Mirror, Mirror on the Road, Is There a VRU too Close?

Vinicius Avena, Rodrigo S. Couto, Miguel Elias M. Campista, Luís Henrique M. K. Costa
Universidade Federal do Rio de Janeiro - GTA/Poli/COPPE/UFRJ - Rio de Janeiro, Brazil
{avena, rodrigo, miguel, luish}@gta.ufrj.br

***Abstract*—This work presents a proactive computer vision-based ADAS system designed to enhance VRU safety in risk zones at the rear of the vehicle and filter lanes. The system combines object detection, multi-object tracking, and risk assessment to generate real-time proximity alerts. Compared to a detection-only approach, the proposed method shows a 19% improvement in precision and a 10% increase in F1-score, while introducing a latency overhead of less than 1 ms. Evaluated on edge devices, the system maintains efficient performance across resource-constrained platforms. To support research in this domain, we introduce FilterLane-VRU, a new dataset of rearview urban traffic scenarios with temporal proximity annotations. Results validate the feasibility of deploying the system in real-world ADAS applications, offering a reliable and cost-effective solution for VRU safety.**

***Index Terms*—VRU detection, object tracking, risk assessment, ADAS.**

I. Introduction

The rapid development of Advanced Driver Assistance Systems (ADAS) has significantly improved vehicle safety, especially with the advancements in real-time environmental perception using deep learning-based solutions. Nonetheless, these systems still face substantial challenges, especially regarding accurate detection and tracking of road users in complex traffic scenarios [1]. The issue is critical in the case of Vulnerable Road Users (VRUs), such as pedestrians, cyclists, and motorcyclists, whose increased exposure and less predictable behavior make them difficult to detect.

VRUs represent a significant share of traffic fatalities. In urban environments, particularly in developing countries, the increasing reliance on motorcycles and bicycles has exacerbated this problem in recent years. Studies indicate that VRUs account for nearly half of all traffic fatalities worldwide [2], with 75% of fatalities in India in 2022 involving VRUs, including 43% motorcyclists [3]. These trends highlight the growing threat faced by VRUs, making their safety a critical concern for modern intelligent transportation systems.

One of the main causes of accidents involving VRUs is the presence of high-risk areas around the vehicle, such as blind spots and filter lanes. While current solutions provide information about nearby objects—such as blind spot warning systems and ultrasonic sensors—they are not activated until the two-wheeler is already in these risk zone. As a result, these systems fail to detect approaching VRUs before they enter in a high-risk area. This issue is particularly problematic in filter lanes—narrow spaces between adjacent lanes where motorcyclists and bicyclists often travel. In these situations, VRUs are typically out of the driver's field of view, making it difficult to anticipate their movement and potential danger.

Several strategies have been proposed to enhance VRU safety. Infrastructure-driven methods utilize fixed sensors, such as cameras, radars, and LiDARs installed along roads, to anticipate risks in complex urban scenarios [4], [5]. However, these systems rely heavily on connectivity and large dedicated infrastructure, limiting adoption. Onboard radar-based detection methods provide robustness under adverse weather conditions but suffer from reduced performance at higher speeds [6]. Conversely, LiDAR-based solutions achieve high precision in VRU segmentation [7] but are constrained by sensor costs and computational processing requirements.

Few studies have explored camera-based VRU detection in blind spots. The system proposed by [8] is limited by its reliance on elevated, fixed cameras designed for large vehicles, restricting its application to conventional vehicles. Furthermore, the monocular depth detection method of [9], targeted at right-turn blind spots, demonstrates low computational complexity yet lacks generalization capability due to reliance on fixed camera parameters. Notably, research addressing VRU detection using rearview cameras remains scarce, despite the distinct challenges presented by risk zones such as filter lanes and blind spots.

In this paper, we propose a novel system for generating alerts for VRUs in high-risk rear vehicle zones, using a rear-facing camera, with a specific focus on detecting two-wheelers in adjacent filter lanes. The system addresses the limitations of existing solutions by detecting approaching VRUs before they enter the blind spot, using advanced object detection, multi-object tracking, and trajectory analysis, even in complex conditions like occlusions and dynamic traffic flow. Moreover, we introduce FilterLane-VRU, a new dataset of real-world rear camera recordings with temporal proximity alerts, designed to address the underexplored challenge of detecting VRUs in rear blind spots. The proposed system significantly improves detection performance, with a 19% increase in precision and provides a computationally efficient solution for embedded systems, such as single-board computers and edge GPUs used in ADAS, demonstrating its applicability in vehicular scenarios.

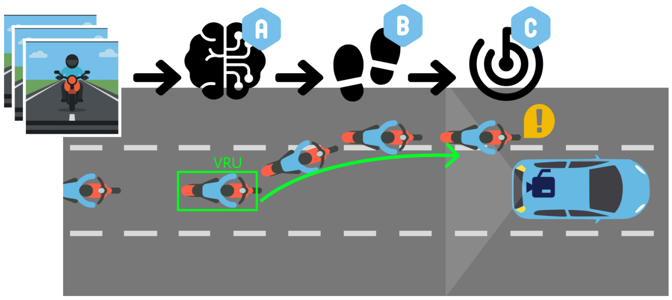

Fig. 1. Overview of the system for VRU detection, tracking, and risk-based alert generation in rear blind spots.

II. System Model

The proposed system is designed to detect and track Vulnerable Road Users in rear blind spots to generate accurate alerts when a collision risk is identified. As shown in Fig. 1, the proposed method operates in three continuous stages: (A) camera frames are processed to detect VRUs through a deep neural network; (B) each detection is passed to a multi-object tracking module, which estimates motion trajectories using temporal information; and (C) the system evaluates the risk they pose based on their trajectory and position relative to the vehicle, which issues an alert only in situations of actual risk, enhancing safety and minimizing false positives.

A. VRU Object Detector

Object detection is a fundamental task in computer vision, where the objective is to identify and localize objects within an image. The YOLO (You Only Look Once) series [10] is a well-established real-time object detection model that predicts both bounding boxes and class probabilities in a single pass, making it particularly efficient for real-time applications requiring low-latency responses, such as the vehicular environment.

For this work, we selected YOLO11 [11] due to its balanced performance in terms of both accuracy and speed. This model incorporates enhanced spatial attention mechanisms, which significantly improve its ability to detect small or overlapping objects—such as partially-occluded VRUs. These improvements allow the model to focus on relevant features, making it ideal for detecting road users in complex, dynamic environments. By using a pre-trained YOLO11 model, we integrate its object detection capabilities efficiently into the system for real-time VRU alert generation in filter lanes, eliminating the need for additional training.

The model is configured to detect only on VRU classes—motorcycles, bicycles, scooters and people—which allows the subsequent tracking and risk assessment stages to focus exclusively on these road users and streamlining the alert generation process.

Furthermore, bounding boxes that represent a same VRU—such as a motorcycle and its rider—are merged into a single box using the Intersection over Union (IoU) metric (Eq. 1). IoU quantifies the overlap between two bounding boxes: when it exceeds a threshold of 0.3, the bounding boxes are merged into a single one. This merging process improves detection precision by defining better-adjusted boundaries for the two-wheeled vehicle and rider and reducing the number of objects to track. The IoU metric is defined as:

$$\mathrm{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}. \tag{1}$$

Finally, the merged bounding box position is passed to the next module, the Multi-Object Tracker, which manages the tracking of all detected objects across subsequent frames.

B. VRU Tracking and Approach Filtering

Tracking is an essential task in dynamic environments (e.g., urban traffic), where the goal is to maintain the identity of detected objects across consecutive frames. This allows the system to monitor an object's movement over time, predict its future trajectory, and assess potential risks. In the case of VRUs, tracking is particularly important for identifying when these objects approach the critical filter lanes.

To ensure both efficiency and accuracy, particularly on edge devices, we selected ByteTrack [12] due to its lightweight design and demonstrated effectiveness in tracking multiple objects in dense scenes and occlusion scenarios. ByteTrack is a state-of-the-art algorithm, that tracks objects by combining two key strategies: motion prediction and appearance modeling. The motion prediction uses a Kalman filter to estimate the future position of an object based on its previous movement, which enables ByteTrack to track VRUs even when they are temporarily occluded or partially hidden. Appearance modeling ensures that the identity of the VRU is maintained across frames by comparing the object's appearance features, such as shape and texture, to re-identify it after occlusions or when it is partially blocked by other vehicles.

Once the VRUs are tracked, it is crucial to assess whether they are approaching along the filter lane, which could lead to a potential collision. To achieve this, we define the approaching vector $\mathbf{v}_{\mathrm{FL}}(t)$, which originates from the VRU's current position and points to the centroid of the predefined risk zone polygon, as shown in Figure 2. We then compare the VRU's trajectory vector $\mathbf{v}_{\mathrm{VRU}}(t)$, computed as the average displacement over a temporal window from Kalman filter tracking, with $\mathbf{v}_{\mathrm{FL}}$ using the cosine similarity as a trajectory alignment metric (θ), defined as:

$$\theta(\mathbf{v}_{\mathrm{FL}}, \mathbf{v}_{\mathrm{VRU}}) = \frac{\mathbf{v}_{\mathrm{FL}} \cdot \mathbf{v}_{\mathrm{VRU}}}{|\mathbf{v}_{\mathrm{FL}}||\mathbf{v}_{\mathrm{VRU}}|}. \tag{2}$$

This metric evaluates the correlation between the VRU's trajectory and the route leading to the rear-side filter lane, which partially overlaps with the vehicle's blind spot, creating a potential risk zone, as depicted the reddish region of Figure 2. If the cosine similarity exceeds a predefined threshold θ_{min}, the VRU is considered to be approaching the high-risk area. This verification, based on trajectory alignment, minimizes false positives by ensuring that alerts are triggered only when the VRU is heading directly towards the risk region, and the proximity to this area is assessed in a subsequent step.

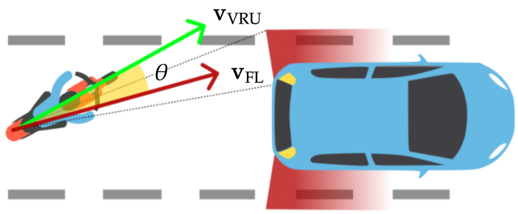

Fig. 2. Illustration of the VRU approaching vector $\mathbf{v}_{\mathrm{FL}}$ and the VRU trajectory vector $\mathbf{v}_{\mathrm{VRU}}$, with the angle θ representing their alignment.

C. VRU Risk Assessment

Once a tracked VRU is identified as following an approaching trajectory, the next step is to assess whether it enters a predefined risk zone around the ego-vehicle's rear, warning when a VRU is approaching dangerously. The risk zone is a predefined polygonal area in the rear camera's image space, which is projected to cover the adjacent filter lanes where rear-side collisions are more likely to occur. The pseudo-code of the system is shown in Algorithm 1.

Algorithm 1 Pseudo-code of the VRU Risk Assessment.

Require: Video sequence V, pre-defined risk zone Z, alignment threshold $\theta_{\min}$

```
for each frame f_k ∈ V do
    Detect VRUs in f_k using YOLO11
    Organize detected VRUs into a priority queue Q
    for each VRU v_i ∈ Q do
        Initialize or update v_i's track using ByteTrack
        Compute v_VRU from Kalman filter predictions
        Compute v_FL from v_i's position to its blind spot
        Calculate alignment metric θ_i using Equation 2
        if θ_i ≥ θ_min and v_i's position intersects Z then
            Flag v_i as high-risk and trigger alert
        end if
    end for
end for
```

In this process, the system first identifies the VRUs detected in each frame and organizes them into a priority queue based on proximity, trajectory alignment, and model confidence, in that order. For each VRU, the system tracks its movement and evaluates the alignment of the VRU's trajectory to the predefined risk zone. The system computes the intersection between the VRU's bounding box and the ego-vehicle's risk zone Z. If the VRU is within the zone, an alert is triggered.

This decision-making process ensures that the system responds exclusively to relevant threats along the filter lanes, avoiding unnecessary alerts for VRUs that traverse these risk zones but are not on a potential collision path, minimizing false positives and ensuring higher reliability in complex urban traffic environments.

III. Experimental Evaluation

In this section, we evaluate the proposed proactive VRU alert system introduced in Section II. The experiments focus on the system's ability to detect and assess VRUs approaching rear blind spots under realistic urban traffic conditions.

A. Dataset: FilterLane-VRU

To support the evaluation of our system under realistic traffic conditions, we have built **FilterLane-VRU**, a dataset focused on rearview perception in filter lanes—a scenario overlooked by existing vehicular benchmarks. Unlike conventional datasets, FilterLane-VRU captures VRU interactions in rear blind spots with temporal annotations for alert-based evaluation[1].

FilterLane-VRU includes 94 video sequences, each 10 seconds long and recorded at 30 frames per second. The videos were captured in the urban area of Rio de Janeiro, Brazil, using a vehicular camera with a resolution of 2560 x 1440 pixels and a wide-angle lens providing a 150° diagonal field of view. This setup effectively covers the lateral regions around the vehicle, especially focusing on the filter lanes. The dataset captures diverse real-world scenarios, including high-speed expressways, congested traffic conditions, and red lights—situations in which lane filtering by VRUs frequently occurs.

Each frame is annotated with alert signals derived from the VRU's motion relative to predefined high-risk zones. An event is defined as a period during which a VRU is within a risk zone and potentially poses danger by approaching it. Within these intervals, the system is expected to generate alerts—real-time warnings triggered when a VRU enters or traverses a risk region. The annotations specify the start time and the side of the vehicle (left or right) where the event occurs, enabling precise evaluation of both timing and localization of alerts. In total, the dataset includes 62 alert events and 30 near-alert events, the latter corresponding to VRUs that approach or pass through the risk zone without posing an immediate threat. Figure 3 illustrates two scenes where alerts should be triggered, according to the dataset annotations.

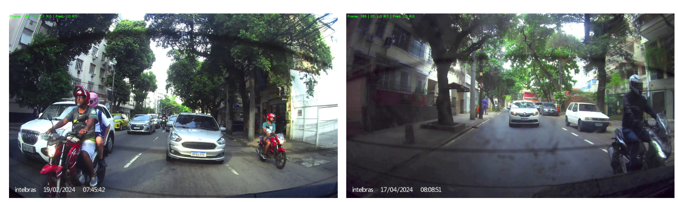

Fig. 3. Samples of scenes from FilterLane-VRU where an alert should be triggered.

B. Hardware Platforms and Implementation

To assess the feasibility of the proposed system in vehicular environments, the complete framework was tested on a heterogeneous set of devices with varying computational resources. The chosen devices represent three levels of processing power: a Raspberry Pi 5 (CPU-only), a Jetson Nano (GPU-accelerated via TensorRT), and a high-resource computing unit with an

[1]The dataset is available for benchmarking purposes upon request to the authors.

NVIDIA RTX 2080Ti. Table I summarizes the specifications of the evaluated hardware.

TABLE I
MAIN CHARACTERISTICS OF THE EVALUATED HARDWARE.

Specs	**Raspberry Pi 5B**	**Jetson Nano**	**Desktop**
CPU	ARM Cortex-A76	ARM Cortex-A57	Intel Core i5-9600K
GPU	–	128-core Maxwell	NVIDIA RTX 2080Ti
RAM	8 GB	4 GB	32 GB

The Raspberry Pi 5, with no AI accelerator, has limited performance for deep learning tasks. To mitigate this, we used NCNN[2], which optimizes inference for CPU-based systems by using multi-threading. For Jetson Nano, which has a GPU, we utilized TensorRT which compiles the model to optimize the inference. These two edge devices highlight the trade-offs between processing power, cost, and real-time performance required for VRU detection in real-world vehicular environments. The desktop machine with an RTX 2080Ti GPU, while not intended for embedded deployment, serves as a reference with greater computational resources aligned with the capabilities of higher-end automotive computer resources.

All models were executed with frozen weights (i.e., using the pre-trained weights available in the open-source YOLO11 repository) and no additional fine-tuning or training. Inference outputs were evaluated across all platforms, ensuring consistent benchmarking for latency, throughput, and alert generation capabilities.

C. Performance Evaluation

In the following experiments, we evaluate the system's ability to generate accurate VRU alerts and real-time operation. The system performs binary classification to trigger blind spot alerts for the left or right filter lanes. Alerts are considered correct if triggered during the VRU's presence in the risk zone.

In the first experiment, we compare our system to a baseline that triggers alerts based only on VRU detection within the risk zone, without tracking and approach filtering (Section II-B). As no prior work directly targets rearview-based VRU alerting, we take this detection-only baseline as reference. To ensure fair comparison, both methods rely on the same YOLO11x model and are evaluated under identical conditions, on the Desktop platform.

TABLE II
COMPARISON BETWEEN THE BASELINE AND PROPOSED METHOD (DESKTOP, YOLOV11X).

Method	**Precision**	**Recall**	**F1-score**
Baseline	0.69	**0.98**	0.81
Proposed	**0.88**	0.94	**0.91**

The results in Table II show a significant precision improvement (+19 pp) with the proposed method, reducing false positives and providing more reliable alerts. Although recall drops slightly (-2 pp), this is due to missed VRUs whose trajectories didn't meet the alignment threshold θ_{min}. As future work we consider to explore training LSTM-based filters to enhance sensitivity to diverse motion patterns without increasing false alerts.

The trade-off between detection accuracy and system responsiveness is evaluated through comparisons across different YOLO11 models executed on the three distinct platforms: a Raspberry Pi 5, Jetson Nano, and a high-resource computing unit (named "Desktop"). Each device was tested with increasing model sizes—Nano (N), Small (S), Medium (M), Large (L), and X-Large (X)—highlighting how performance scales with model complexity and computational capacity.

Figure 4 shows the trade-off between latency and F1-score across model sizes evaluated on the Desktop. The latency representing the total per-frame processing time. This experiment evaluates the improvements of the proposed method. The Desktop platform was selected as it is the only one capable of running the largest YOLO11 models, enabling us to assess the maximum achievable performance.

The proposed method consistently outperforms the baseline in detection quality while maintaining comparable latency. Despite adding tracking and risk assessment, the average latency increase is only 0.8 ms for equivalent model sizes. This minor overhead is justified by the significant precision improvement, confirming the method's feasibility for real-time applications.

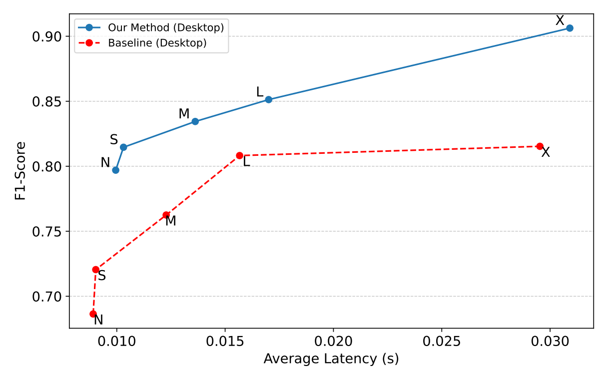

Fig. 4. Latency vs. F1-score for different model sizes comparing baseline and proposed method on the Desktop.

The cross-device benchmark results in Figure 5 show trade-offs between computational capacity, latency, and alert generation performance. As expected, the Desktop outperforms edge devices in both latency and F1-score, achieving 0.91 with the X-Large model at 31 ms latency. This aligns with the capabilities of high-end automotive GPUs like the NVIDIA DRIVE Orin series. Due to computational limitations, the Large and X-Large models could not be evaluated on edge devices as they exceeded their processing capacity.

Despite limited resources, both the Jetson Nano and Raspberry Pi 5 performed effectively with smaller models. As model size increased, memory and architectural constraints

[2]https://github.com/Tencent/ncnn

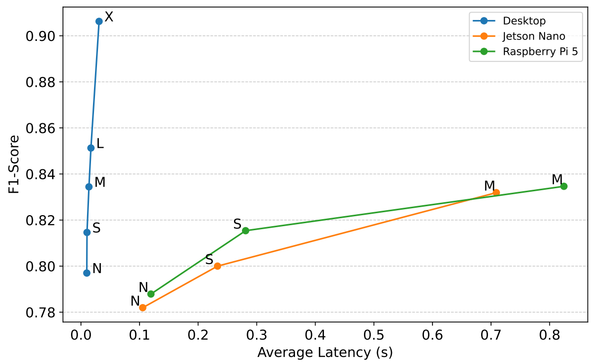

Fig. 5. Latency vs. F1-score for different model sizes across different devices.

hindered real-time inference, especially with the medium model (M), which exceeded 700 ms latency. The Jetson Nano slightly underperformed compared to the Raspberry Pi 5, primarily due to TensorRT quantization. While this optimization improves latency, it can reduce detection accuracy by introducing numerical imprecision. Nonetheless, both devices achieved F1-scores of 0.81-0.83 with the small model, demonstrating their viability for embedded deployment.

To contextualize system latency, we compare it with average driver reaction times to VRUs and blind spot hazards. Typical perception-response times are around 1 second [13], but delays are often longer when hazards are perceived via mirrors [14]. The system is designed to emit alerts faster than human response times, ensuring timely warnings that outperform human perception, even in real-world blind spot scenarios.

To assess the impact of our method on runtime execution, we break down the latency components for the Medium YOLOv11 model across devices, shown in Table III. While the Desktop platform has the lowest latency, the Jetson Nano and Raspberry Pi 5 experience higher latencies due to the limited resources required to run the object detection model. Despite the added complexity of tracking and risk assessment, the increase in latency is minimal—under 1 ms across all devices—ensuring the system maintains real-time performance without significant penalty.

TABLE III
AVERAGE LATENCY OF DIFFERENT DEVICES FOR THE MEDIUM MODEL.

Component	Desktop (s)	Jetson Nano (s)	Raspberry Pi 5 (s)
Detection	0.01220	0.70380	0.82207
Tracking	0.00142	0.00935	0.00751
Risk Assess.	0.00003	0.00013	0.00006
Total Latency	**0.01365**	**0.71328**	**0.82439**

IV. CONCLUSION

In this work, we proposed a proactive VRU alert system for risk zones at the rear of the vehicle, leveraging object detection, multi-object tracking, and risk assessment to generate accurate real-time alerts. The system performed well across different edge devices, even with their limited resources, maintaining latency below driver reaction times. Our experiments demonstrated that the proposed method outperforms a detection-only approach in precision and F1-score. Despite the added complexity, the system remains efficient and suitable for embedded deployment in ADAS applications. Future work can focus on training detection models tailored to the vehicular context observed by rear cameras, improving the handling of diverse VRU motion patterns through techniques like LSTMs, and exploring multi-sensor fusion to enhance system performance in complex and dynamic traffic environments.

ACKNOWLEDGMENT

This work was supported by CAPES (Finance Code 001), CNPq, FAPERJ (E-26/204.562/2024 and E-26/204.122/2024), FAPESP (grants 23/00673-7 and 23/00811-0), and Development and Research Foundation - Fundep - Rota 2030, and our partners Stellantis and Mobway.

REFERENCES

[1] P. Mannion, “Vulnerable road user detection: state-of-the-art and open challenges,” *arXiv preprint arXiv:1902.03601*, 2019.
[2] W. H. Organization, *Global Status Report on Road Safety 2023*. Geneva, Switzerland: World Health Organization, 2023.
[3] Ministry of Road Transport and Highways, “Road accidents in India 2022,” 2022, accessed: 2025-03-25. [Online]. Available: https://morth.nic.in/sites/default/files/RA_2022_30_Oct.pdf
[4] T. de Borba, O. Vaculín, H. Marzbani, and R. N. Jazar, “Increasing safety of vulnerable road users in scenarios with occlusion: A collaborative approach for smart infrastructures and automated vehicles,” *IEEE Access*, vol. 13, pp. 8851–8885, 2025.
[5] M. Malinverno, G. Avino, C. Casetti, C.-F. Chiasserini, F. Malandrino, and S. Scarpina, “Performance analysis of C-V2I-based automotive collision avoidance,” in *IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2018, pp. 1–9.
[6] A. A. Alaqeel, A. Alburadi, A. Y. Nashashibi, K. Sarabandi, and H. Shaman, “Detection and identification of pedestrians and bicyclists using J-band automotive radars,” in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2023, pp. 6157–6159.
[7] B. Wu, A. Wan, X. Yue, and K. Keutzer, “SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1887–1893.
[8] K. Van Beeck and T. Goedemé, “The automatic blind spot camera: A vision-based active alarm system,” in *European Conference on Computer Vision (ECCV) Workshops*. Springer, Oct. 2016, pp. 122–135.
[9] L. Han, P. Zheng, H. Li, J. Chen, Z. Hua, and Z. Zhang, “A novel early warning strategy for right-turning blind zone based on vulnerable road users detection,” *Neural Computing and Applications*, pp. 1–20, 2022.
[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.
[11] G. Jocher and J. Qiu, “Ultralytics YOLO11,” 2024. [Online]. Available: https://github.com/ultralytics/ultralytics
[12] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, “ByteTrack: Multi-object tracking by associating every detection box,” in *European Conference on Computer Vision (ECCV)*, 2022, pp. 1–21.
[13] T. J. Triggs and W. G. Harris, *Reaction time of drivers to road stimuli*. National Academies of Sciences, Engineering, and Medicine, Transportation Research Board (TRB), 1982, no. HFR-12 Monograph.
[14] V. Beanland, M. G. Lenné, and G. Underwood, “Looking forward to safer HGVs: The impact of mirrors on driver reaction times,” *Accident Analysis & Prevention*, vol. 106, pp. 179–186, 2017.