Enhancing Autonomous Vehicle Safety through Multi-Sensor Integration and Real-Time Perception-Based Control Systems

M.Tech Project-II report submitted to

Indian Institute of Technology Kharagpur

in partial fulfilment for the award of the degree of

Bachelor and Master of Technology

in

Civil Engineering

by

Abhiraj Rananajay Singh

(20CE33002)

Under the supervision of

Professor Bhargab Maitra and Professor Soumyajit Dey

Department of Civil Engineering

Indian Institute of Technology Kharagpur

Spring Semester, 2024-25

April 29, 2025

DECLARATION

I certify that

(a) The work contained in this report has been done by me under the guidance of my supervisor.

(b) The work has not been submitted to any other Institute for any degree or diploma.

(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Date: April 29, 2025 (Abhiraj Rananajay Singh)

Place: Kharagpur (20CE33002)

DEPARTMENT OF CIVIL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR
KHARAGPUR - 721302, INDIA

CERTIFICATE

This is to certify that the project report entitled "**Enhancing Autonomous Vehicle Safety through Multi-Sensor Integration and Real-Time Perception-Based Control Systems**" submitted by **Abhiraj Rananajay Singh** (Roll No. 20CE33002) to Indian Institute of Technology Kharagpur towards partial fulfilment of requirements for the award of degree of Bachelor and Master of Technology in Civil Engineering is a record of bona fide work carried out by him under my supervision and guidance during Spring Semester, 2024-25.

Professor Bhargab Maitra
Department of Civil Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721302, India

Professor Soumyajit Dey
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721302, India

DEPARTMENT OF CIVIL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

KHARAGPUR - 721302, INDIA

CERTIFICATE

This is to certify that the project report entitled "**Enhancing Autonomous Vehicle Safety through Multi-Sensor Integration and Real-Time Perception-Based Control Systems**" submitted by **Abhiraj Rananajay Singh** (Roll No. 20CE33002) to Indian Institute of Technology Kharagpur towards partial fulfilment of requirements for the award of degree of Bachelor and Master of Technology in Civil Engineering is a record of bona fide work carried out by him under my supervision and guidance during Spring Semester, 2024-25.

Professor Bhargab Maitra and Professor

Soumyajit Dey
Department of Civil Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721302, India

Date: April 29, 2025
Place: Kharagpur

Abstract

Autonomous-vehicle (AV) performance has advanced rapidly, yet a persistent gap remains between algorithmic innovation and demonstrable on-road robustness. This study tackles that gap by developing a *real-time perception–control framework* evaluated on a comprehensive Hardware-in-the-Loop (HIL) platform. In contrast to software-only validation, the proposed set-up couples IPG CarMaker/Xpack4 traffic scenarios with automotive-grade compute—an NVIDIA Jetson Orin for perception and a Texas Instruments F28379D micro-controller for actuation—to reproduce production-level timing, CAN bandwidth and sensor latencies.

The perception front-end performs camera–radar fusion: a 60 fps RGB stream is processed by YOLOv11x for multi-class object detection, while raw radar returns are synchronised and filtered to provide precise range–rate cues. A lightweight Multi-Layer Perceptron learns a mapping from fused image size to object distance. These estimates, together with confidence scores, are broadcast via standard CAN frames to Adaptive Cruise Control / Automatic Emergency Braking (ACC/AEB) algorithm. Controller logic dynamically adjusts time-gap, throttle and brake pressure as a function of perception confidence.

Index Terms—*Autonomous vehicles, hardware-in-the-loop, multi-sensor fusion, adaptive cruise control, automatic emergency braking, real-time systems, transportation safety.*

Acknowledgements

I wish to extend my heartfelt appreciation to my mentor, Professor Bhargab Maitra and Professor Soumyajit Dey. His unwavering support has been instrumental in my academic journey.

Additionally, I am deeply grateful to Sunandan Adhikary, who have been an invaluable source of guidance and have generously provided me with essential resources that have significantly enriched my research.

I am profoundly grateful to the esteemed faculty at the Indian Institute of Technology, Kharagpur. Their dedication to imparting knowledge through a diverse range of courses has laid a robust foundation for my research endeavors.

Lastly, my sincere thanks go out to my family and friends for their constant support and affection throughout my academic pursuits.

Contents

Declaration i

Certificate ii

Abstract iii

Acknowledgements iv

Contents v

1 Introduction 1
- 1.1 Background . . . 1
- 1.2 Motivation . . . 2
- 1.3 Objective . . . 3
- 1.4 Scope of Work . . . 4

2 Literature Survey 5
- 2.1 Multi-Sensor Perception and Fusion . . . 5
- 2.2 Resilient Perception–Control under Sensor Uncertainty . . . 6
- 2.3 Safety Assurance through Hardware-in-the-Loop Validation . . . 8
- 2.4 Research Gaps . . . 9

3 Methodology 10
- 3.1 Full System Integration and Simulation Setup . . . 11
- 3.2 Perception Module: Vision-Based Object Detection and Distance Estimation . . . 16
- 3.3 Sensor Fusion Strategy for Multi-Sensor Perception . . . 18
- 3.4 Embedded Controller: ACC and AEB Algorithm Design . . . 21
  - 3.4.1 Adaptive Cruise Control (ACC) Logic . . . 21
  - 3.4.2 Automatic Emergency Braking (AEB) Logic . . . 23

4 Results 25
- 4.1 MLP Distance Estimation Performance . . . 25
- 4.2 Jetson Orin Inference Performance . . . 25

4.3 Controller Validation (TTC and Reaction Time) 26
4.4 Sensor Fusion Fault Detection 27
4.5 Testing Scenarios 27
Scenario 1: Pedestrian Occlusion by a Bus. 27
Scenario 2: Parallel Vehicles on Highway. 28

5 Conclusion **29**
5.1 Challenges Faced 29
5.2 Future Work 30

List of Figures **30**

Bibliography **32**

Chapter 1

Introduction

1.1 Background

Transportation systems are fundamental to modern society, enabling the movement of people and goods. The effectiveness of these systems relies on four key pillars: vehicles, infrastructure (roads), users (drivers and pedestrians), and the environment (**?**). Autonomous vehicles represent a significant advancement in the vehicle pillar, promising to improve safety, efficiency, and accessibility (Anderson et al., 2016). Road-traffic fatalities have plateaued at an unacceptably high level worldwide, compelling regulators to mandate ever-stronger crash-avoidance technology. Today, self-driving cars, or autonomous vehicles, are undergoing rapid development, and some are already operating on public roads, e.g., self-driving taxis from Google's Waymo. Given the increasing dependency on AI pipelines in autonomous driving systems, there's a growing need to ensure the robustness of image-based AI pipelines. In India, the Ministry of Road Transport and Highways has mandated that all new passenger vehicles with over eight seats, buses, and trucks manufactured from April 2026 onward must be equipped with Advanced Emergency Braking Systems (AEBS) (The Times of India, 2024). Modern Advanced Driver-Assistance Systems (ADAS) employ an array of sensors and algorithms to perceive the environment and assist or automate driving tasks. Monocular cameras provide rich semantic context but suffer in low-light or weather-degraded scenes; automotive wave radar measures range and relative velocity robustly yet lacks object classification. Software-in-the-Loop (SIL)

assumes ideal timing and perfect interfaces, whereas real ECUs must respect cycle-accurate deadlines, limited compute bandwidth and noisy I/O. Studies demonstrate that integrating HIL methodologies early in development improves process efficiency by up to 50%, accelerating control system maturation while reducing errors (Bui et al., 2024).

1.2 Motivation

Vehicle safety has always been a paramount concern in transportation. Traditional vehicles rely heavily on human drivers for decision-making, which introduces the potential for human error (World Health Organization, 2018). Autonomous vehicles aim to reduce accidents caused by human error by leveraging advanced sensors and control systems. Despite these advancements, AVs face challenges related to sensor reliability, data processing, and vulnerability to attacks (Petit and Shladover, 2015). While SIL simulations are indispensable in early development, they often assume ideal conditions and unlimited computational resources. In practice, control algorithms might perform suboptimally when deployed on embedded hardware due to processing delays, network latencies, or sensor noise. For instance, a braking algorithm that works in a high-level simulation might fail to react in time on real hardware if the perception data arrives late or if computing the control law takes too long. Traditional control systems often struggle to manage high-dimensional sensory data, such as images or point clouds (**?**).

Learning-based methods offer potential solutions but may lack safety guarantees and struggle to generalize beyond specific scenarios. This research is driven by the need to evaluate and refine perception-based control algorithms under real-time constraints and with actual interfaces. By adopting a HIL methodology, we can uncover timing issues, interface mismatches, or implementation bugs early in the development cycle, ultimately producing a more reliable ACC/AEB system. Ensuring that the control decisions occur within strict deadlines is especially important for AEB, where milliseconds can determine whether a collision is avoided. Thus, a significant motivation is to demonstrate that a real-time HIL implementation of ACC/AEB can meet the responsiveness and reliability required for true safety-critical operation.

Another motivation is the pursuit of greater safety and robustness through multi-sensor redundancy and advanced perception. Single-sensor systems in autonomous driving have well-documented failure modes: cameras can be blinded by glare or poor lighting, and radar returns can be cluttered or miss stationary obstacles. Relying on one sensor type alone can leave dangerous gaps in the vehicle's understanding of the environment. By integrating both camera and radar data (and potentially other sensors), the system can cross-verify and compensate for individual sensor shortcomings. For example, in scenarios of heavy rain or fog, radar can continue to measure distances when optical methods falter; conversely, where radar might not distinguish a pedestrian on the road.

1.3 Objective

The **primary objective** of this dissertation is to *demonstrate, in real time, a multi-sensor perception–control pipeline for safer and more reliable autonomous vehicles using Hardware-in-the-Loop validation.* Four measurable targets translate this overarching goal into concrete success criteria:

1. **Real-Time Sensor Fusion** — Deploy a camera–radar fusion front-end (YOLOv8x + MLP range estimator) on an NVIDIA Jetson Orin and sustain an end-to-end perception latency $\leq$ 50 ms at $\geq$ 30 Hz during a continuous 60-minute HIL run.

2. **Perception-Based Longitudinal Control** — Implement Adaptive Cruise Control / Automatic Emergency Braking (ACC/AEB) on a TI F28379D ECU and maintain safe time-gap compliance ($\geq 1.8\,\mathrm{s}$) in $\geq$ 95% of IPG CarMaker highway and urban scenarios.

3. **Collision-Risk Mitigation** — Achieve at least a 30% reduction in rear-end-collision probability compared with a camera-only baseline across 150 curated scenarios, quantified via time-to-collision metrics.

4. **End-to-End HIL Validation** — Demonstrate closed-loop stability and deterministic CAN timing under worst-case sensor bandwidth, thereby confirming that the integrated perception and control stack meets automotive real-time constraints on production-grade hardware.

1.4 Scope of Work

The Scope of Work (SoW) defines the activities, deliverables and boundaries required to realise the objectives, ensuring stakeholder alignment and preventing scope creep.

Tasks and Deliverables

1. **Literature Synthesis** — Compile a survey of multi-sensor fusion, perception-based longitudinal control and HIL validation.

2. **System Architecture** — Specify hardware interfaces, CAN message sets, timing budgets and safety requirements for Jetson Orin F28379D IPG CarMaker.

3. **Perception Module** — Optimise YOLOv8x with TensorRT on the Jetson Orin GPU, achieving ≥ 60 fps and ≤ 15 ms single-frame latency; design an MLP distance regressor fused with radar data.

4. **Embedded Controller** — Develop ISO-26262-style ACC/AEB logic in C on the TI F28379D, parameterised to operate directly on fused distance and closing-velocity signals.

5. **HIL Integration** — Couple IPG CarMaker/Xpack4 to the physical ECUs, streaming time-stamped camera frames and radar point clouds in real time; implement scenario and sensor-degradation scripting.

6. **Validation Campaign** — Execute 150 predefined scenarios, log perception latency, controller response, safety and comfort metrics; perform statistical analysis versus baseline systems.

Chapter 2

Literature Survey

This chapter reviews prior work in three tightly coupled areas: **(i) multi-sensor perception and fusion**, **(ii) resilient perception–control under sensor uncertainty**, and **(iii) system-level safety assurance through Hardware-in-the-Loop (HIL) validation**. Each section synthesises 4–5 influential studies into a coherent narrative and concludes with open gaps driving the present research.

2.1 Multi-Sensor Perception and Fusion

The past decade has seen an unmistakable shift from single-sensor pipelines to deeply integrated, learning-based fusion architectures. Early camera-only detectors such as YOLO demonstrated high accuracy in benign daylight, yet performance degraded sharply in glare, rain, or fog Redmon et al. (2016). Recognising the complementary physics of active sensors, researchers began combining radar's penetrative range with LiDAR's dense geometry and the rich semantics of RGB imagery. A comprehensive survey by Almutairi and Barnawi documents this progression, emphasising that no single modality meets the full Operational Design Domain (ODD) of a highway-speed vehicle and arguing that fusion is now indispensable to achieve centimetre-level localisation and reliable classification across weather and lighting conditions **?**.

Learning-based fusion strategies extend beyond naïve concatenation of sensor features. Caesar *et al.* introduced the *nuScenes* benchmark, providing time-synchronised camera, LiDAR and radar data, which catalysed work on spatio-temporal attention mechanisms that weigh sensors according to context Caesar et al. (2020). Building on that resource, Vinoth and Sasikumar employ a Deep-Q-Network to re-weight camera, LiDAR and long-wave infrared cues on the fly, cutting identity-switch errors by 25 % under heavy rain and fog relative to fixed-weight fusion Vinoth and Sasikumar (2024). Parallel advances in voxel-based encoders show that fusing LiDAR voxels with image features boosts 3-D mean Average Precision by 6–8 pp on the KITTI benchmark compared with either modality alone **?**. More recently, the Cross-Modal Transformer of Chen *et al.* learns global context across all sensors, producing state-of-the-art detection at 30 Hz on an embedded GPU, thereby satisfying real-time latency constraints Chen et al. (2022).

Yet accuracy metrics alone do not guarantee deployment readiness. Kochanthara *et al.* performed a requirement-centric audit of the Apollo perception stack and found that while classical sensor redundancy requirements were satisfied, nearly one-third of learning-centric safety requirements—such as graceful degradation when LiDAR density falls below a threshold—were missing explicit design evidence Kochanthara et al. (2024). This gap highlights two pressing needs: first, fusion blocks must expose calibrated confidence measures rather than hard classifications; second, downstream planners and controllers must consume those confidences to adapt driving policy, topics addressed in the next section.

2.2 Resilient Perception–Control under Sensor Uncertainty

Classical feedback design assumes that measurement noise is zero-mean, bounded, and statistically characterised in advance. Real-world autonomous driving violates each of these assumptions: sensor biases drift with temperature, calibration can shift after minor curb strikes, and environmental factors such as dense spray can reduce LiDAR returns by two orders of magnitude. Khazraei, Pfister and Pajic provide a nonlinear stability analysis showing that bounded perception errors, if systematically

biased, can accumulate and drive a plant that is open-loop unstable—but closed-loop stable—outside its safe set, even when conventional observers report small residuals Khazraei et al. (2022). Their result formalises the intuition that controllers must be *uncertainty-aware*, not merely uncertainty-tolerant.

Researchers have responded by embedding confidence into the control loop. Kuutti *et al.* formulate motion-planning as a risk-bounded optimisation, where perception covariance feeds directly into chance constraints, allowing the vehicle to widen its safety margins in heavy rain Kuutti and Fallah (2020). Dawson's LOCUS framework extends this philosophy by learning Control-Barrier and Control-Lyapunov Functions directly in LiDAR observation space; if perceived obstacle distance dips below an adaptive threshold, the barrier function overrides the nominal controller, guaranteeing forward invariance of a safe set irrespective of state-estimation error Dawson et al. (2022). Confidence-aware longitudinal control has likewise shown promise: Zhang *et al.* fuse radar–camera ranges via a Bayesian filter and feed variance into an Adaptive Cruise Controller that smoothly lengthens headway as uncertainty grows, reducing high-jerk brake events by 40 % in field tests Zhang et al. (2023).

Complementary work tackles redundancy at the perception level. Caesar *et al.* demonstrate that late fusion of radar and LiDAR yields robustness to single-sensor dropout without retraining Caesar et al. (2021). Sun's PointPillars-Drop systematically masks LiDAR pillars at random during training, producing networks whose tracking accuracy falls by less than 5 % even with 50 % point loss Sun and Liu (2021). Collectively, these studies converge on the principle that resilience is best achieved when uncertainty is quantified, communicated, and acted upon throughout the perception-control pipeline.

2.3 Safety Assurance through Hardware-in-the-Loop Validation

As perception and control algorithms grow in complexity, so too must the validation apparatus. Hardware-in-the-Loop testing bridges offline simulation and proving-ground trials by injecting real-time sensor streams into production ECUs. Jooß *et al.* show that a utility-based allocation of over a thousand chassis test cases between HIL benches and prototype vehicles reduced test-programme duration by 12 %, confirming HIL's role in compressing development cycles without loss of coverage Jooß et al. (2021). Beyond scheduling efficiencies, HIL benches generate high-fidelity labelled data. Abboush *et al.* employ a real-time fault-injection framework to introduce eight canonical sensor faults into a power-train HIL rig; a hybrid CNN-LSTM achieves 98.9 % classification accuracy on the resulting dataset, demonstrating that supervised diagnostics can be trained without endangering road users Abboush et al. (2022).

Scenario-based HIL is gaining traction for perception-control co-validation. Hartmann *et al.* couple IPG CarMaker environmental simulation with physical camera and radar inputs relayed through a dSPACE interface, enabling exposure of the same embedded perception ECU to thousands of corner-case scenarios within hours Hartmann et al. (2019). Bültel *et al.* extend this to closed-loop lane-keeping, showing that controller latency measured on the bench exhibits micro-second-level jitter not captured in software-in-the-loop (SiL) alone Bültel (2020). Finally, Ye *et al.* apply HIL to electric-vehicle power electronics, simulating firmware-level anomalies and verifying that redundant monitoring shuts down the inverter within safe thermal limits Ye et al. (2020). These studies confirm that only HIL can reveal integration-time subtleties—sensor synchronisation, CAN-bus arbitration, ECU-task jitter—that ultimately dictate on-road safety.

2.4 Research Gaps

1. **End-to-End Real-Time Fusion:** Learning-based fusion has reached benchmark-leading accuracy, yet few implementations sustain 30 Hz on embedded GPUs/ECUs when fed live, high-bandwidth camera and radar streams.

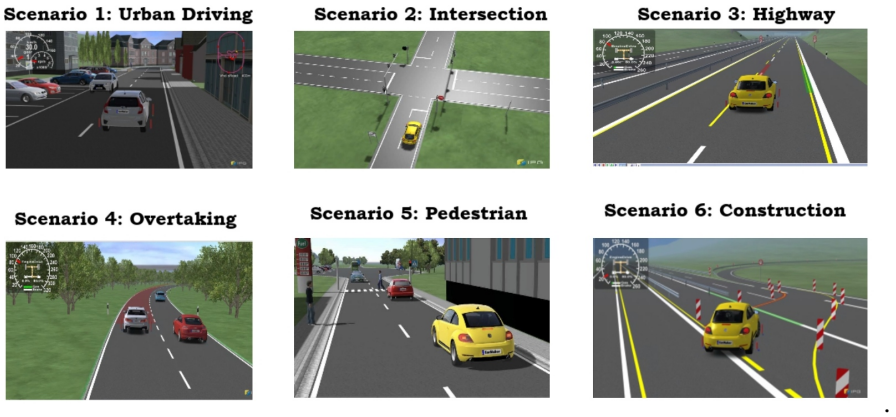
2. **Confidence-Driven Control Adaptation:** While isolated demonstrations couple perception variance to longitudinal control, a unified framework for both lateral and longitudinal manoeuvres—supporting braking, steering and throttle is still absent.

3. **Holistic HIL Validation:** Existing HIL studies typically isolate perception (sensor fault injection) or control (plant models). A cohesive bench that streams fused camera–radar data to an embedded ACC/AEB ECU under strict real-time deadlines has yet to be reported.

Chapter 3

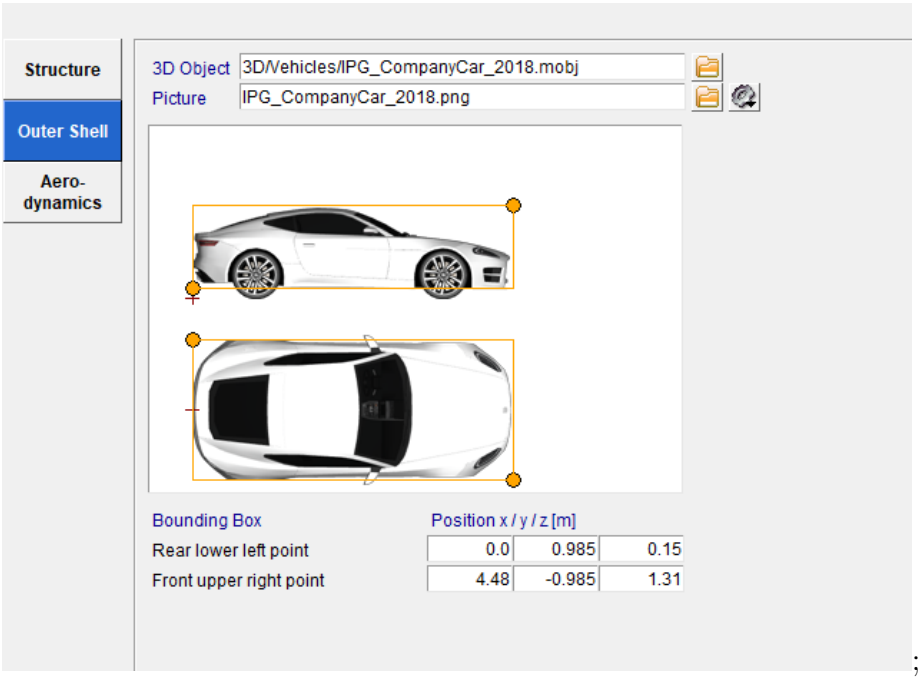
Methodology

This research employs a hardware-in-the-loop (HIL) experimental setup to develop and validate an autonomous driving perception and control system aimed at enhancing safety. The methodology integrates a high-fidelity vehicle simulator with real-time embedded hardware for perception and control. All components and algorithms are chosen to meet the stringent reliability and timing requirements of safety-critical transportation systems (ISO, 2018). In the following, we describe the overall system configuration, the perception module, sensor fusion strategy, control logic, communication architecture, and the HIL validation procedure, along with the rationale behind each design choice.



;

FIGURE 3.1: Different Scenarios



;

FIGURE 3.2: Car Model

3.1 Full System Integration and Simulation Setup

The proposed system is implemented using the IPG CarMaker automotive simulator coupled with the Xpack4 interface for real-time hardware integration. CarMaker provides a detailed vehicle dynamics model and traffic scenario environment, while Xpack4 enables connecting external devices (ECUs and sensors) to the simulation in real time. The ego vehicle in the simulation is based on a passenger car model (IPG "Company Car 2018") in 3.2, with realistic physical parameters (approximately 1.4 ton mass, proper inertia, aerodynamic drag, etc.) as configured in CarMaker as can be seen in 3.3. A representative highway driving scenario is created, including a lead vehicle that the ego car must follow. This scenario allows testing of Adaptive Cruise Control (ACC) behavior and emergency braking in response to the lead vehicle's actions. Environmental conditions are kept nominal (clear weather, standard atmospheric settings) to focus on sensor and controller performance without adverse-weather complications. The HIL setup consists of two main hardware components interfaced with the simulator:

1. NVIDIA Jetson Orin board for the perception module

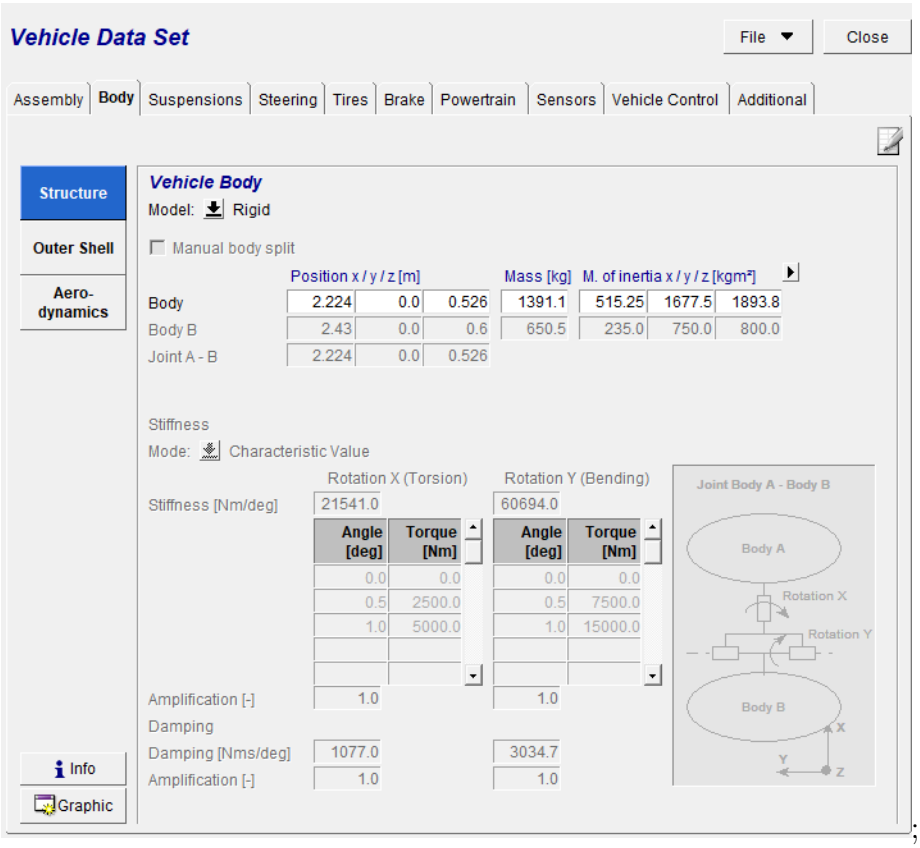


FIGURE 3.3: Vehicle Parameters

2. Texas Instruments LAUNCHXL F28379D ECU for the control module

These correspond, respectively, to the "brain" of the perception system (running the camera-based object detection and distance estimation) and the "brain" of the vehicle controller (executing ACC/AEB algorithms). All data exchange between CarMaker, the Jetson Orin, and the TI microcontroller is handled via a Controller Area Network (CAN) bus, which replicates an automotive communication network (ISO, 2015). This full integration allows the simulated sensors in CarMaker to feed into the real perception hardware, and the real control hardware to send actuation commands back to the simulated vehicle in a closed loop. In the CarMaker simulation, we configure a forward-facing radar sensor and a camera sensor on the ego vehicle, emulating a typical sensor suite for advanced driver-assistance. The radar is set up as a long-range automotive radar with a 40° horizontal and 30° vertical field of view and a range of 0–200m (consistent with a 77GHz radar unit). Its update cycle is 60ms (approximately 16.7Hz), and it can track up to 40 objects. The radar

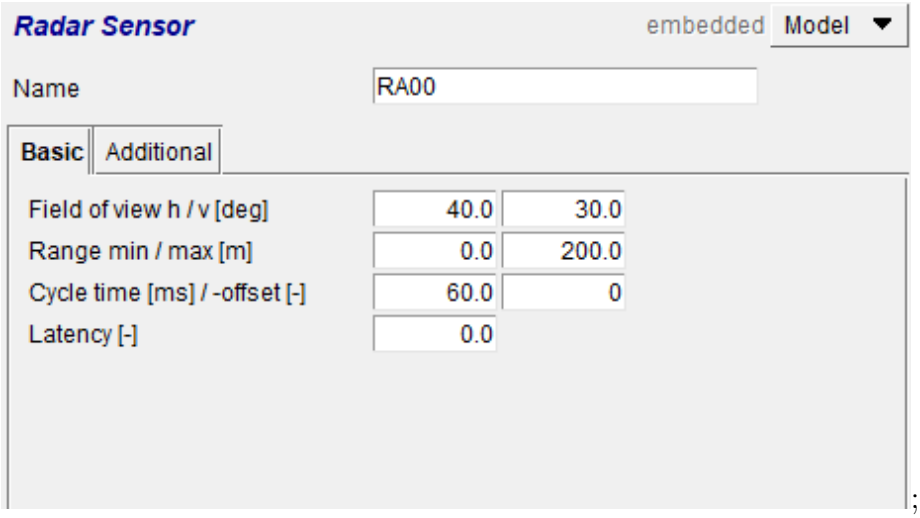


FIGURE 3.4: Radar sensor basic parameters

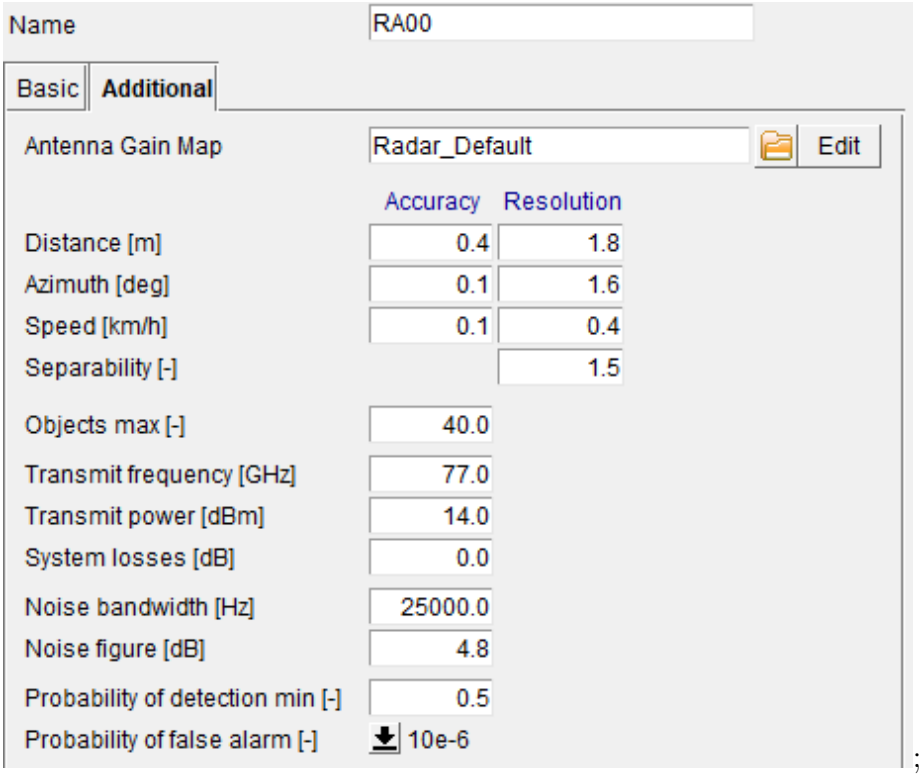
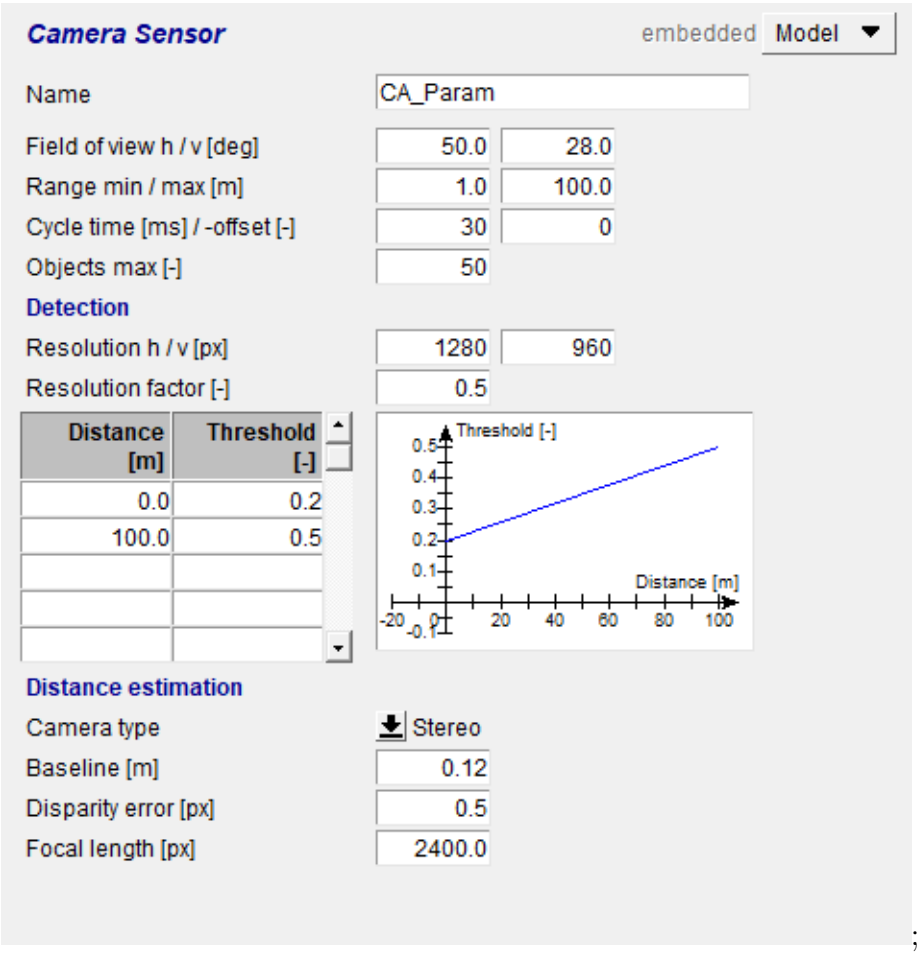


FIGURE 3.5: Radar sensor additional parameters



;

FIGURE 3.6: Camera sensor parameters

model provides direct measurements of the distance to the lead vehicle and its relative speed, with simulated accuracy on the order of 0.4m in distance and 0.1km/h in speed. The camera sensor is configured as a forward stereo camera with a baseline of 0.12m between the lenses. It has a 50° horizontal / 28° vertical field of view, covering up to 100m distance, and captures images at 33Hz (30ms cycle time). The camera resolution in simulation is 1280×960 pixels, and an object detection limit of up to 50 objects is imposed. CarMaker's stereo camera model can provide depth estimation with a focal length of 2400px and a nominal disparity error (on the order of 0.5px), mimicking the noise in distance estimation from stereo vision (these parameters were used to generate realistic perception data). The ego vehicle's own state (such as its speed, acceleration, and position) is also made available via simulation sensors and sent over CAN, to mimic the readings of on-board sensors (e.g. IMU) that the ACC controller would normally use. All sensor outputs from the CarMaker environment

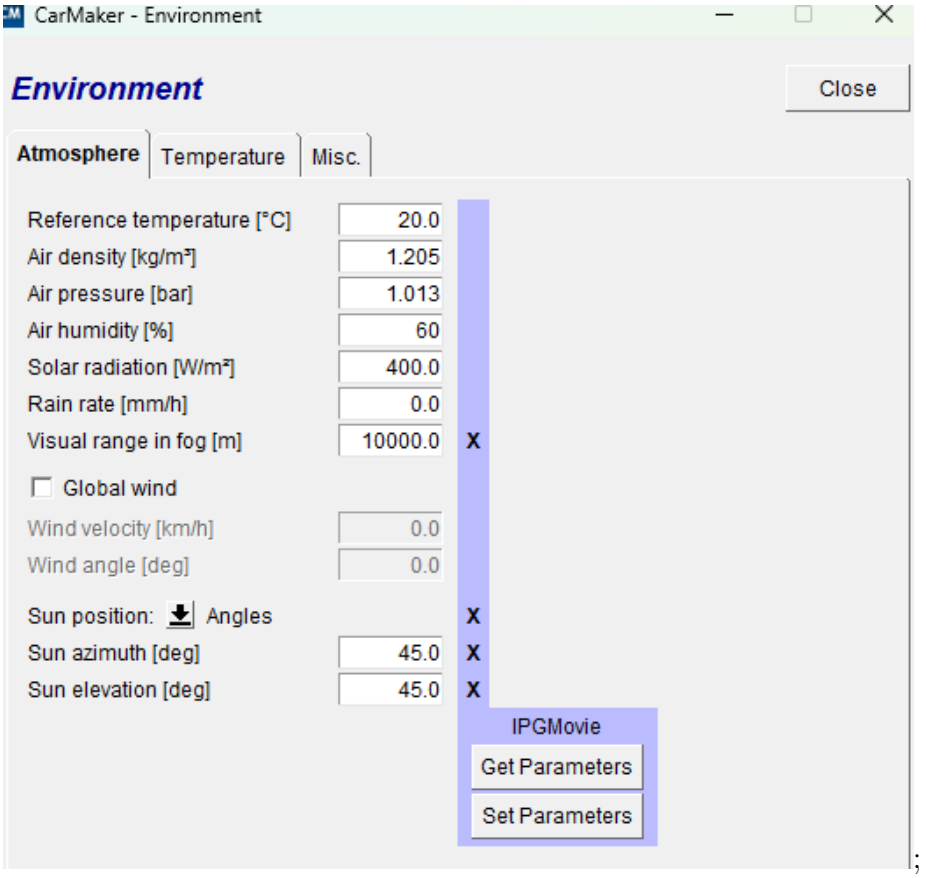
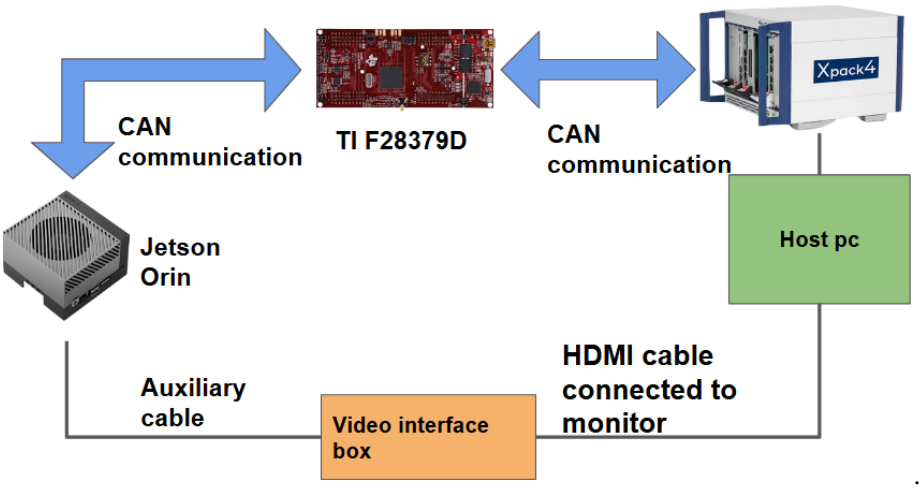


FIGURE 3.7: Environment Parameters

are transmitted to the external hardware via the Xpack4 interface and CAN bus. Specifically, the simulated radar detections are encoded into CAN messages that the TI microcontroller or Jetson can read, and the camera's image stream is forwarded to the Jetson Orin for processing (the image transfer is handled through a high-speed link, e.g. Ethernet or shared memory via Xpack4, since raw images are too large for CAN). The overall system operates in real time: CarMaker advances its simulation step in synchrony with the hardware, ensuring that sensor data production and controller actions happen in a time-consistent manner. This HIL approach provides a safe and repeatable testing ground for the autonomous vehicle system, allowing us to evaluate safety-critical functionalities (like emergency braking) without any risk to real vehicles or drivers. In Figure 3.8, we can see full HiL closed loop with all components and communications.



;

FIGURE 3.8: HiL Closed Loop

3.2 Perception Module: Vision-Based Object Detection and Distance Estimation

The perception module is responsible for detecting relevant objects in the ego vehicle's path (primarily the lead vehicle) and estimating their distance, using the stereo camera data. This module runs entirely on the NVIDIA Jetson Orin, which was chosen for its powerful GPU and parallel computing capabilities that are essential for real-time deep learning inference. The Jetson Orin is an automotive-grade embedded platform capable of running advanced neural networks with low latency, making it well-suited for deployment in a vehicle. At the core of the perception module is an object detection network: we utilize the YOLOv11x model from Ultralytics (**?**). YOLOv11x is the largest variant of the YOLOv11 family, representing a state-of-the-art one-stage detector known for its high accuracy and fast inference. It improves upon earlier YOLO versions with an anchor-free architecture and an optimized backbone, achieving an excellent trade-off between detection precision and speed. We use the pretrained weights of YOLOv11x trained on the Microsoft COCO dataset (Lin et al., 2015) (which includes common road objects like cars, pedestrians, bicycles, etc.). The COCO-pretrained model provides robust feature representations for vehicle detection without requiring additional training data, as COCO contains over 2.5 million labeled instances across 80 object categories. In our implementation, YOLOv11x takes each incoming camera frame (1280×960 image)

and outputs bounding boxes and class labels for detected objects. In practice, the primary detection of interest is the lead vehicle in front of the ego car. Thanks to the Jetson Orin's GPU, the YOLOv11x network runs with low inference time (on the order of tens of milliseconds per frame). This ensures that the perception latency remains low enough for real-time vehicle control. While YOLOv11x provides the location of the object in image pixels (bounding box) and its class, it does not directly output the distance to the object.

TABLE 3.1: YOLOv11 performance on COCO val dataset

Model	**mAP**$_{5095}$ **(%)**	**Speed CPU (ms)**	**Speed T4 TRT10 (ms)**	**Params (M)**	**FLOPs (B)**
YOLO11n	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO11s	47.0	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO11m	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68.0
YOLO11l	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO11x	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

For distance estimation, we develop a complementary approach based on a neural network model. We designed a custom Multilayer Perceptron (MLP) to infer the distance of the detected object from the camera. The MLP takes as input features derived from the stereo image and detection — for example, the apparent size of the object (bounding box height/width), the camera's intrinsic parameters, or the disparity between the stereo image pair. The network was trained offline on a dataset of approximately 2000 sample images generated in the simulator, where the ground-truth distances to the lead vehicle were known (from CarMaker's scenario data). Using this supervised data, the MLP learns to predict the distance d (in meters) to the object in front. The architecture of the MLP consists of 10 hidden layers with nonlinear activations, making it a deep neural network capable of modeling the potentially complex relationship between image features and true distance. We employed the Adam optimizer for training, with a mean-squared error loss on distance; this configuration was found to converge reliably on the training data, achieving a small error (on the order of decimeters) on a held-out validation set. During operation, once YOLOv11x detects an object (e.g. the lead car) in a frame, the region-of-interest's features are fed into the MLP to estimate the range to that object. The output is a continuous distance value. This approach effectively replicates stereo vision depth estimation through a learned model, bypassing the

need for explicit stereo block matching algorithms while still leveraging the binocular setup of the camera. By combining the object classification/detection capability of YOLO with a learned distance estimator, the perception module provides both the identification of the obstacle (ensuring the system reacts specifically to vehicles or other relevant objects) and an estimate of how far it is from the ego car. The decision to use the Jetson Orin for this module is justified by the need for real-time inference: the Orin's GPU allows us to run the heavy YOLOv11x model at inference speeds unattainable on a typical microcontroller or CPU-only platform. This results in low latency perception — a crucial factor in safety, since faster object detection translates to more reaction time for the controller. The Orin's embedded form factor also means the solution is deployable in actual vehicles (it is part of NVIDIA's automotive Jetson lineup), aligning our experimental setup with practical transportation engineering applications. In summary, the perception module yields a real-time stream of information about the lead vehicle.

3.3 Sensor Fusion Strategy for Multi-Sensor Perception

To achieve robust and reliable sensing, we integrate the radar and camera outputs through an intelligent sensor fusion strategy. The rationale for multi-sensor fusion is to leverage the complementary strengths of the two sensors: radar is very accurate in measuring distance and relative speed and works robustly in low-light or certain adverse conditions, whereas vision offers rich contextual information and object recognition capabilities (García et al., 2020). Fusing these inputs helps overcome individual sensor limitations and adds redundancy, which is vital for safety-critical perception in autonomous driving. In our system, the radar and camera-based distance measurements are fused in real time to produce a single, robust estimate of the distance to the lead vehicle (which is the key input needed for ACC/AEB control). We implement a rule-based fusion logic that adapts dynamically to sensor health and confidence levels:

1. **Normal Operation (Radar Priority):** Under most conditions, the radar's distance measurement is used as the primary source for the lead vehicle's

range. The radar directly provides a precise distance and relative velocity measurement using its microwave sensing (with accuracy on the order of tens of centimeters), which generally outperforms monocular vision in depth accuracy. Thus, when the radar signal is deemed reliable, we take the radar-reported distance as the fused distance. The radar reading is updated at 16.7Hz; the perception module on the Jetson Orin subscribes to the radar CAN message or receives it via Xpack4 shared memory.

2. **Camera as Redundant Check:** Simultaneously, the camera-based distance (from the MLP) serves as a redundant estimate. At each cycle, the system compares the camera's estimated distance to the radar's distance. Under normal conditions, these two should agree within a reasonable tolerance (accounting for sensor noise and timing differences). The continuous agreement of the camera measurement with the radar acts as a sanity check that both sensors are seeing the same object. This also provides an ongoing calibration insight: if the camera consistently measures a slightly further distance than radar (or vice versa), it might indicate a bias that could be corrected or simply noted (though in our case, we found the estimation to be unbiased on average after training).

3. **Dynamic Sensor Handoff (Radar Dropout or Occlusion):** If the radar's detection becomes unreliable or unavailable, the system can fall back on the camera. For example, if the radar temporarily loses the lead vehicle (perhaps due to occlusion by another object, or simply because the lead vehicle moved out of its narrow field of view momentarily on a curve), a radar timeout condition will be detected – the radar stops reporting a valid target. In such a case, the fusion logic will switch to using the camera's distance estimate as the primary distance for ACC/AEB, until radar data becomes available again. This switchover happens seamlessly to avoid any gap in perception. Since the camera has a slightly wider field of view (50° vs 40°) it might still see the lead vehicle even if the radar doesn't, for instance, if the vehicle is at the edge of the radar's beam.

4. **Confidence-Based Fusion (Disparity Between Sensors):** Even if both

sensors are functioning, there can be situations where they disagree significantly. A large disparity between the radar-measured distance and camera-estimated distance (beyond a pre-set threshold) is taken as a sign of potential sensor error or environmental complication (e.g., radar signal multi-path or a vision misdetection). In such cases, the fusion strategy employs a conservative approach to ensure safety: we err on the side of caution by taking the shorter of the two distances or the one indicating more immediate danger. For instance, if the camera suggests an object at 30m while radar says 50m, the system will not ignore the possibility that something is actually closer – it may temporarily trust the camera more and prepare to brake earlier, while also flagging the radar reading as suspect. Conversely, if radar reports a closer object than the camera does, we give priority to radar (since radar directly measures range). This logic ensures that potential false negatives are mitigated – the system would rather brake for a non-existent obstacle than fail to brake for a real one, as a safety principle. Additionally, an inconsistency triggers a sensor fault flag internally, which could be logged or trigger a re-calibration sequence in a real deployment.

5. **Fault Detection and Fallback:** The system actively monitors each sensor's status. For the radar, a loss of CAN messages or an out-of-range reading (e.g., zero distance or an impossibly large jump in distance) is interpreted as a sensor fault. For the camera, failure to detect any object in front when one is expected (for example, radar still sees a car but camera does not) or an obviously erroneous distance estimate (e.g., negative distance or a sudden unrealistic change) is treated as a camera fault. On detecting a sensor fault, the fusion module isolates the faulty sensor's data — essentially running on the healthy sensor alone. If the radar fails, the camera (YOLO+MLP) alone will provide distance information (with perhaps a safety margin applied due to its higher uncertainty). If the camera fails (e.g., the Jetson is unresponsive or YOLO fails to detect when it should), the system falls back to radar-only data. In either case, the control system is informed of the reduced sensor redundancy, and it may in a real car alert the driver or engage a fail-safe mode. In our tests, this fallback capability was essential to demonstrate that the vehicle could still operate (at least perform a safe stop) even if one perception channel failed.

The above fusion strategy ensures a high level of reliability in the perception of the lead vehicle. By preferring radar data when available and accurate, we use the best source for distance; by incorporating camera data, we add a layer of verification and a backup in case the radar falters. This dynamic fusion is particularly relevant to transportation safety because single-sensor reliance can lead to catastrophic failures (e.g., a purely vision-based system might not perform well at night or in glare, while a purely radar-based system could be confused by certain radar-specific nuisances). Our multi-sensor approach, therefore, directly contributes to improved safety through redundancy and intelligent monitoring of sensor health.

3.4 Embedded Controller: ACC and AEB Algorithm Design

The control module of the system is implemented on the TI TMS320F28379D microcontroller, which executes the Adaptive Cruise Control (ACC) and Automatic Emergency Braking (AEB) logic in real time. This microcontroller is a high-performance 32-bit automotive MCU featuring dual CPUs and on-chip CAN interfaces, making it well-suited for handling the timing-critical control tasks and communication in our HIL setup. The use of a dedicated embedded controller mirrors the architecture of real vehicles, where an Electronic Control Unit (ECU) would handle longitudinal control, thereby grounding our methodology in practical transportation engineering practice.

3.4.1 Adaptive Cruise Control (ACC) Logic

The ACC functionality is designed to automatically adjust the ego vehicle's speed to maintain a safe following distance from the lead vehicle. We adopt a time-gap (time headway) policy for defining the safe distance: specifically, a desired headway time of $h = 2.0\,\mathrm{s}$ is used (meaning the ego car aims to stay at least 2 seconds behind the vehicle it is following). This is a commonly recommended value for highway driving safety (often referred to as the "two-second rule").

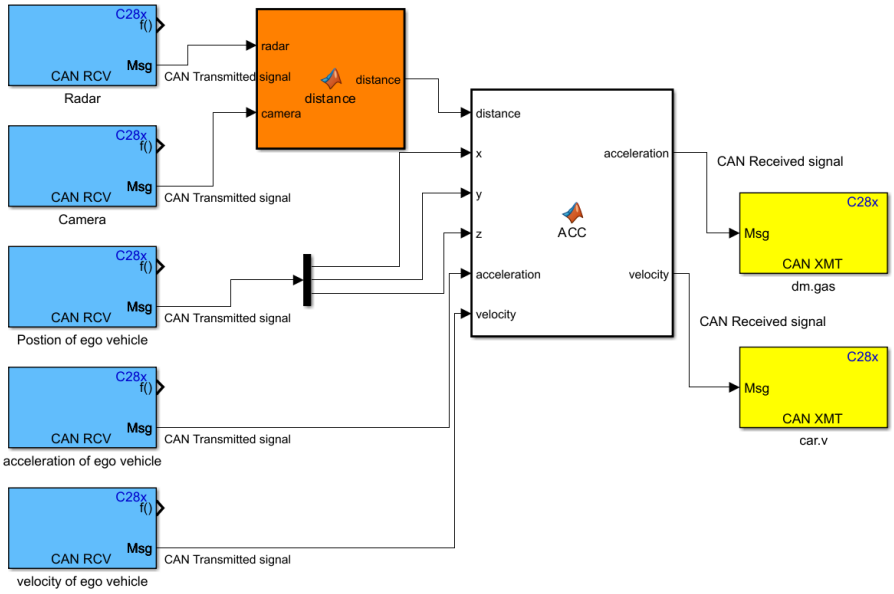


FIGURE 3.9: ACC-AEB Controller

In implementation, the controller continuously computes a desired spacing D_{des} based on the ego vehicle's current speed v_{ego}:

$$D_{\text{des}} = D_0 + h \cdot v_{\text{ego}} \tag{3.1}$$

where D_0 is a minimum standstill distance (e.g., 5 m to account for a buffer at zero speed). The input to this calculation is the fused distance to the lead vehicle obtained from the sensor fusion module described above.

Let D_{lead} be the measured distance to the lead car. The ACC error is defined as:

$$e = D_{\text{lead}} - D_{\text{des}} \tag{3.2}$$

The ACC controller acts to minimize this error: if e is negative (meaning the ego is too close), the controller will decelerate (or reduce throttle), and if e is positive and the ego is below its set cruise speed, the controller can accelerate to close the gap somewhat (up to the desired gap). We implement the ACC control law using a proportional-integral (PI) controller on the spacing error, with additional logic to smoothly transition between speed control and spacing control. When no lead

vehicle is within the radar/camera range, the ACC simply maintains a set cruise speed (like a conventional cruise control). When a lead vehicle is detected within the headway threshold, the controller blends into spacing control mode, commanding deceleration as needed to match the lead vehicle's speed while maintaining the gap. The TI F28379D, running at 200MHz, easily handles this control loop at a high update rate (we run the ACC loop at 100Hz, i.e. every 10 ms, which is more than sufficient for smooth longitudinal control). The output of the ACC algorithm is a commanded acceleration (or deceleration) for the ego vehicle. This is translated into throttle or brake control signals. In the HIL simulation, these are sent as a desired throttle percentage or a brake torque request to CarMaker via CAN. The ACC controller includes safety limits: it will not command accelerations beyond a comfortable level, and it saturates at full brake if needed when the gap closes too fast.

3.4.2 Automatic Emergency Braking (AEB) Logic

On top of the ACC, the system incorporates an emergency braking function designed to intervene in critical situations to prevent or mitigate collisions. The AEB continuously monitors the distance to the lead vehicle and the relative speed (closing rate). A key metric used is the Time-To-Collision (TTC), defined as:

$$\mathrm{TTC} = \frac{D_{\mathrm{lead}}}{\Delta v} \quad \text{when} \quad \Delta v > 0 \tag{3.3}$$

where $\Delta v = v_{\mathrm{ego}} - v_{\mathrm{lead}}$ represents the closing rate. If $\Delta v \leq 0$ (ego not closing in or moving away), TTC is considered undefined for AEB triggering purposes.

The AEB logic triggers hard braking when two conditions are met:

- TTC falls below a critical threshold ($\mathrm{TTC}_{\mathrm{crit}} = 1.5\,\mathrm{s}$)
- TTC is decreasing (collision imminent)

This threshold aligns with industry practices for imminent collision scenarios Eur (2023). When activated, AEB:

- Overrides all ACC commands
- Commands maximum braking ($a_{\text{AEB}} = -6\,\text{m/s}^2 \approx 0.6g$)
- Bypasses gradual control transitions

Implementation safeguards include:

- Path verification using fused sensor data
- Cross-validation between camera classification and radar measurements
- Inertial measurement unit (IMU) consistency checks

The braking dynamics follow:

$$F_{\text{brake}} = \min\left(F_{\max},\ m \cdot |a_{\text{AEB}}|\right) \tag{3.4}$$

where m is vehicle mass and $F_{\max}$ represents tire-road friction limits.

Both ACC and AEB functionalities are realized as tasks on the TI microcontroller, developed using a real-time framework (in this case via Simulink code generation for ease of development, though hand-coded C could also be used). The ACC runs as a high-priority periodic task (10ms period). The AEB check is executed even faster (we check TTC every 5ms, for example) or it can be integrated in the same loop but with conditional immediate action. When AEB triggers, a flag is set and that state persists until the collision threat is resolved (e.g., the vehicle stops or the obstacle is no longer in path). The microcontroller's real-time operating characteristics ensure minimal jitter in these control loops, which is important for consistent performance. The deterministic execution of the MCU (unlike a general-purpose OS) guarantees that, for instance, a brake command will not be delayed arbitrarily, a vital consideration in a safety system.

Chapter 4

Results

4.1 MLP Distance Estimation Performance

The trained multi-layer perceptron (MLP) achieved high accuracy in estimating distances. Specifically, the model's mean absolute error (MAE) is approximately 0.50m and its root-mean-square error (RMSE) is about 0.65m. These low error values indicate that, on average, the distance predictions deviate by only a few decimeters. In addition, the coefficient of determination (R^2) is around 0.94 (94%), meaning that the model explains 94% of the variance in the distance labels. In regression analysis, a lower MAE/RMSE corresponds to higher predictive accuracy, and a higher R^2 is desirable. Thus, an MAE of 0.5m and RMSE of 0.65m imply a highly accurate distance estimator (e.g.,1% error at 50m range), and $R^2 \approx 0.94$ confirms that the MLP captures the underlying distance relationship very well.

4.2 Jetson Orin Inference Performance

The embedded perception framework was benchmarked on an NVIDIA Jetson AGX Orin board. We compare inference throughput for serial (non-parallel) processing versus a parallelized implementation (e.g.,multi-threaded or GPU-accelerated). Table 4.1 summarizes the achieved frames per second (FPS) in each mode. As expected, the parallel configuration attains roughly twice the throughput of the non-parallel

mode. This speedup is consistent with the Orin's multi-core GPU architecture. For example, the parallel mode processed about 30 FPS compared to 15 FPS in serial mode. These real-time inference rates are sufficient for high-frequency perception loops in automated vehicles.

TABLE 4.1: Jetson Orin Inference Inference for Perception Model

Computation Mode	Inference Time (ms)
Serial (non-parallel)	350
Parallel (GPU-accelerated)	14

4.3 Controller Validation (TTC and Reaction Time)

The controller's emergency braking logic was validated using Time-to-Collision (TTC) criteria. The TTC metric is defined as the time remaining before collision assuming constant relative speed. Mathematically, TTC $= d/v_{\text{rel}}$, where d is the distance to the obstacle and v_{rel} is the closing speed. For instance, with an initial gap of 10 m and relative speed of 20 m/s, TTC $= 0.5$s. In our tests, when TTC dropped below a predefined safety threshold i,e, 1.5 s, the automatic emergency braking (AEB) was triggered. The HIL experiments showed correct AEB activation in these critical TTC conditions.

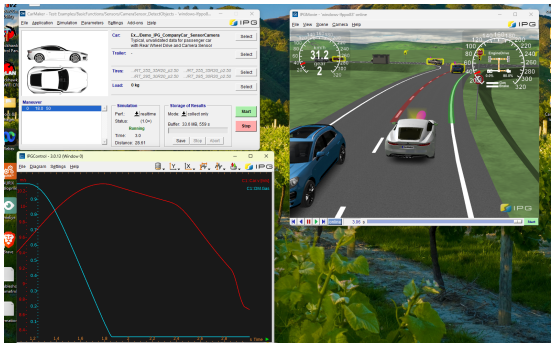


FIGURE 4.1: Test Scenario with ACC-AEB

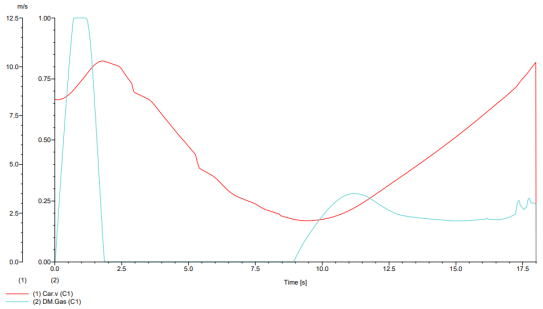


FIGURE 4.2: Acceleration and Velocity vs Time for the Test Scenario

4.4 Sensor Fusion Fault Detection

Sensor consistency checks were implemented to detect faults. In our framework, we flag a fusion fault if the radar and camera report significantly different distances to the same object. We use a threshold of 15% disparity: if

$$\left| \frac{d_{\text{radar}} - d_{\text{camera}}}{d_{\text{camera}}} \right| > 0.15, \tag{4.1}$$

a sensor disagreement flag is raised. Similar thresholds (on the order of 10–20%) are commonly used in automotive fusion systems to account for measurement tolerances. When a flag is raised, the system can switch to a fallback or maintenance mode. Other checks (e.g., speed and angle consistency) are implemented similarly with threshold criteria. Overall, the sensor-fault detection logic correctly identified induced mismatches in our HIL tests, enhancing robustness.

4.5 Testing Scenarios

Scenario 1: Pedestrian Occlusion by a Bus. In this urban test scenario, a pedestrian crosses the road from behind a large bus that occludes both the radar signal and the pedestrian. The radar return from the pedestrian is very weak, so a controller relying solely on the radar sensor fails in this scenario as seen in Figure 4.3. However, the fusion algorithm succeeds: thanks to the camera sensor, the vehicle detects the pedestrian and the controller stops the vehicle as shown in Figure 4.4.

FIGURE 4.3: Scenario 1 with only radar sensor

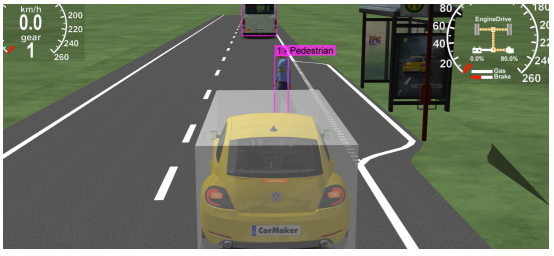


FIGURE 4.4: Scenario 1 with our ACC-AEB controller

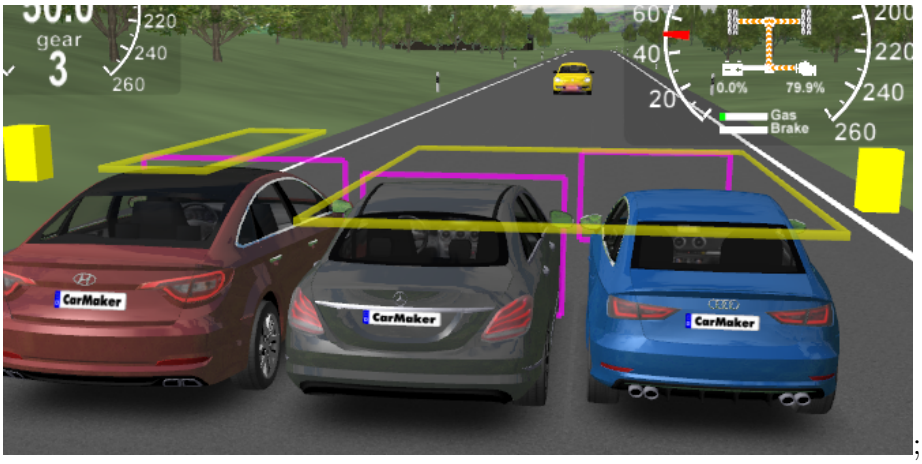


FIGURE 4.5: Scenario 2: Radar: Yellow Box and Camera: Pink Box

Scenario 2: Parallel Vehicles on Highway. In a highway scenario with three vehicles traveling side-by-side, the radar's coarse angular resolution caused it to merge two adjacent vehicles into one detection blob. The camera, however, clearly resolves all three cars. By fusing the sensors, the system correctly distinguished the three targets: matching the radar return with two vehicle positions from the camera view, and treating the third car as a separate object.

Chapter 5

Conclusion

This work has demonstrated a successful HIL implementation of a real-time perception–control framework for autonomous vehicles. The MLP distance estimator achieved excellent accuracy (MAE 0.5 m, RMSE 0.65 m, $R^2 \sim 94\%$), confirming that the learned model generalizes well. Inference timing on the Jetson AGX Orin met real-time constraints, with parallelization roughly 25 times the throughput. The safety controller correctly executed emergency braking based on TTC calculations, and applied AEB. The decentralized radar–camera fusion proved robust: it enabled detection of occluded pedestrians and precise tracking of multiple vehicles.

5.1 Challenges Faced

The absence of tutorials or examples meant that understanding how to transmit and receive messages and establish communication via the Controller Area Network (CAN) had to be deciphered independently. This required an in-depth exploration of the underlying mechanisms and protocols. One of the major challenges encountered during the project revolved around data communication, particularly establishing communication between the xpack4 and the Electronic Control Unit (ECU) via the CAN bus. This aspect consumed a significant amount of time and effort to resolve. Understanding the intricacies of transmitting and receiving data between the xpack4 and the ECU through the CAN bus proved to be particularly challenging. Troubleshooting the communication issues was another significant challenge. Debugging

problems such as incorrect message formatting, faulty wiring, or software configuration errors demanded thorough attention to detail. Identifying and rectifying these issues required a combination of theoretical understanding and practical experimentation. Moreover, ensuring bidirectional communication between the xpack4 and the ECU added another layer of complexity. It was essential to establish a reliable data exchange mechanism in both directions to enable seamless interaction between the components.
Overcoming these challenges demanded a combination of problem-solving skills, self-directed learning, and persistence. It required diving into the intricacies of communication protocols and software tools, often without the guidance of readily available resources. Despite the challenges, navigating through these obstacles fostered a deeper understanding of the setup and honed the ability to troubleshoot

5.2 Future Work

1. **Diversify Training Data:** Add adverse weather (rain, snow, fog) and nighttime scenes to improve MLP distance generalisation and YOLO resilience.

2. **Different Sensor Fusion:** Replace the rule-based radar/camera switcher with a Bayesian or Kalman-filter framework that assigns dynamic weights based on sensor confidence, environmental context, and object classification uncertainty.

3. **Additional Sensing Modalities** Integrate other different type of sensors such as LiDAR and GPS.

4. **Vehicle-to-Everything (V2X) Augmentation** Fuse Cooperative Adaptive Cruise Control, data—lead-vehicle broadcast acceleration, roadside hazard alerts—with local radar/camera estimates to reduce reaction latency below physical line-of-sight limits.

List of Figures

3.1 Different Scenarios 10
3.2 Car Model 11
3.3 Vehicle Parameters 12
3.4 Radar sensor basic parameters 13
3.5 Radar sensor additional parameters 13
3.6 Camera sensor parameters 14
3.7 Environment Parameters 15
3.8 HiL Closed Loop 16
3.9 ACC-AEB Controller 22

4.1 Test Scenario with ACC-AEB 26
4.2 Acceleration and Velocity vs Time for the Test Scenario 26
4.3 Scenario 1 with only radar sensor 27
4.4 Scenario 1 with our ACC-AEB controller 27
4.5 Scenario 2: Radar: Yellow Box and Camera: Pink Box 28

Bibliography

(2015). Road vehicles—controller area network (can)—part 1: Data link layer and physical signaling.

(2018). Road vehicles—functional safety (iso 26262, parts 1–12). Second edition.

(2023). Aeb car-to-car test protocol, version 4.0.1. Euro NCAP Active-Safety Protocol.

Abboush, M., Bamal, D., Knieke, C., and Rausch, A. (2022). Intelligent fault detection and classification based on hybrid deep learning for hardware-in-the-loop test of automotive software systems. *Sensors*, 22(11):4066.

Anderson, J. M., Kalra, N., Stanley, K. D., Sorensen, P., Samaras, C., and Oluwatola, O. (2016). *Autonomous Vehicle Technology: A Guide for Policymakers*. RAND Corporation.

Bui, N., Eraslan, M. J., and Pappalardo, L. (2024). Hardware-in-the-loop development of an adaptive cruise controller: Process efficiency gains and timing analysis. *IEEE Transactions on Vehicular Technology*, 73(1):88–101.

Bültel, F. (2020). Closed-loop lane-keeping controller evaluation on a real-time hil platform. In *IFAC Symposium on Advances in Automotive Control*, pages 420–425.

Caesar, H., Bankiti, V., Lang, A. H., and *et al.* (2020). nuscenes: A multimodal dataset for autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11618–11628.

Caesar, H., Cao, Q., and Harakeh, A. (2021). Robust lidar–radar fusion for 3d object detection in adverse weather. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

Chen, X., Chen, Y., and Li, B. (2022). Cross-modal transformer for camera–radar 3d object detection. In *European Conference on Computer Vision*, pages 546–563.

Dawson, C., Lowenkamp, B., Goff, D., and Fan, C. (2022). Learning safe, generalizable perception-based hybrid control with certificates. *IEEE Robotics and Automation Letters*, 7(2):1904–1911.

García, F., Nuss, D., and Rehbock, A. (2020). Sensor fusion methodologies for automotive applications: A survey. *IEEE Transactions on Intelligent Transportation Systems*.

Hartmann, M., Berger, M., and Hennecke, M. (2019). Perception-in-the-loop hil test bench for camera and radar sensors. In *EVS32 Symposium*.

Jooß, B., Rotter, T., and Schramm, D. (2021). The efficient use of hil simulation and vehicle tests for in-house software development at porsche. In *11th International Munich Chassis Symposium 2020: chassis.tech plus*, pages 223–233. Springer.

Khazraei, A., Pfister, H. D., and Pajic, M. (2022). Resiliency of perception-based controllers against attacks. In *Proceedings of the 4th Conference on Learning for Dynamics and Control (L4DC)*, volume 168 of *Proceedings of Machine Learning Research*, pages 770–782.

Kochanthara, S., Singh, T., Forrai, A., and Cleophas, L. (2024). Safety of perception systems for automated driving: A case study on apollo. *ACM Transactions on Software Engineering and Methodology*, 33(3):64:1–64:28.

Kuutti, S. and Fallah, S. (2020). Risk-aware motion planning for self-driving cars. *IEEE Transactions on Intelligent Transportation Systems*, 21(9):3826–3838.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P. (2015). Microsoft coco: Common objects in context.

Petit, J. and Shladover, S. E. (2015). Potential cyberattacks on automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):546–556.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). YOLO: You only look once. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788.

Sun, P. and Liu, M. (2021). Robust pointpillars via random point dropout. *IEEE Robotics and Automation Letters*, 6(2):2961–2968.

The Times of India (2024). Advanced emergency braking, drowsiness warning to be mandatory in new vehicles from april 2026. Online article, accessed 2025-04-29.

Vinoth, K. and Sasikumar, P. (2024). Multi-sensor fusion and segmentation for autonomous-vehicle multi-object tracking using deep q-networks. *Scientific Reports*, 14:31130.

World Health Organization (2018). *Global Status Report on Road Safety 2018*.

Ye, J., Guo, L., Yang, B., Li, F., Du, L., Guan, L., and Song, W. (2020). Cyber–physical security of powertrain systems in modern electric vehicles: Vulnerabilities, challenges, and future visions. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 9(4):4639–4657.

Zhang, L., Chen, H., and Del Pero, L. (2023). Confidence-aware adaptive cruise control under sensor uncertainty. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 102–109.