NHL STENDEN
computer vision & data science

Vision-Based Cyclist Counting, and Speed Estimation Using Homographic Calibration

NHL Stenden Lectoraat in Computer Vision & Data Science

Avishkar Borkar

Supervisor: Maya Aghaei Gavari

Abstract— This study evaluates a DETR-DeepSORT pipeline for cyclist detection, tracking, counting, and speed estimation in urban environments, addressing the need for robust multi-object tracking in traffic analysis. We systematically compare three feature extractors (CLIP ViT-B32, OSNet-AIN-x1.0, and MobileNet) integrated with DeepSORT, implement a line-intersection logic for precise cyclist counting, and develop a perspective-corrected speed estimation method. Our experiments reveal that OSNet achieves marginally superior tracking performance (72.8% IDF1, 58.6% MOTA) compared to CLIP and MobileNet, though all extractors yield comparable speed estimation accuracy (MAE: 1.44–1.47, R^2: 0.885–0.889). The entry/exit line validation reduces counting errors by excluding non-relevant cyclists.Our findings indicate that while the choice of feature extractor substantially influences the consistency of cyclist tracking, it has minimal impact on the precision of speed estimates.

Index Terms— multi-object tracking, DeepSORT, DETR, cyclist analysis, speed estimation, feature extractors.

1 Introduction

According to the Netherlands Institute for Transport Policy Analysis, Dutch residents made 4.5 billion bicycle trips in 2016, covering 15.5 billion kilometers [1]. With increasing urban traffic density on cycle tracks has increased significant risks to all road users. The Dutch Safety Board reports that vulnerable road users, including cyclists and pedestrians, accounted for nearly 30% of all road fatalities in 2022, marking a 27% increase over the past decade [2]. Furthermore, the European Road Safety Observatory states that 60% of fatal accidents involving vulnerable cycle users occur at intersections, emphasizing the need for accurate detection and tracking of cyclists to enhance traffic safety and accident prevention [3].

1.1 Challenges in Cyclist Detection, Tracking, and Speed Estimation

Detecting, tracking, and analyzing cyclists in video-based traffic surveillance presents several unique challenges due to the dynamic and complex nature of real-world environments. Traditional object detection models such as Faster R-CNN [4] and YOLO [5] often struggle with real-time performance and adaptability, particularly in scenarios involving occlusions, varying lighting conditions, and cluttered backgrounds [6, 7]. Although recent deep learning-based models like Detection Transformers (DETR) [8] have improved robustness, real-world deployments still face practical limitations.

Tracking cyclists reliably across frames is complicated by their unpredictable motion patterns, frequent changes in direction or speed, and interactions with other road users. These factors contribute to identity switches and tracking failures in multi-object tracking systems [9, 10]. Challenges such as low resolution, lighting variation, and partial visibility further degrade appearance-based tracking performance.

Accurate cyclist counting and speed estimation build upon detection and tracking, and thus inherit these complexities. Dense traffic scenes, occlusions, and overlapping cyclists often lead to missed or duplicated counts [11]. Additionally, estimating speed requires precise spatial-temporal calibration of the camera, which can be disrupted by perspective distortion, variable frame rates, or inconsistent detection intervals [12, 13]. Road geometry and lighting conditions also affect the extraction of accurate motion trajectories, making reliable speed estimation even more difficult [14].

1.2 Goal of This Research

To address the challenges outlined above, this research investigates a modern tracking pipeline that combines Detection Transformers (DETR) [8] with DeepSORT [15]. While DETR provides robust detection in complex urban scenes, it is not inherently optimized for multi-object tracking. DeepSORT complements this by leveraging both motion and appearance cues to preserve cyclist identities over time.

To enhance the tracking pipeline, we evaluate various DeepSORT feature extractors, including OSNet [16], CLIP ViT-B/32 [17], and MobileNet [18], focusing on their ability to handle occlusions, lighting variations, and identity switches. Additionally, a Region-of-Interest (ROI) based bidirectional counting system [19] is implemented to accurately count cyclists and analyze directional flow. For speed estimation, we integrate homographic calibration [20], enabling metric speed measurements directly from 2D video data without requiring specialized sensors [21].

Following are the research questions:

1. How well does DETR combined with Deepsort track cyclists to validate their counts?
2. Which DeepSORT feature extractor (OSNet_AIN, CLIP_ViT-B/32, or MobileNet) achieves the optimal performance for cyclist tracking?
3. What minimum speed estimation error can be achieved using homography with uncalibrated traffic cameras ?

2 State of the Art

Advancements in deep learning have significantly improved object detection and tracking, enabling more accurate cyclist monitoring in urban environments.

- *Avishkar Borkar is a Computer Vision and Data Science student at the NHL Stenden University of Applied Sciences, E-mail: avishklar.borkar@student.nhlstenden.com*
- *Maya Aghaei Gavari is a teacher researcher at the NHL Stenden Lectoraat in Computer Vision & Data Science, E-mail: maya.aghaei.gavari@nhlstenden.com*

2.1 Object Detection Models

Early object detectors such as Faster R-CNN [4] achieved high accuracy but required significant computational resources due to anchor tuning and their multi-stage processing pipelines. YOLO [5] introduced a single-stage architecture that significantly improved inference speed, albeit with reduced precision in cluttered or complex scenes.

The Detection Transformer (DETR) [8] represents a paradigm shift in object detection by removing the need for anchor boxes and non-maximum suppression. Instead, DETR employs a self-attention mechanism to model global relationships between objects, making it more robust to occlusions, varying lighting conditions, and background clutter. Although DETR converges more slowly during training and struggles with small objects [22], it achieves mean Average Precision (mAP) comparable to Faster R-CNN on the COCO dataset [23], while offering a simpler and fully end-to-end architecture. These advantages informed our choice to adopt DETR for cyclist detection in this research.

2.2 Object Tracking Models

SORT [9] uses a Kalman filter and Hungarian algorithm for fast frame-to-frame object association but struggles with identity switches in dense or occluded scenes.

DeepSORT [15] enhances this by incorporating deep appearance embeddings to maintain identities over time. These embeddings are extracted using a lightweight re-identification network, allowing better handling of occlusions and trajectory intersections.

2.3 Feature Extractors in DeepSORT

The quality of appearance features in DeepSORT affects tracking stability. We evaluate:

OSNet-AIN-x1.0 [16]: Designed for person re-ID, it balances accuracy and efficiency for multi-scale appearance modeling.

CLIP ViT-B/32 [17]: A contrastive vision transformer providing robust embeddings, especially useful for generalizing to cyclists in varied poses or outfits.

MobileNet-v3 [18]: Efficient for edge deployment but offers lower accuracy in crowded scenes.

CLIP offers the highest tracking robustness, while OSNet offers a strong balance; MobileNet is preferred for low-latency applications.

2.4 Detection-Tracking Integration for Cyclist Counting

Combining DETR with DeepSORT enables accurate frame-to-frame identity preservation, critical for reliable cyclist counting in dense, real-world conditions. This setup forms the foundation of our system, enabling downstream tasks like bidirectional ROI-based counting and speed estimation through trajectory tracking and spatial calibration.

3 Materials and Methods

3.1 Dataset

The research utilizes a private dataset. Dataset "Utrecht," consists of three primary folders: PSFV-01-N413-Zuid, PSFV-02-N237-Zuid, and PSFV-03-N237-Noord. Each folder contains subfolders organized by date in the format YYYYMMDD, spanning from 20221215 to 20221228. Within each date folder, video files are stored in hourly subfolders (00 to 23), with each video representing a 15-minute interval. The videos are recorded in 1920x1080 resolution at a frame rate of 25 fps, with each 15-minute video containing 22,500 frames. Consequently, each day folder contains 24 videos, amounting to 540,000 frames per day. Over the 14-day period, each folder contains a total of 7,560,000 frames. Fig. 1

For this research, we utilized the subfolder PSFV-01-N413-Zuid across all days from 20221215 to 20221228. However, to focus on the most relevant periods, we selected only the hours 0800, 0900 as these are statistically the busiest cycling hours in the Netherlands [24]. This selection allowed to concentrate on peak traffic conditions, ensuring the dataset's relevance to the study of cycling behavior during high-traffic periods.



Fig. 1: Sample image from the dataset: pc5-1832-PSFV-01-N413-Zuid_20221227_084500765_frame_12135.jpg

3.2 Preprocessing

3.2.1 Extracting frames from videos

For each timestamp folder within a specific day, there were four 15-minute long videos, corresponding to one hour of footage. To facilitate the downstream tasks, these videos needed to be converted into frames that could be input into the models. Given that the original video frame rate was 25 frames per second (fps), which is the standard for MP4 videos, we chose to downsample the videos by extracting every 5th frame. This strategy reduced the frame rate to 5 fps, effectively decreasing the computational load and improving processing efficiency while maintaining sufficient temporal resolution for the tasks at hand.

3.2.2 Formulating Ground Truth

The raw frames were annotated using the CVAT tool [25] also every individual cyclist was given a unique track ID to ensure multiple object tracking across frames, thereby establishing a reliable ground truth for subsequent analysis.

Split	Frames	Tracks
Train	13877	96
Validation	7947	64
Test	9241	71

Table 1: Manually Annotated Frames and Corresponding Tracks

3.3 Model Architecture

In the first stage, an object detector is utilized to identify and predict multiple cyclists within each frame, generating bounding boxes that represent the cyclists. These bounding boxes are characterized by the coordinates - x, y, width, height. In the second stage, the extracted bounding box coordinates are fed as inputs into the object tracking algorithm. This tracker processes the information of the bounding box in consecutive frames, enabling the continuous tracking of cyclists throughout the video sequence. Once the cyclists are tracked, we then calculate the count of cyclists and estimate their speeds.

3.3.1 Detection Transformer (DETR) For Detecting Cyclists

Detection Transformer (DETR) [26] is an object detection framework that formulates the task as a set prediction problem using a transformer architecture. Unlike traditional methods, DETR does not require anchor boxes or post-processing techniques like Non-Maximum Suppression. The model utilizes a backbone network to extract feature maps, which are then processed by a transformer encoder-decoder. The encoder captures global spatial dependencies, while the decoder directly predicts bounding box coordinates and class labels for each object. DETR excels in detecting objects in crowded or occluded scenes by leveraging global context through the

transformer's self-attention mechanism. In this work, we initialized the model with COCO-pretrained weights [23] and finetuned it on our dataset, comprising 13,877 training frames and 7,947 validation frames. The model was trained for 30 epochs using the AdamW optimizer with an initial learning rate of 2×10^{-4}, a batch size of 4, and a learning rate drop after 20 epochs. To mitigate overfitting, early stopping based on validation loss was employed, and the model with the best validation performance was retained.

3.3.2 DeepSort For Cyclists Tracking Across Frames

DeepSORT (Deep Simple Online and Realtime Tracker) is an advanced tracking algorithm that builds upon the original SORT (Simple Online and Realtime Tracker) by integrating deep learning-based appearance features for improved robustness [27]. It combines a Kalman filter for motion prediction [28] with a deep association metric learned via a convolutional neural network trained on person re-identification data [29]. DeepSORT assigns unique identities to detected objects across frames by leveraging Mahalanobis distance for gating associations [30] and solving data association through the Hungarian algorithm [31]. This significantly enhances tracking performance in crowded environments with occlusions. For this research we have experimented with different feature extractors like OSNet AIN, CLIP ViT-B/32, and MobileNet.

3.4 Bidirectional Cyclists Counting Methodology

We only count valid cyclists that enter from the bottom of the cycle track and only end at the opposite side of the cycle track and same for the other way. If a cyclist enters from any other entry points or exit points we do not want to consider that particular cyclist in the tally count.

Our system implements directional cyclist counting through a three-stage process (illustrated in Figure 2):

1. **ROI Definition**: Manual annotation of polygonal region of interest (ROI)
2. **Virtual Lines**: Establishment of two detection lines for opposite directions
3. **Trajectory Validation**: Tracking of center bounding box coordinates

An object is considered to have entered the ROI when the centroid of its bounding box intersects the designated entry line (blue). Each entry is recorded, and the trajectory is tracked until the object intersects the exit line. A valid count is registered only if both entry and exit occur through the blue lines. Intersections with red boundary lines at any point render the detection invalid, and such instances are excluded from the final count.

3.5 Speed Estimation Using Homography

Another indicator that we studied in this research was to estimate speeds of the individual cyclists. One of the major problem faced in this was that the videos were from an angled traffic surveillence camera, meaning the images had perspective distortion.

3.5.1 Perspective Distortion Correction

The angled surveillance camera introduces projective distortion that makes direct pixel-to-meter conversions invalid. We rectify this using homographic transformation, leveraging known physical dimensions of road markings. The specifications indicate repeating 0.5×0.5m painted blocks with 0.5m gaps, creating a 1m periodicity (Fig. 3).

3.5.2 Homography Matrix Calculation

The homography matrix $\mathbf{H}$ defines a perspective transformation between world coordinates $(x, y, 0)$ on the ground plane and image pixel coordinates (u, v). The 3×3 matrix is computed using point correspondences under the following conditions:

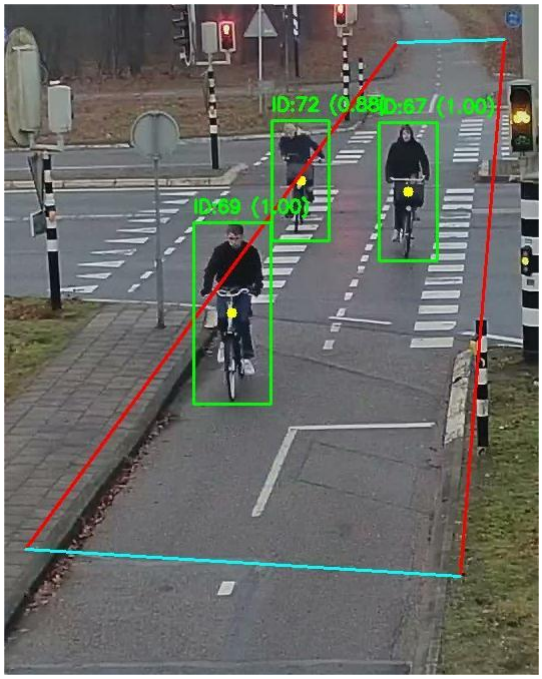

Fig. 2: The rectangular polygon denotes the Region of Interest (ROI). Blue lines represent the designated entry and exit lines, while yellow dots indicate the centroids of detected bounding boxes.

- **Planar Constraint**: All world points must lie on a flat surface ($z = 0$). For this research we have assumed the road is considerably flat.
- **Minimum Correspondences**: Requires $N \geq 4$ point pairs

The homogeneous transformation is expressed as:

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad \text{where} \quad \mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{1}$$

The homography matrix encapsulates:

- Rotation and translation components $(h_{11}, h_{12}, h_{21}, h_{22}, h_{13}, h_{23})$
- Perspective effects (h_{31}, h_{32})
- Overall scaling (h_{33})

Computation Method:

1. For each correspondence $(x_i, y_i) \leftrightarrow (u_i, v_i)$, construct two equations:

$$u_i(h_{31}x_i + h_{32}y_i + h_{33}) = h_{11}x_i + h_{12}y_i + h_{13}$$
$$v_i(h_{31}x_i + h_{32}y_i + h_{33}) = h_{21}x_i + h_{22}y_i + h_{23}$$

2. Stack all equations into matrix form $\mathbf{Ah} = 0$ where $\mathbf{h}$ is the flattened homography matrix
3. Solve using Singular Value Decomposition (SVD), taking the singular vector corresponding to the smallest singular value
4. Normalize the matrix such that $h_{33} = 1$ (or $\|\mathbf{h}\| = 1$)

3.5.3 Implementation Details

The homography matrix $\mathbf{H}$ is computed using four reference points with known world coordinates (Fig. 4):

$$\begin{aligned} &\text{Image Points } (u, v): \quad [937.30, 390.20], \ldots, [862.40, 504.00] \\ &\text{World Points } (x, y): \quad [0, 0], \ldots, [0, 10.5] \quad \text{(in meters)} \end{aligned} \tag{2}$$

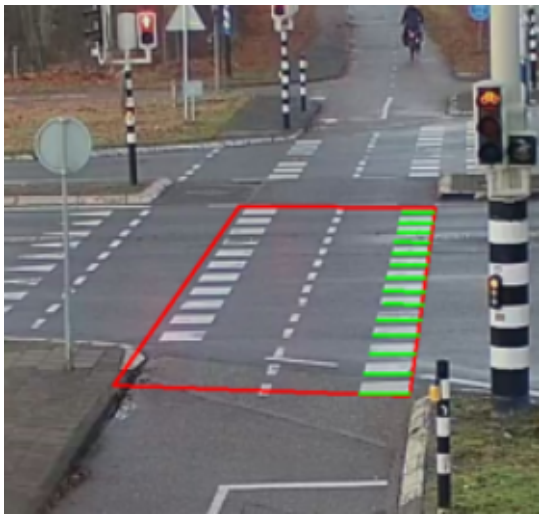

Fig. 3: Ground truth road markings: 0.5m squares (white) with 0.5m gaps (grey) creating 1m repeating pattern.

3.5.4 Establishing Ground Truth for Speed Calculation

To compute the Mean Absolute Error (MAE) in the estimated speeds of cyclists, we first establish a ground truth reference by restricting speed calculations to a known region of the cycle track. This region has precisely measured real-world dimensions, ensuring accurate speed validation.

3.5.5 Speed Calculation

To estimate speed, we analyzed the movement trajectories of tracked cyclists using the following steps:

1. **Frame Processing Parameters:**

   - Effective frame rate: 5 fps (after 5× downsampling from 25 fps)

2. **Tracking Initialization:** For each detected cyclist track:

$$\begin{cases} \text{bbox} = (x_{min}, y_{min}, x_{max}, y_{max}) \\ \text{center}_{pix} = \left(\frac{x_{min}+x_{max}}{2}, \frac{y_{min}+y_{max}}{2} \right) \\ \text{center}_{real} = H \cdot \text{center}_{pix} \quad \text{(homography transform)} \end{cases}$$

3. **Speed Computation:** For consecutive positions of the same track:

$$\text{speed}_{m/s} = \frac{\|\text{center}_{real}^{(t)} - \text{center}_{real}^{(t-\Delta t)}\|_2}{\Delta t} \tag{3}$$

where Δt is computed from frame difference:

$$\Delta t = \frac{\text{frames_elapsed}}{5} \tag{4}$$

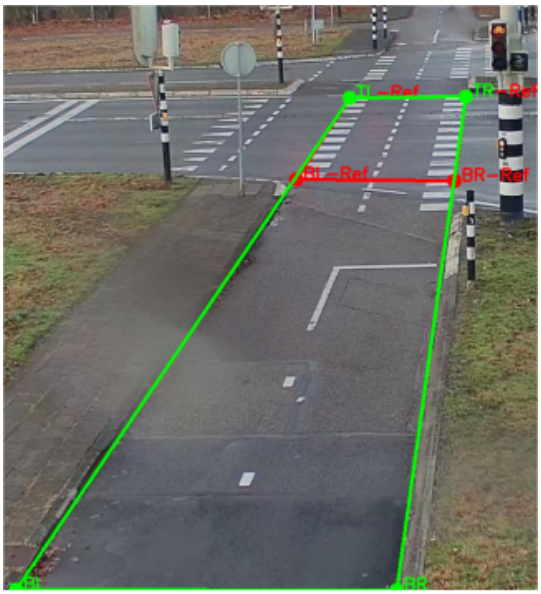

Fig. 4: Calibration points (red) and transformed ROI (green). Scale factor: 100 pixels/meter.

Algorithm 1 Speed Calculation Pipeline

Require: Video frames at 5 fps, homography matrix H
Ensure: Track speeds in km/h

```
Initialize empty track_info dictionary
for each frame do
    for each tracked cyclist do
        if track is new then
            Initialize tracking record with:
                • Real-world positions array
                • Speed history
                • Last frame counter
        else if position updated then
            Δt ← (current_frame − last_frame)/5
            d ← ||current_pos − last_pos||_2
            speed_{m/s} ← d/Δt
            Update track statistics
        end if
    end for
end for
```

4 Evaluation Metrics

To evaluate our cyclist tracking and speed estimation pipeline, we used different metrics for each task.

Tracking Metrics:

- **IDF1 (ID F1 Score):** Measures how well the system keeps track of each cyclist's identity over time. It balances precision and recall for identity matching.

$$\text{IDF1} = \frac{2 \cdot \text{IDTP}}{2 \cdot \text{IDTP} + \text{IDFP} + \text{IDFN}}$$

- **MOTA (Multiple Object Tracking Accuracy):** Combines errors from missed detections, false positives, and identity switches into a single score. Higher MOTA means better overall tracking.

$$\text{MOTA} = 1 - \frac{\sum(\text{FN} + \text{FP} + \text{IDSW})}{\sum \text{GT}}$$

- **Precision:** The percentage of correctly detected cyclists out of all detected cyclists. It shows how many detections were actually correct.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **Recall:** The percentage of actual cyclists that were correctly detected. It shows how many true cyclists the system was able to track.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Speed Estimation Metrics:

- **MAE (Mean Absolute Error):** Measures the average absolute difference between predicted ($\hat{y}_i$) and true (y_i) speeds. Lower MAE means more accurate predictions.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- **RMSE (Root Mean Square Error):** Similar to MAE, but gives more weight to larger errors. It helps identify how severe the biggest mistakes are.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **R^2 (Coefficient of Determination):** Measures how well the predicted speeds match the true speeds. A value close to 1 means the predictions are very accurate.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Counting Metrics:
To evaluate counting accuracy, we manually reviewed videos to determine the ground-truth number of cyclists and compared this with the model's predicted count. We report:

- **Total Cyclists (GT):** The actual number of cyclists manually counted in the video.
- **TP (True Positives):** The number of cyclists correctly counted by the system.
- **F1-Score:**

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5}$$

where $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$

5 Experiments and Results

We conducted 3 systematic experiments to evaluate and optimize our cyclist tracking pipeline. First, we explore enhancements to the DeepSORT tracker. Specifically, we tested different feature extractor architectures for the appearance model to evaluate their effect on tracking performance. Key metrics included increase in precision, recall, improvements in Multiple Object Tracking Accuracy (MOTA), and better ID consistency across frames.

For counting of cyclists, we manually counted the number of valid cyclists through a visual inspection and then compared with the tracking pipeline's counting output.

We then finally validated our system's ability to estimate cyclist speeds. We computed speed using homography-projected trajectories and compared these estimates to ground truth speeds collected within a calibrated region. The accuracy of speed estimation was quantified using the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) between the predicted and reference values.

5.1 Experiment 1: Feature Extractors Analysis for DeepSORT

We conducted a comparison of three ReID feature extractors integrated with our DETR-DeepSORT pipeline. The evaluation focuses on tracking continuity and lower false positive predictions:

Table 2: Tracking Metrics for Feature Extractor Experiment using DETR

Model	Feature Extractor	IDF1 ↑	MOTA ↑	Precision ↑	Recall ↑
DETR	CLIP ViT-B32	71.9%	58.2%	78.2%	81.3%
DETR	OSNet-AIN-x1.0	72.8%	58.6%	78.5%	81.4%
DETR	MobileNetV3	71.9%	58.2%	78.2%	81.4%

Table 2. presents the tracking performance of the DETR model using three different feature extractors. Among them, DETR with OSNet-AIN-x1.0 achieved the highest IDF1 score (72.8%) and MOTA (58.6%), indicating improved identity preservation and overall tracking accuracy. It also slightly outperformed the other configurations in both precision (78.5%) and recall (81.4%). These results demonstrate that the choice of feature extractor influences tracking performance, with OSNet-AIN-x1.0 providing the most balanced and effective representation in this setup.

5.2 Experiment 2: Counting Valid Cyclists

For this experiment, as illustrated in Fig. 2, only cyclists whose entry and exit points intersect the designated blue lines are included in the final count. Any cyclists entering or exiting through other lines are excluded. The cyclist detections were verified through manual inspection to ensure an accurate ground truth reference. Table 3 summarizes the detection results across different feature extractors used with DeepSORT.

Each configuration yielded five false positives, which were primarily caused by scooters or mopeds being misclassified as cyclists. Additionally, a few cyclists were missed due to extreme weather conditions such as raindrops on the camera lens, which affected visibility. In other cases, erratic cyclist movement led them to temporarily exit the defined region (e.g., swaying outside the valid area) before re-entering and exiting through valid lines. This partial visibility within the region resulted in broken tracking, thereby excluding them from the final count. These final counts were also verified by manually watching the predicted video and classifying true positives and false positives.

Table 3: Cyclist Detection Performance by Model and Feature Extractor

Model	Feature Extractor	Predicted Cyclist	TP	FN	F1-Score ↑
DETR	CLIP ViT-B32	59	54	12	86.4%
DETR	OSNet-AIN-x1.0	59	54	12	86.4%
DETR	MobileNetV3	60	55	11	87.3%

5.3 Experiment 3: Speed Estimation Using Perspective Correction

In this experiment, we compute the average speed of each individual cyclist. However, for validation purposes, we calculate the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) and R^2 only within the known reference region, where real-world measurements are available Fig 3.

To evaluate the impact of different feature extractors within the DeepSORT tracking pipeline, we used DETR for object detection and compared CLIP (ViT-B32), MobileNetV3, and OSNet. The results show that all three models perform comparably, with OSNet achieving the lowest RMSE of 2.17 km/h and the highest R^2 value of 0.889, indicating better correlation with ground truth speeds. MobileNetV3 had the lowest MAE of 1.44 km/h, while CLIP slightly

Table 4: Regression Metrics Using DETR for Detection and DeepSORT with Different Feature Extractors for Tracking

Model	Feature Extractor	MAE ↓	RMSE ↓	R^2 ↑
DETR	CLIP ViT-B32	1.47	2.20	0.885
DETR	MobileNetV3	1.44	2.18	0.888
DETR	OSNet-AIN-x1.0	1.45	2.17	0.889

lagged behind on all three metrics. These findings suggest that OSNet provides the most accurate tracking features when paired with DETR and DeepSORT in this setup for speed estimation.

6 Discussion, Conclusion & Future Work

6.1 Discussion

This study assessed the performance of the DETR-DeepSORT system in tracking cyclists and estimating their speeds under urban surveillance conditions. We focused on three parts: the choice of feature extractor, the cyclist counting method, and the accuracy of speed estimation. OSNet-AIN-x1.0 gave slightly better tracking results than CLIP ViT-B32 and MobileNet, but the improvement was small. This raises the question of whether the extra processing power required by OSNet is viable for such a small gain. For cyclist counting, our entry/exit line method helped reduce false positives by only counting relevant cyclists. In speed estimation, all models performed similarly well, with only small differences in accuracy. This suggests that the choice of feature extractor has less impact on speed estimation than on tracking.

The idea behind counting cyclists and the estimation of speed of cyclists is that it can provide baseline traffic metrics that are maybe useful to city planners. These metrics help assess infrastructure usage and detect trends over time, such as speeding near intersections or overcrowding. Insights derived from these data points can help make informed decisions such as redesigning junctions, adjusting signal timing, or placing physical barriers to improve safety and efficiency.

6.2 Conclusion

Our study shows that DETR-DeepSORT with OSNet-AIN-x1.0 offers the best overall performance for cyclist tracking, though only slightly better than simpler models like MobileNet. Speed estimation was consistently accurate across all feature extractors (MAE - 1.5 km/h), indicating that tracking accuracy is the main limitation, not the estimation of speed itself. This pipeline is suitable for applications like urban mobility analysis and traffic monitoring, but improvements in real-time efficiency and further validation of the counting method are needed as DETR is comparatively a heavy model to deploy for real time scenarios.

6.3 Future Work

Future work should focus on three important areas. First, we want to improve camera calibration by using the errors in speed estimation to adjust camera settings like angle and zoom. This would reduce the need for manual setup. Second, adding LiDAR or radar sensors can help check and improve speed measurements, especially in bad visibility. This requires careful timing between devices and smart ways to combine the data. Third, we need to test the system at night or in low light. This could involve using brighter images, thermal cameras, or training the model on night-time data. These steps are important to make the system work better in real-world, everyday situations.

7 Ethical, Legal and Social Aspects

The recording of cyclists in public bicycle lanes necessitates careful consideration of ethical, legal and social implications, even when conducted for safety improvement purposes.

From an ethical perspective, the video data used in this study originates from a private dataset that has not been published or made publicly accessible. The recordings were collected as part of a closed research initiative, and their use complies with applicable ethical guidelines and Dutch privacy laws governing research in non-public contexts. If this research is to be implemented in practice for real world application, The system must provide clear notice of recording activities while ensuring collected footage cannot be used for purposes beyond its intended safety-focused scope. Legal considerations can be making provisions for users to delete their personal data on demand and finally on the social aspect, keeping the system transparent to the community can avoid any social conflicts.

Acknowledgements

This project is financially supported by Regieorgaan SIA (part of NWO) under the RAAK Publiek project BikeSafeAI.

References

[1] Netherlands Institute for Transport Policy Analysis. Cycling facts 2016. 2016.

[2] Dutch Safety Board. Road safety report 2022. 2022.

[3] European Road Safety Observatory. Cyclist safety in the eu. 2023.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[5] Joseph Redmon, Shyam Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*, 2016.

[6] X. Zhang and Y. Liu. Occlusion handling in object detection: A survey. *IEEE Transactions on Image Processing*, 29:1234–1249, 2020.

[7] M. Hnewa and H. Radha. Robust object detection under adverse weather conditions. *IEEE Access*, 9:56789–56799, 2021.

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872*, 2020.

[9] Alex Bewley, Sanghyun Ge, Lian Ott, Fabio Ramos, and Ben Upcroft. Sort: Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468. IEEE, 2016.

[10] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *IEEE International Conference on Image Processing (ICIP)*, 2017.

[11] Longyin Chen and et al. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[12] V. Kastrinaki, M. Zervakis, and K. Kalaitzakis. A survey of video processing techniques for traffic applications. *Image and Vision Computing*, 21(4):359–381, 2003.

[13] Zengyang Sun, George Bebis, and Ronald Miller. A novel vehicle speed detection method using a single camera. In *IEEE Workshop on Applications of Computer Vision (WACV)*, 2006.

[14] Yuan Yuan and et al. Vehicle and pedestrian traffic monitoring with on-road cameras: A review. *IEEE Transactions on Intelligent Transportation Systems*, 20(6):1926–1943, 2019.

[15] Nicolai Wojke, Alex Bewley, David Paulus, Mark Luiten, and Ben Upcroft. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[16] Kaiyang Zhou, Yongxin Yang, Andrea Cavallaro, and Tao Xiang. Omni-scale feature learning for person re-identification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3702–3712, 2019.

[17] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

[18] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[19] Meng Yang, Guanbin Li, Xiang Li, and Liang Lin. Bidirectional counting network for crowd counting. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(10):3448–3460, 2019.

[20] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[21] Ninad Kanhere, Stanley Birchfield, and Wayne Sarasua. Vision-based vehicle speed estimation: A survey. *IEEE Intelligent Transportation Systems Magazine*, 14(1):2–24, 2022.

[22] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations (ICLR)*, 2021.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755, 2014.

[24] Netherlands Institute for Transport Policy Analysis. Cycling facts 2023. Available at: `https://english.kimnet.nl/binaries/kimnet-english/documenten/publications/2024/01/10/cycling-facts-2023/KiM%2Bbrochure%2BCycling%2Bfacts%2B2023_def.pdf`, 2023. Accessed: 2024-03-22.

[25] Anton Mikhailov, Vladimir Karpukhin, Alexey Karpov, et al. Cvat: Open-source tool for annotating videos and images. `https://github.com/openvinotoolkit/cvat`, 2018. Accessed: 2025-03-22.

[26] N. Carion, F. Massa, G. Synnaeve, and et al. End-to-end object detection with transformers. *European Conference on Computer Vision (ECCV)*, pages 213–229, 2020.

[27] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracker with a deep association metric. 2017.

[28] Greg Welch and Gary Bishop. An introduction to the kalman filter. 1995.

[29] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. 2017.

[30] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (India)*, 1936.

[31] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 1955.