

Enhancing Functional Reliability of Autonomous Vehicle Safety Monitoring in Curves

Junnan Pan^a, Mohak Mansharamani^b, Prodromos Sotiriadis^c, Vladislav Nenchev^d and Ferdinand Englberger^e

Department of Electrical and Computer Engineering, University of the Bundeswehr, Werner-Heisenberg-Weg 39, Neubiberg, Germany

Keywords: Near-Field Monitoring, Obstacle Avoidance, Motion Corridor Modeling, Active Safety, Autonomous Vehicles.

Abstract: Active safety systems that operate alongside a higher-level autonomy stack and perform autonomous braking require reliable obstacle detection within the vehicle's motion corridor. Many runtime-constrained monitoring approaches simplify road geometry, producing blind spots on high-curvature roads where obstacles could not be detected. In this paper, we eliminate these blind spots by aligning the monitoring region with a locally framed, predicted motion corridor. We compute a predicted corridor boundary and filter incoming lidar point clouds in real time using an azimuth-sorted visible boundary and a moving window. In a case study, our method avoids curvature-induced blind spots while maintaining performance on straight segments.

1 INTRODUCTION

Recent advances in (Artificial Intelligence (AI)-enhanced) autonomous driving have greatly improved perception, prediction, and decision-making (Muhammad et al., 2020; Alahmed et al., 2024). However, hardware limitations, Advanced Driver Assistance System (ADAS) software stack complexity, or the black box nature of AI components create challenges for functional safety and verifiability. A complementary, traditionally implemented active safety layer – designed to operate largely independently of a High-Level Autonomy System (HLAS) – provides transparent, verifiable behavior and supports the HLAS in handling complex driving scenarios, thereby increasing overall reliability and safety.

Near-field active safety systems on embedded hardware can detect and respond to hazards inside the vehicle's predicted motion corridor even when the HLAS is late or fails to act. However, to meet computational and latency constraints, they often simplify road geometry, which can leave certain obstacles within the motion corridor undetected.

In this work, we propose Predictive Motion Corridor Monitoring (PMCM) that ensures reliable obstacle detection and reaction on curved roads. Our contributions are:

1. Proposing a driving corridor boundary in the Frenet frame that accurately covers roads with high curvatures;
2. Developing a lightweight filter using an azimuth-ordered visible boundary and a triangle-window;
3. Evaluating the approach in a case study showing reduced False Negative (FN) compared to baselines.

2 RELATED WORK

Active safety systems such as Autonomous Emergency Braking System (AEBS) use sensors to perceive the environment of the vehicle, evaluate risks, and apply braking action preemptively (Yang et al., 2022). Modern AEBS account for road slope and friction coefficients (Hwang and Choi, 2018; Sevil et al., 2019), potential rear-end collisions (Shin et al., 2019), or employ machine learning techniques such as a fuzzy neural network trained on experienced driver braking data to adaptively assess collision risk and determine warning and braking actions (Yang et al., 2019). However, most of these systems assume that

^a  <https://orcid.org/0009-0007-7853-380X>

^b  <https://orcid.org/0009-0000-0482-622X>

^c  <https://orcid.org/0009-0002-0089-8933>

^d  <https://orcid.org/0000-0002-9261-2746>

^e  <https://orcid.org/0009-0000-2616-6170>

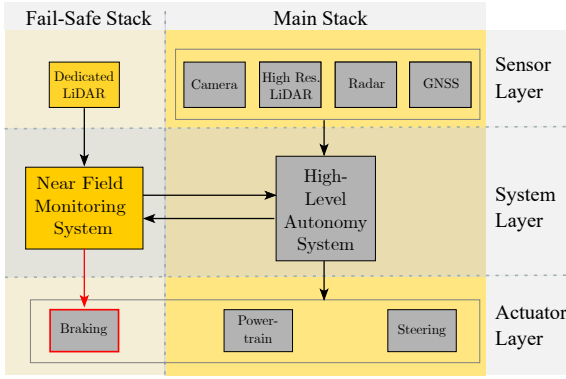


Figure 1: System architecture.

a driver is present and focus on longitudinal-motion-related reaction. This results in suboptimal performance on sharp curves, emphasizing the need for precise motion corridor modeling.

Motion corridor modeling typically relies on an overapproximation of the drivable areas using reachable sets (Söntges and Althoff, 2017; Manzingier et al., 2020). Recent approaches derive curvature-bounded corridor envelopes for refinement in optimal control, ensuring collision avoidance through sequential convex programming (Schäfer et al., 2021). However, reachability-based approaches often suffer from an overly conservative overapproximation in high-curvature scenarios and high computational costs for real-time use.

Light Detection and Ranging Sensor (LiDAR)-based perception typically uses iterative segmentation filtering to extract background from organized scans (Zermas et al., 2017) or applying octree and ray casting (Wu et al., 2024). Other adaptive methods employ k-d trees and PCA but suffer from high latency (Zheng et al., 2022). For near-field monitoring, BEV representations are handy in enabling low-latency 3D detection via efficient 2D convolutions (Mohapatra et al., 2021), and boundary detection although they struggle with occlusions (Sun et al., 2019). Such methods often suffer from high computational burden due to repeated traversals or variable runtime, limiting their use in embedded systems.

Using contracts and model checking, longitudinal (Nenchev et al., 2025a) and lateral (Nenchev et al., 2025b) controllers can be verified offline, offering code-level guarantees for safe automated driving vehicle operation. A lightweight monitor (Nenchev and Sotiriadis, 2025) can detect navigation failures early with low false alarms, motivating our real-time, low-overhead failure detection for near-field emergency braking scenarios.

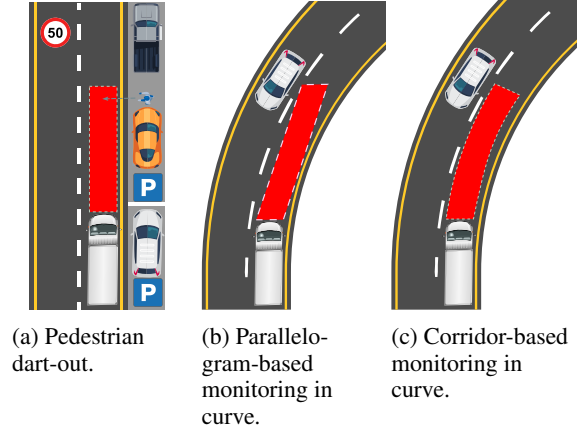


Figure 2: Typical driving scenarios with different monitoring corridors (marked in red).

3 PROBLEM STATEMENT

3.1 System Description

To improve vehicle safety, a Near-Field Monitoring System MORE (NeFiMMORE) has been developed to monitor a vehicle's immediate surroundings using dedicated 3D LiDAR sensors as part of the Munich Mobility Research Campus (MORE) project.

The primary challenge is to design an embedded solution that operates within limited computational resources while maintaining the required level of active safety and enabling seamless integration with the LiDAR hardware. An extended hardware description of NeFiMMORE system is provided in our previous work (Pan et al., 2023).

The NeFiMMORE forms part of a broader autonomous vehicle platform, within the framework of the MORE project. The complete system architecture is illustrated in Figure 1. As this figure shows, NeFiMMORE operates with the HLAS, which has been engineered to ensure that the vehicle adheres to a collision-free trajectory, enabling safe navigation along the designated route. The HLAS relies on a high-performance computing platform as its core component; however, in certain situations, it may fail to respond promptly enough to avoid a collision. Such cases may arise due to blind spots, computational overload, or unexpected events, such as a pedestrian suddenly appearing from behind an obstacle.

Our Fail-Safe stack uses dedicated sensors and microcontrollers. Guaranteeing collision avoidance by the HLAS can be hindered by sensor blind spots, computational overload, or faulty functionality. To mitigate these cases, the embedded safety system provides streamlined functions: monitoring, emergency

braking, and collision avoidance/mitigation. The system uses low-mounted 3D LiDAR sensors positioned around the vehicle to monitor the near-field environment, complementing the often roof-mounted sensors used by the HLAS. It consumes the HLAS-provided trajectory and computes a predicted motion corridor that depends on the trajectory geometry and vehicle speed. When a collision is detected by the system within the corridor, the system automatically triggers emergency braking. Representative scenario is shown in Figure 2a.

3.2 Problem Formulation

Although the existing parallelogram-based monitoring approach in Figure 2b offers notable benefits for deeply embedded systems by reducing computational cost, the under-approximation yields coverage gaps that may leave obstacles undetected. Therefore, we have to compute a predicted motion corridor or monitoring that conforms to the planned trajectory and covers as much of the drivable area as possible while remaining feasible for deeply embedded implementation.

Specifically, given a trajectory and the current speed and curvature profiles, we determine a predicted motion corridor that (i) provides contiguous coverage of the motion corridor across curvature regimes, (ii) enables efficient geometrical and window filtering.

4 METHODOLOGY

The proposed safety monitoring approach employs a corridor-based LiDAR point filtering along the vehicle's planned motion trajectory, provided by a HLAS. The methodology consists of two core stages: (i) constructing a motion corridor using sliding-window extrema of the vehicle's corner tracks, and (ii) filtering LiDAR points in real time through an azimuth-sorted visible boundary and triangle window. This pipeline ensures accurate near-field coverage with a low computational cost, making it suitable for real-time deployment on embedded microcontrollers.

4.1 Motion Corridor Construction

Urban maneuvers often involve smaller turning radii than highway driving. Accordingly, a curvature-aware mathematical representation is required to model the geometric boundary of the feasible driving corridor.

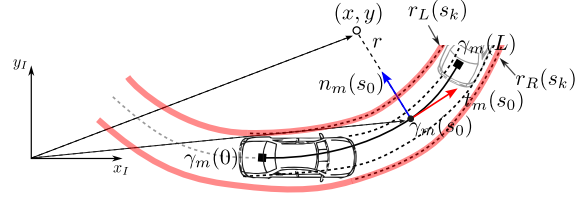


Figure 3: A predictive motion curve $\gamma_m(s)$ with a bold black line segment on a Curvilinear coordinate has a beginning point $\gamma_m(0)$ and an ending point $\gamma_m(L)$.

We introduce the Frenet coordinate system along the driving corridor to better represent the morphological corridor's coordinates in this paper. As shown in Figure 3, the curve $\gamma_m(s)$ represents the predicted reference track of the vehicle, which is based on the global coordinate system (x_I, y_I) . Let the reference track (in our case, it is the track of the middle point of the rear axle) be a C^2 planar curve $\gamma_m : [0, L] \rightarrow \mathbb{R}^2$ parametrized with respect to its arclength s with

$$\gamma_m(s) := \begin{pmatrix} x_m(s) \\ y_m(s) \end{pmatrix}.$$

Let the vector $n_m(s)$ denote the normal (to the left) vector

$$n_m(s) = \begin{pmatrix} -y'_m(s) \\ x'_m(s) \end{pmatrix},$$

which is perpendicular to the unit tangent vector $t_m(s) = \gamma'_m(s)$ at $\gamma_m(s)$. Then, any point in Cartesian coordinate system is given by $\mathbf{p}(x, y) = \gamma_m(s) + n_m(s)r$, using coordinates $s \in [0, L]$ and $r \in \mathbb{R}$.

The vehicle is modeled as a rectangle. The body set on the vehicle is represented as

$$\mathcal{R} = \{(a, b) \in \mathbb{R}^2 : -l_r \leq a \leq l_f, -\frac{w}{2} \leq b \leq \frac{w}{2}\},$$

where (a, b) is the local vehicle Cartesian coordinate on the point $\gamma_m(s)$ (forward longitudinal and left lateral taken as positive), and l_r, l_f denote the rearward and forward overhang from the baselink and w denotes the outer width of the vehicle. At the baselink pose s_0 , a body point $(a, b) \in \mathcal{R}$ maps to world coordinates via $\mathbf{p}_{car}(s_0, x, y) = \gamma_m(s_0) + at_m(s_0) + bn_m(s_0)$. Then, the corridor set is $S = \bigcup_{s_0 \in [0, L]} (\gamma_m(s_0) + \mathcal{R}_{s_0})$. The goal is to seek outer boundary $r_L(s)$ and $r_R(s)$ such that

$$S \subseteq \{(s, r) : r_R(s) \leq r \leq r_L(s)\}.$$

In practice, we use the tracks of the four corner points of the vehicle to build the corridor:

$$\{(s_i^{lf}, r_i^{lf})\}, \{(s_i^{lr}, r_i^{lr})\}, \{(s_i^{rf}, r_i^{rf})\}, \{(s_i^{rr}, r_i^{rr})\},$$

where the superscripts lf, lr, rf, and rr denote the corner points of the rectangular vehicle (left front, left rear, right front, right rear).

At a reference position s_0 , the vehicle spans the arclength interval $[s_0 - l_r, s_0 + l_f]$. Therefore, the left/right boundary point at arc length $s = s_0$ can be either the left-front/right-front corner at some $s_0 \leq s \leq s_0 + l_f$ or the left-rear/right-rear corner at some $s_0 - l_r \leq s \leq s_0$. On this given reference track $\gamma_m(s)$, we define a sliding window with the forward window interval $[s, s + l_f]$ and the backward window interval $[s - l_r, s]$. The sliding extrema lie within this window.

We sample a common grid $s_k = k\Delta s$, where the Δs is the sample step length. The forward window length in the sampled grid is $F = \lceil l_f/\Delta s \rceil$. The backward window length in the sampled grid is $R = \lceil l_r/\Delta s \rceil$. Compute

$$r_L(s_k) = \max \left\{ \max_{j \in [k, k+F]} r_{lf}(s_j), \max_{j \in [k-R, k]} r_{lr}(s_j) \right\},$$

$$r_R(s_k) = \min \left\{ \min_{j \in [k, k+F]} r_{rf}(s_j), \min_{j \in [k-R, k]} r_{rr}(s_j) \right\}.$$

Each sliding max/min has time complexity $O(N)$. The overall complexity scales linearly with the number of grid points. An example for the final corridor boundary result along the reference track is illustrated with the red line in Figure 3.

4.2 Corridor Filtering

To enable a resource-limited, deeply embedded implementation, the initial monitoring boundary should be further optimized to reduce unnecessary monitoring. Our monitoring boundary filtering is based on the following two stages.

4.2.1 Safety Corridor Construction

An effective monitoring boundary is defined to satisfy three conditions: (1) compliance with the braking-distance constraint, (2) absence of occluded discrete points within the given corridor, (3) and full utilization of the sensor's Field-of-View (FOV), in this case, a LiDAR.

The first step in this stage is *boundary preconstruction*, where the effective monitoring boundary is derived from the previously established motion corridor described in Section 4.1. The LiDAR used on our experimental vehicle is mounted at the vehicle's front center; therefore, only the boundary points located in the forward region need to be considered:

$$S = \{\mathcal{P}_i^r \mid x \geq 0, i \in \mathbb{N}_0\} \cup \{\mathcal{P}_j^l \mid x \geq 0, j \in \mathbb{N}_0\},$$

where $\mathcal{P}_i^r$ and $\mathcal{P}_j^l$ denote the discrete point sets along the right and left corridor boundaries, respectively. The employed reference Cartesian coordinate system, illustrated in Figure 4, corresponds to the LiDAR's local coordinate frame, which differs from

both the global and Frenet coordinate systems. Using the local coordinate system can facilitate boundary construction. From the LiDAR's perspective, the portion of the corridor boundary behind it, i.e., $x < 0$, can be ignored. When no discrete points in $\mathcal{P}_i^r$ and $\mathcal{P}_j^l$ exist at $x = 0$, points are linearly interpolated at $x = 0$ on both sides to ensure a closed boundary. The segment distance is defined as $\Delta_k := \|(x_{k+1}, y_{k+1}) - (x_k, y_k)\|_2, k \in \mathbb{N}_0$, which represents the length of each segment along the predicted path of the front LiDAR. The cumulative path length until the stopping distance is $L_s = \sum_{k=0}^{s-1} \Delta_k$, where s is the number of points up to the stopping point. If the braking distance d_b lies within the driving corridor, it satisfies $L_s \geq d_b$. The stopping point $\mathbf{v}_m^s$ is interpolated between $\mathbf{v}_s$ and $\mathbf{v}_{s+1}$ such that $L_s = d_b$. By projecting the stopping point $\mathbf{v}_m^s$ onto the right and left boundaries $\mathcal{P}_i^r, \mathcal{P}_j^l$, the corresponding stopping points $\mathbf{v}_r^s$ and $\mathbf{v}_l^s$ are obtained, as illustrated in Figure 4a. Hence, the prefinding process outputs the following boundary set:

$$S^p = S^r \cup S^l = \{\mathcal{P}_i^r \mid x \geq 0, i \in \{0, 1, 2, \dots, s-1\}\} \\ \cup \{\mathcal{P}_j^l \mid x \geq 0, j \in \{0, 1, 2, \dots, s-1\}\}.$$

An example for this step is illustrated in Figure 4a.

The next step is *Visible reduction*. The objective in this step is to identify all visible points from the sensor's viewpoint on the boundary established in the previous step. As the boundary points in S^r and S^l are discretely ordered along the corridor with indices $i \in \{0, 1, 2, \dots, s-1\}$, two initial sets are defined: $\hat{S}^r = \emptyset, \hat{S}^l = \emptyset$, which store the current extreme points $\mathbf{v}_i \mid \alpha_i = \text{atan2}(y_i, x_i) + \pi/2$, where α_i is the azimuth of the current point $\mathbf{v} = (x, y)$ on the boundary referring to the viewpoint at the LiDAR. For the right boundary S^r , it is the point with the current maximum azimuth in the set $\hat{S}^r$, and for the left boundary S^l , it is the point with the current minimum azimuth in the set $\hat{S}^l$. In our example Figure 4b, $\mathbf{v}_{11}^r, \mathbf{v}_{12}^r$ and points behind $\mathbf{v}_{13}^r$ are occluded by other boundary points on its boundary. Hence, those points are excluded from the $\hat{S}^r, \hat{S}^l$.

In addition to the self-occlusion within each corridor, there could also exist points that are occluded by points on the opposite boundary. Consequently, we need to remove those points from the visible boundary extraction. This step is referred to as *Occlusion filtering*. In our example Figure 4c, $\mathbf{v}_{17}^r$ is occluded by the point $\mathbf{v}_{13}^l$. Those points are invalid within an effective monitoring boundary and are trimmed from $\hat{S}^r$ and $\hat{S}^l$. The visible sets after the trim are

$$\hat{S}_{\text{trim}}^r = \{\mathbf{v}_i^r \in S^r \mid \forall \alpha_j > \alpha_i\},$$

$$\hat{S}_{\text{trim}}^l = \{\mathbf{v}_j^l \in S^l \mid \forall \alpha_j > \alpha_i\}.$$

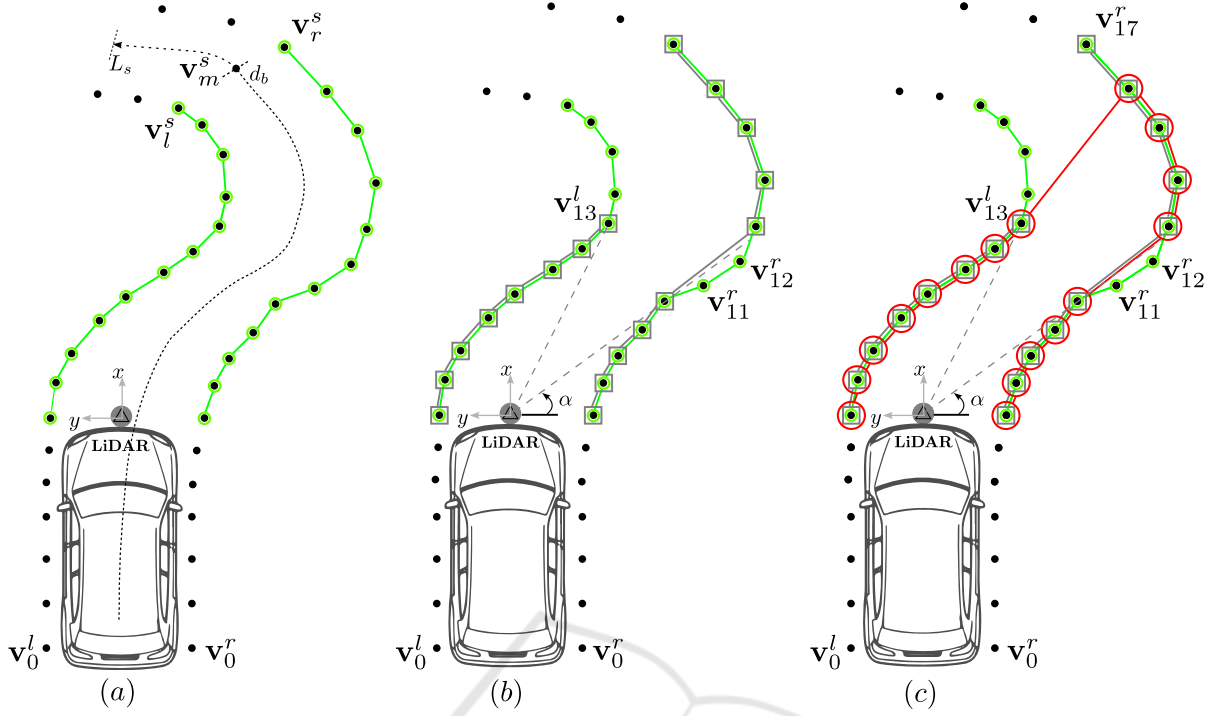


Figure 4: Boundary computation: (a) *Boundary preconstruction*: The discrete black boundary construction is computed from the motion corridor. The monitoring boundary is limited to the braking distance, illustrated by green circles connected by line segments. Points located behind the ego LiDAR are ignored. (b) *Visible reduction*: All visible points from the LiDAR viewpoint are selected along both left and right discrete corridor boundaries (denoted by gray squares). The dotted lines indicate the azimuthal rays from the ego LiDAR to boundary point candidates, used in the visibility check. Points located behind $\mathbf{v}_{13}^l$, or points like $\mathbf{v}_{11}^r$, $\mathbf{v}_{12}^r$ are occluded by preceding points and are excluded from the current boundary. (c) *Occlusion filtering*: Boundary points occluded by the opposite corridor are removed. For example, $\mathbf{v}_{17}^r$ is occluded by $\mathbf{v}_{13}^l$ and is therefore excluded from the final boundary, shown in red circles connected by line segments.

Define the global visible boundary sequence $(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M)$ as the elements of the union, which is the azimuth-sorted enumeration $(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M) := \text{sort}_\alpha(\hat{\mathcal{S}}_{\text{trim}}^r \cup \hat{\mathcal{S}}_{\text{trim}}^l)$, such that $\alpha(\mathbf{v}_1) < \alpha(\mathbf{v}_2) < \dots < \alpha(\mathbf{v}_M)$.

4.2.2 Window Points Filter

In the previous process, a global visible boundary sequence with monotonically increasing azimuth is computed. This boundary constrains further filtering LiDAR points within the effective monitoring range. We form an open polygonal chain by connecting successive vertices $\mathbf{v}_i \mathbf{v}_{i+1}$, $i = 0, 1, \dots, M-1$. All vertices $(\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_{M-1})$ of this polygon are visible to the LiDAR. The edge of this open-polygon $\mathbf{v}_i \mathbf{v}_{i+1}$ serves as a local filtering window to filter out-of-range LiDAR points.

At the beginning of each traversal in the $\mathbf{v}_i \mathbf{v}_{i+1}$, we project a candidate 3D LiDAR point onto the x - y plane. The point $\mathbf{p}_j$ is retained if there exists i such

that

$$\alpha(\mathbf{v}_i) \leq \alpha(\mathbf{p}_j) < \alpha(\mathbf{v}_{i+1}), \quad \mathbf{p}_j \in \overline{\Delta(\mathbf{o}, \mathbf{v}_i, \mathbf{v}_{i+1})}, \quad (1)$$

where $\overline{\Delta(\mathbf{o}, \mathbf{v}_i, \mathbf{v}_{i+1})}$ denotes the closed triangle. Membership in (1) is tested by cross-product sign consistency:

$$\begin{aligned} s_1 &= (\mathbf{v}_{i+1} - \mathbf{o}) \times (\mathbf{p}_j - \mathbf{o}), \\ s_2 &= (\mathbf{v}_{i+1} - \mathbf{v}_i) \times (\mathbf{p}_j - \mathbf{v}_i), \\ s_3 &= (\mathbf{o} - \mathbf{v}_{i+1}) \times (\mathbf{p}_j - \mathbf{v}_{i+1}). \end{aligned}$$

The condition

$$(s_1 \geq 0 \wedge s_2 \geq 0 \wedge s_3 \geq 0) \vee (s_1 \leq 0 \wedge s_2 \leq 0 \wedge s_3 \leq 0) \quad (2)$$

is necessary for (1). In 2D, all oriented areas must have the same sign. For the example in Figure 5, the LiDAR point $\mathbf{p}_j$ is within the triangle $\overline{\Delta(\mathbf{o}, \mathbf{v}_i, \mathbf{v}_{i+1})}$ and thus, (2) is satisfied.

Checking (2) can be expensive for large point clouds. Thus, we apply pre-filtering by checking only

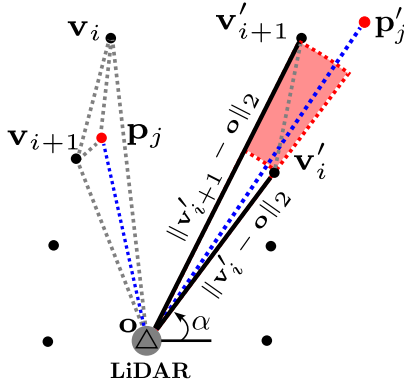


Figure 5: Schematic for window points filter with point p_j inside and point p'_j outside the boundary range.

points that satisfy

$$\|p_j - o\|_2 \leq \max \{ \|v_i - o\|_2, \|v_{i+1} - o\|_2 \},$$

which implies that the candidate point's distance on the x - y plane must be within the maximum distance between the two adjacent edges with vertex o . For the point p'_j in Figure 5, the distance $\|p'_j - o\|_2$ does not satisfy the above requirement for the successive vertices v'_i, v'_{i+1} . In this work, we evaluate the proposed filtering algorithm on preprocessed point-cloud data to assess its effectiveness; however, it can also operate directly on raw LiDAR measurements, i.e., on the ordered packet stream.

5 CASE STUDY

This section presents the evaluation of monitoring coverage and detection reliability of the proposed PMCM method. The experiments were conducted in a simulation environment consistent with our prior validation work, focusing on near-field obstacle-detection performance across varying road curvatures.

Inspired by prior studies on near-field monitoring in a validated simulation environment (Pan et al., 2025), we employ Gazebo to simulate the proposed approach with a validated sensor model of the real Velodyne VLP-16. We conducted various controlled collision-driving scenarios to evaluate the monitoring performance enhancement of the proposed approaches in the current research work.

A cylinder obstacle with a radius of 0.35 m and a height of 0.6 m was positioned at the edge of the driving corridor. The vehicle maintained a constant velocity on a circular or a straight road in each scenario, ranging from 1 km/h to 30 km/h in increments of 2 km/h. The designed road radius in the simulation experiments was as follows:

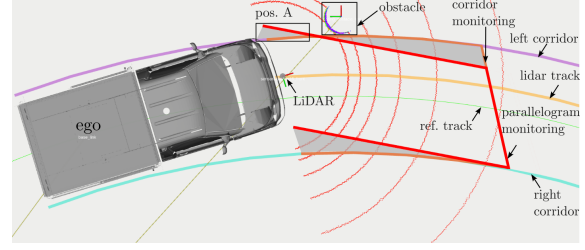


Figure 6: Bird's eye view of the collision-driving in the simulation.

$\{6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, \infty\}$ m. The last parameter with an infinite radius indicated a straight road. Figure 6 presents one of the simulated collision cases. We applied the same vehicle geometry dimensions as our experimental vehicle to the simulation vehicle. The position of the detected obstacle in the simulations was also recorded to verify whether the applied braking could avoid the collision. In some cases, when the obstacles were too late to be discovered, the emergency braking maneuver could not prevent the collision.

The parallelogram monitoring strategy developed in (Pan et al., 2025) was used as a baseline in the experiments. We also calculated the ratio of the parallelogram's area to the motion corridor monitoring area for each case, which is not presented in the comparison table. This parameter provides a quantitative representation of the parallelogram's loss area for each driving radius. In addition, a simplified variant for corridor filtering, named two-point corridor filtering, was implemented for comparison. This method constructs the monitoring boundary using only the two rear corner points of the vehicle as lateral references along the predicted path, forming a simplified corridor that approximates the vehicle's motion corridor.

The result of this simulation comparison in Figure 1 suggests that both the parallelogram and PMCM strategy achieve 0% False Negative Rate (FNR). However, a notable difference was observed in the resulting Collision Rate (CR). The CR indicates the number of collisions, even when braking is applied. The parallelogram monitoring approach had a significantly higher CR than the PMCM approach. This suggests that certain obstacles were detected too late for the vehicle to brake successfully. It was observed in the metadata, by recording the detected position of the obstacle, that the obstacle was observed when it reached the *pos. A* in Figure 6, which is within the braking distance d_b .

Both the two-point corridor filter and PMCM attained a low CR in the conducted experiment series, as outlined in Table 1. This finding indicates that both monitoring strategies can successfully detect all ob-

Table 1: Comparison of FNR, True Positive Rate (TPR), and CR between different monitoring filtering strategies.

Measures	Parallelogram Filtering	Two-point Corridor Filtering	PMCM
FNR	0%	57.74%	0% ↓
TPR	100%	42.26%	100% ↑
CR	36.1%	1.03%	0% ↓

stacles within the monitored region across all test scenarios. However, the two-point approach has a higher FNR compared to the PMCM approach. It was observed that most negative cases occur at smaller road radii (radius within 16 m). This tendency arises because, as the road curvature increases, the effective monitoring area of the parallelogram filtering shrinks relative to the motion corridor. Therefore, an obstacle near the outer edge of the vehicle corridor may not fall within the parallelogram region. This leads to a missed detection. In contrast, for larger road radii, the parallelogram approximation provides a sufficiently accurate representation of the driving corridor. As the curvature decreases, the geometric deviation between the parallelogram and the true path corridor becomes marginal, and the monitoring coverage remains stable. The calculated area ratio between the parallelogram and the corridor regions increases with road radius. It approaches unity in straight-road scenarios, confirming that the parallelogram strategy maintains good approximation performance under low-curvature conditions. The reason for this discrepancy lies again in the difference in monitoring coverage. The two-point corridor filtering method uses only two lateral constraint points to define the monitoring region, limiting its ability to precisely track the vehicle’s path geometry. As a result, its adequate coverage is smaller than that of the PMCM, leaving unmonitored areas along the outer boundaries. Conversely, the PMCM method determines the monitoring area directly from the vehicle geometry, thereby ensuring comprehensive spatial coverage of the driving corridor. This extensive coverage enables earlier obstacle detection and effectively eliminates collision cases across all test scenarios.

To further verify the robustness and stability of the proposed algorithm, we tested the corridor monitoring on recorded campus driving data. A test driving snippet is represented in Figure 7. The dataset contains multiple urban-like driving scenarios. Unlike the controlled simulation cases, this reflects realistic sensor noise, ego-motion fluctuations, and localization drift. The result demonstrates stable, continuous updates to the monitoring boundaries along the predicted path. The monitoring area has been shown



Figure 7: Monitoring corridor based on the autonomous driving data.

to adapt effectively to varying turning radii and to maintain consistent spatial alignment with the vehicle corridor. These findings confirm the reliability of the proposed method when applied to real-world data, thereby validating its suitability for deployment in embedded near-field active safety systems.

6 CONCLUSION

In this work, we present a predictive motion corridor monitoring approach for emergency braking of an autonomous vehicle. We model the vehicle’s motion corridor in a Frenet coordinate frame and use an azimuth-ordered boundary filter with a triangle-window filter for effective corridor filtering on curved roads. Our case study confirms that the method reliably identifies obstacles within the whole predicted motion corridor. The method performs well on recorded real-world LiDAR data. Future work will involve evaluating the performance of the method in more challenging driving conditions.

ACKNOWLEDGEMENTS

This research is funded by dtcc.bw – Digitalization and Technology Research Center of the Bundeswehr, which we gratefully acknowledge. dtcc.bw is funded by the European Union – NextGenerationEU. The authors would also like to thank Matthias Gerdt and Thomas Rottmann for valuable discussions.

REFERENCES

- Alahmed, Y., Abadla, R., and Al Ansari, M. J. (2024). Enhancing safety in autonomous vehicles through advanced ai-driven perception and decision-making systems. In *2024 Fifth International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, pages 208–217. IEEE.

- Hwang, Y. and Choi, S. B. (2018). Adaptive collision avoidance using road friction information. *IEEE Transactions on Intelligent Transportation Systems*, 20(1):348–361.
- Manzinger, S., Pek, C., and Althoff, M. (2020). Using reachable sets for trajectory planning of automated vehicles. *IEEE Transactions on Intelligent Vehicles*, 6(2):232–248.
- Mohapatra, S., Yogamani, S., Gotzig, H., Milz, S., and Mader, P. (2021). Bevdetnet: Bird’s eye view lidar point cloud based real-time 3d object detection for autonomous driving. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2809–2815. IEEE.
- Muhammad, K., Ullah, A., Lloret, J., Del Ser, J., and De Albuquerque, V. H. C. (2020). Deep learning for safe autonomous driving: Current challenges and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4316–4336.
- Nenchev, V., Imrie, C., Gerasimou, S., and Calinescu, R. (2025a). Code-level safety verification for automated driving: A case study. In Platzer, A., Rozier, K. Y., Pradella, M., and Rossi, M., editors, *Formal Methods*, pages 356–372, Cham. Springer Nature Switzerland.
- Nenchev, V., Imrie, C., Gerasimou, S., and Calinescu, R. (2025b). Compositional code-level safety verification for automated driving controllers. *Journal of Systems and Software*, 230:112499.
- Nenchev, V. and Sotiriadis, P. (2025). Monitoring progress and failure in autonomous robot navigation: A case study. In *Runtime Verification: 25th International Conference, RV 2025, Graz, Austria, September 15–19, 2025, Proceedings*, page 317–335, Berlin, Heidelberg. Springer-Verlag.
- Pan, J., Sotiriadis, P., and Englberger, F. (2023). Collision Avoidance of Autonomous Driving at Low Speed in the Near Field of Vehicle. *International Journal of Emerging Engineering and Technology*, 2(1):57–62.
- Pan, J., Sotiriadis, P., Nenchev, V., and Englberger, F. (2025). Improving functional reliability of near-field monitoring for emergency braking in autonomous vehicles. *arXiv preprint arXiv:2507.15594*.
- Schäfer, L., Manzinger, S., and Althoff, M. (2021). Computation of solution spaces for optimization-based trajectory planning. *IEEE Transactions on Intelligent Vehicles*, 8(1):216–231.
- Sevil, A. O., Canevi, M., and Soylemez, M. T. (2019). Development of an adaptive autonomous emergency braking system based on road friction. In *2019 11th International Conference on Electrical and Electronics Engineering (ELECO)*, pages 815–819. IEEE.
- Shin, S.-G., Ahn, D.-R., Baek, Y.-S., and Lee, H.-K. (2019). Adaptive aeb control strategy for collision avoidance including rear vehicles. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2872–2878. IEEE.
- Söntges, S. and Althoff, M. (2017). Computing the drivable area of autonomous road vehicles in dynamic road scenes. *IEEE Transactions on Intelligent Transportation Systems*, 19(6):1855–1866.
- Sun, P., Zhao, X., Xu, Z., Wang, R., and Min, H. (2019). A 3d lidar data-based dedicated road boundary detection algorithm for autonomous vehicles. *Ieee Access*, 7:29623–29638.
- Wu, J., Zhuang, X., Tian, Y., Cheng, Z., and Liu, S. (2024). Real-time point cloud clustering algorithm based on roadside lidar. *IEEE Sensors Journal*, 24(7):10608–10619.
- Yang, L., Yang, Y., Wu, G., Zhao, X., Fang, S., Liao, X., Wang, R., and Zhang, M. (2022). A systematic review of autonomous emergency braking system: impact factor, technology, and performance evaluation. *Journal of advanced transportation*, 2022(1):1188089.
- Yang, W., Zhang, X., Lei, Q., and Cheng, X. (2019). Research on longitudinal active collision avoidance of autonomous emergency braking pedestrian system (aeb-p). *Sensors*, 19(21):4671.
- Zermas, D., Izzat, I., and Papanikolopoulos, N. (2017). Fast segmentation of 3d point clouds: A paradigm on lidar data for autonomous vehicle applications. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5067–5073. IEEE.
- Zheng, Z., Zha, B., Zhou, Y., Huang, J., Xuchen, Y., and Zhang, H. (2022). Single-stage adaptive multi-scale point cloud noise filtering algorithm based on feature information. *Remote sensing*, 14(2):367.