coordinates to GPS could have been an alternative. With more time, this could have been implemented for all sequences, potentially refining the results.
- **Lack of Fine-Tuning for Different Cameras**: Our fine-tuning was limited to C010, meaning the model may not generalize well to other cameras. A better optimization of parameters and configurations could improve overall performance.
- **Optimization for Multi-Camera Tracking:** Using optimization algorithms for multi-camera matching based on computed metrics could lead to better results.
- **Evaluation Challenges:** Assessing the quality of multi-camera tracking is difficult because evaluation relies on ground truth from individual camera tracking rather than a dedicated multi-camera tracking benchmark.