International Transactions on Electrical Engineering and Computer Science
Zeina et.al., Vol: 3, No: 1, pp: 41-56, March 2023

An Efficient Design of a Basic Autonomous Vehicle Based on CAN Bus

Zeina Ali[1], Qutaiba I Ali[2]

Abstract: The evolution of autonomous vehicles (AVs) has captured widespread interest lately, offering prospects for enhanced road safety, alleviation of traffic congestion, and heightened fuel efficiency. The essential role of Controller Area Network (CAN) bus protocols in the operational integrity of AVs is undeniable. This research outlines the architecture of an elementary autonomous vehicle predicated on diverse CAN bus frameworks. We introduce a design that integrates a multiplexed CAN bus arrangement, fostering streamlined interactions among the AV's various subsystems. This design is inherently scalable, promoting ease of modification in step with evolving autonomous vehicular technologies. The paper delves into the intricacies encountered throughout the design phase and articulates the methodologies adopted to surmount these obstacles. Empirical simulations substantiate the efficacy of our design, underscoring its dependability and strength across a multitude of vehicular contexts. Fundamentally, this design lays the groundwork for the advancement of sophisticated AVs, thereby contributing to the realization of optimized vehicular ecosystems.

Keywords: Electronic Control Unit, CANoe, Controller Area Network, Autonomous Vehicles

1. Introduction

1.1 Background

Vehicle designed to operate independently of human control are known as autonomous vehicles (AVs). These vehicles use an array of sensors, including cameras, LiDAR, and radar, as well as a variety of electronic control units (ECUs), to detect their environment and make decisions in real-time [1-2].

Article History
Received: 01-12-2023; Revised: 05-02-2024;
Accepted: 15-02-2024

✉ Zeina Ali
zinah.mohammed@uoninevah.edu.iq

[1]Computer and Information Engineering Department, Ninevah University / Electronics Engineering College, Mosul- 41002, Iraq.

[1,2]Department of Computer Engineering, University of Mosul / College of Engineering, Mosul-41002, Iraq.

Research, as referenced in [3], has delved into architectural models for the technical and functional design and development of AV systems. These models aim to streamline the development lifecycle and avoid duplication of effort in each new AV project. An AV's functional architecture is organized into modules or blocks determined by the sequence and processing of data, from its initial collection to the final vehicle control systems, as well as the internal monitoring mechanisms of the vehicle itself. The architectures proposed by researchers typically feature four main functional blocks—perception, planning and decision-making, motion and vehicle control, and system supervision. These blocks handle the processing of vehicle data collected by the ECUs, a process typically illustrated in Fig. 1 [1], [2-6].

In the initial perception phase, the ECUs gather data from the vehicle's surroundings and send it to the AV's central processing unit via networks such as the Controller Area Network (CAN) bus. This data is then analyzed during the planning and decision-making phase, where artificial intelligence algorithms determine the vehicle's course of action [7].

https://doi.org/10.62760/iteecs.3.1.2024.70

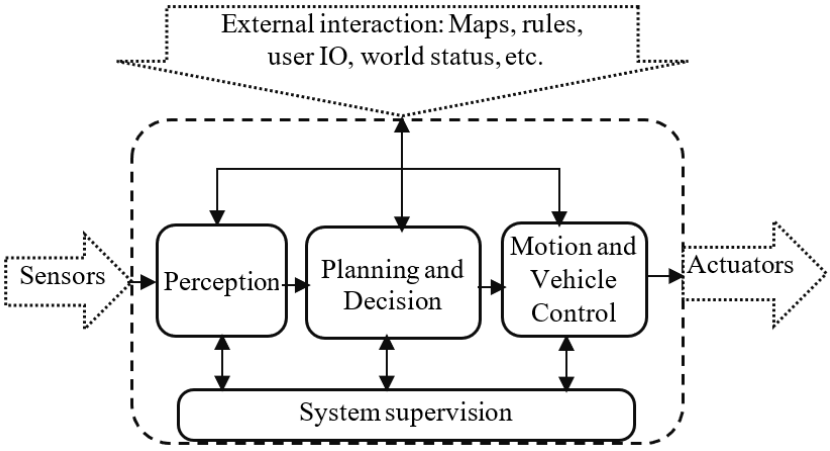


Fig. 1: Main functional modules of the AV system

Following this, the control phase involves the AV's system executing these decisions by sending instructions to various vehicle components through the CAN bus, ensuring smooth and safe vehicle operation. The perception stage is tasked with accurately and dependably sensing the environment using sensors like cameras, LiDAR, radar, and additional ECUs. This stage is responsible for real-time identification and tracking of various elements such as vehicles, pedestrians, and road signs, and for calculating these elements' speed, distance, and direction to facilitate informed and safe autonomous driving [3-5].

The CAN bus is a robust asynchronous serial bus system that employs a multi-master protocol, granting equal communication rights to all connected ECUs [6]. It's a prominent standard for automotive applications, operating at the OSI model's physical and data link layers. Its arbitration method relies on CSMA/CD principles and prioritizes messages according to the AMP technique [8]. There are several versions of the CAN bus, each offering different capabilities. Classical CAN, as per ISO11898-1, connects ECUs at speeds up to 1 Mbps. The CAN FD, or ISO 11898-2, supports data transfers up to 8 Mbps. The latest development, CAN XL (ISO 11898-3), is poised to facilitate data transfer rates as high as 10 Gbps [9-10].

The CAN bus is instrumental for communication between external and internal ECUs in an AV [11-13]. External ECUs provide crucial environmental data, while internal ECUs monitor the vehicle's status. This real-time data exchange is vital for the AV's central computer, which uses machine learning and other advanced technologies to make navigational decisions [5], [14-15]. Given that modern intra-vehicle communication systems typically consist of several subnetworks with diverse protocols, automotive gateways are essential. They act as interfaces between these subnetworks, ensuring the integrity and security of the intra-vehicle communication network, a role that should never be overlooked [16-17].

1.2 Related Works

The articles over the last decade demonstrate a large amount of interest in the CAN bus technology in automotive communication systems, and they cover a variety of the following subjects:

a. Security and Privacy of CAN Bus

Several publications addressed the security and privacy issues related to the CAN bus in vehicular communication systems. These works proposed various techniques to enhance the security of the CAN bus, such as message authentication, intrusion detection, and access control [18-20].

b. Fault Diagnosis and Recovery

Several publications proposed techniques for fault diagnosis and recovery in the CAN bus. These works aimed to increase the reliability of the CAN bus in vehicular communication systems, especially in safety-critical applications [21-23], [18].

c. CAN Bus Performance Analysis

Several publications presented performance analysis of the CAN bus in vehicular communication systems. These works focused on the effects of various parameters, such as bus load, message size, and bus topology, on the performance of the CAN bus [24-27].

d. CAN Bus Network Management

Several publications proposed techniques for managing the CAN bus network in vehicular communication systems. These works aimed to optimize the utilization of the CAN bus resources, such as bandwidth and message latency, and to reduce the overall network congestion [28-30].

e. CAN Bus in Autonomous Vehicles

Several publications investigated the use of the CAN bus in autonomous vehicles. These works proposed various techniques for designing efficient and reliable CAN bus-based communication systems for autonomous vehicles [31-32].

1.3 Our contributions

This paper's key contributions are summarized as follows:

- The research constructs a foundational model for an Autonomous Vehicle (AV) by examining various Controller Area Network (CAN) bus standards. It particularly assesses how both the internal and external Electronic Control Units (ECUs) of the AV influence the performance of the CAN bus.
- Modifications to the CAN network's structure are proposed based on performance outcomes, which resulted in enhanced operational efficiency and reduced communication delays.
- The study proposes specifications for a multiprotocol CAN bus system, tailored for the development of AV models.

The structure of the remainder of this paper is organized as follows: Section II provides an overview of the AV's external ECUs and the software tools, which are elaborated upon in Section III. The research approach is delineated in Section IV, followed by a detailed discussion of the simulation scenarios and their outcomes in Section V. The paper concludes with final thoughts and reflections in Section VI.

2. AV external ECUs

Electronic Control Units (ECUs) are tasked with measuring events or environmental alterations for further analysis. These units are essential for providing a detailed and positional understanding of the surroundings, enabling the vehicle to navigate and make decisions in real time. This section introduces the four primary ECUs essential for environmental perception in AV technology: the camera, radar, INS, and LiDAR, all of which are integral to the research presented here.

2.1 Camera

The Vision Detection Generator ECU consists of both a general camera ECU and a specialized monocular camera ECU. The camera ECU's configuration includes intrinsic parameters, such as the focal length and optical center of the camera, as well as extrinsic parameters that involve the camera's orientation in terms of pitch, yaw, and roll [33].

2.2 Radar

Radar, short for Radio Detection and Ranging, operates by emitting electromagnetic waves into the surrounding area and interpreting the waves that bounce back from various objects. It leverages the Doppler effect of these electromagnetic waves to ascertain the velocity and position of potential obstacles [34]. Vehicle radar systems are generally categorized into Medium-Range Radar (MRR), Long-Range Radar (LRR), and Short-Range Radar (SRR).

2.3 INS

The Inertial Navigation System (INS) is a positional ECU designed to determine a vehicle's pose—including its position and orientation—and velocity. It is a critical component for precise navigation.

2.4 LiDAR

LiDAR stands for Light Detection and Ranging. This technology operates by emitting laser or infrared light pulses through rotating lenses and timing the reflection of each pulse to gauge distances.

This section provides a brief introduction to each of the four basic ECUs for environmental perception in AV applications: camera, radar, INS and the LiDAR ECUs will be used in this research.

3. Software tool

Recent studies [35-36] suggest that conducting real-world field tests to assess the safety of autonomous vehicles (AVs) within a practical timeframe is unfeasible. Consequently, leveraging virtual environment testing and verification emerges as a solution to expedite, control, and comprehensively evaluate AV systems. The simulation approach will utilize two distinct software tools: Matlab 2021a's Driving Scenario Designer for crafting virtual driving scenarios, and Vector CANoe 16 for simulation and analysis within CAN networks. The combination of these programs creates a cohesive simulated network.

3.1 Driving Scenario Designer Application

Matlab's Driving Scenario Designer is a tool designed for creating artificial driving scenarios, which are essential for evaluating autonomous driving systems. This tool is highly versatile, enabling the creation of a wide range of scenarios crucial for the development of autonomous vehicles [33]. This software facilitates the construction of virtual roads and the integration of dynamic elements within these environments. Users can populate these roads with various actors, representing the dynamic components, through an intuitive drag-and-drop interface [33].

3.2 Vector-Canoe

CANoe, developed by the German company Vector, is an advanced system for designing and analyzing electronic networks. It provides capabilities for live monitoring of actual bus communications, complete digital simulation of bus systems using virtual nodes, and hybrid simulations that blend physical and virtual nodes. CANoe streamlines the entire bus development process, from conception through simulation to testing and analysis. It supports an array of network types, including Ethernet, MOST, FlexRay, LIN, and CAN, making it a versatile tool widely adopted by OEMs and suppliers globally for its range of features and configurations [37].

The simulations and testing protocols within CANoe are scripted using the Communication Access Programming Language (CAPL), an event-driven language that can respond to various triggers, such as message reception, signal changes, simulation start or stop events, and variable modifications. Written in a syntax similar to the C programming language, CAPL supports communication protocols like J1939 and CANopen in addition to CAN, LIN, FlexRay, Ethernet, MOST, and others [38]. The simulation approach will utilize two distinct software tools: Matlab 2021a's Driving Scenario Designer for crafting virtual driving scenarios, and Vector CANoe 16 for simulation and analysis within CAN networks. The combination of these programs creates a cohesive simulated network.

4. Research methodology

Data for the external ECUs of the AV, capturing details from the vehicle's immediate environment, are configured using the Driving Scenario Designer application in Matlab. This data is subsequently transferred to CANoe, where it is integrated with the data from the AV's internal ECUs to examine their collective impact on the CAN bus network. In this section, we outline the prerequisites for setting up the envisioned network. This includes detailing the simulated roadways in Matlab and the integration framework for the Matlab/CANoe programs. It also encompasses the specifications for the external ECUs and the protocols for internal messaging.

4.1 Roads Traffic Map

Fig. 2 offers a broad depiction of a vehicular situation, highlighting an intersection with two roads, an Ego vehicle in blue, and two additional vehicles in yellow and red. Each vehicle follows a designated path marked by waypoints, with specified speeds in meters per second and wait times in seconds. The illustration includes barriers and pedestrians positioned along the roadside.

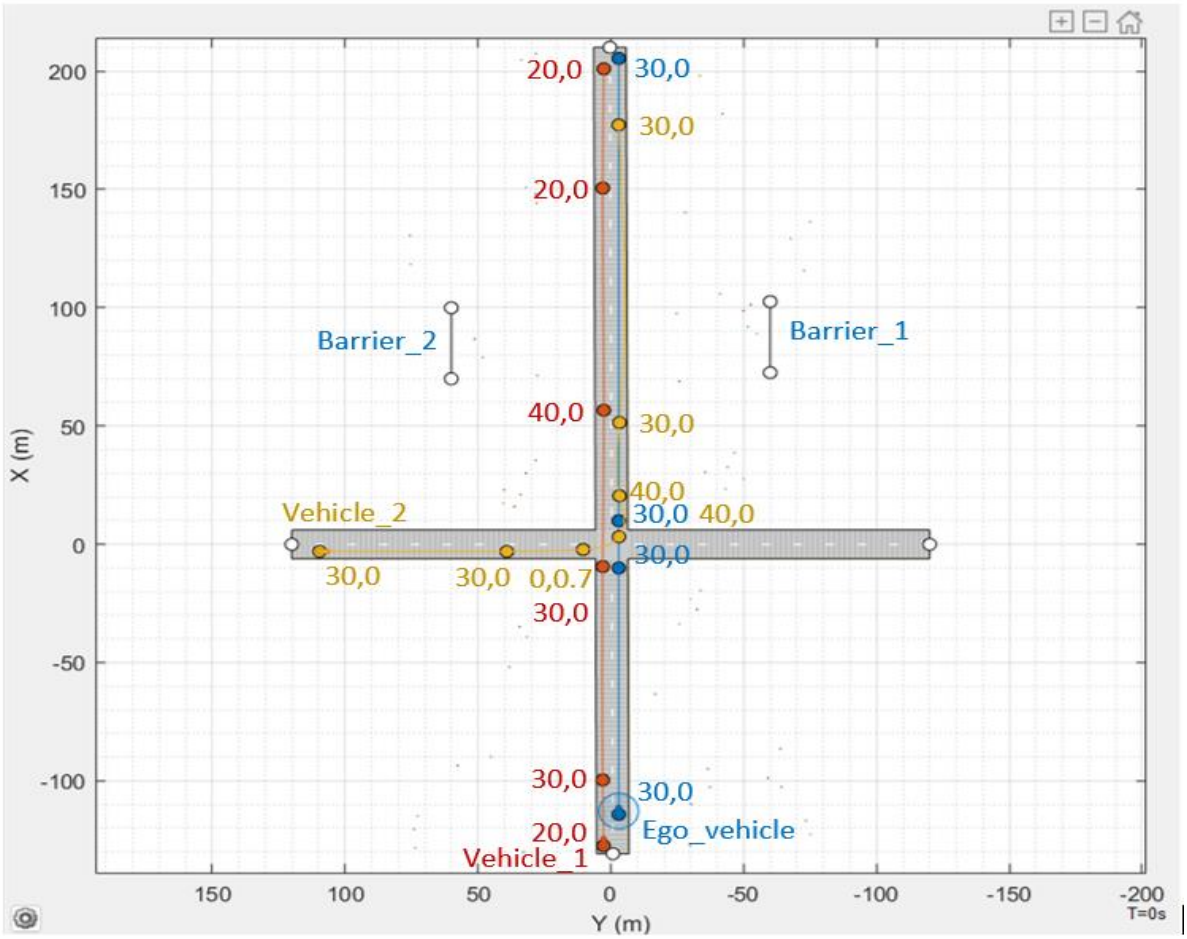


Fig. 2: General description of the scene

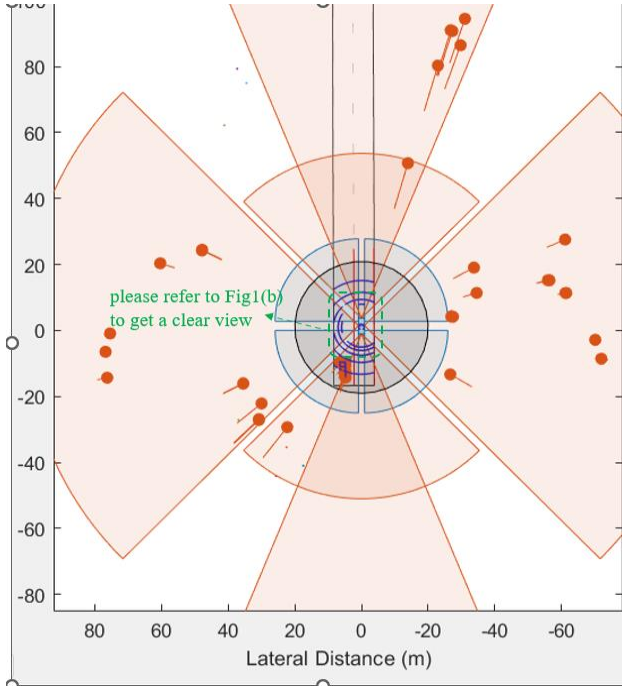
Table. 1: Types of external ECUs

ECU type	Number of ECUs	Maximum number of detection objects
Camera	4	11 object, 2 lanes
Radar	6	20 objects
INS	1	-
LiDAR	1	-

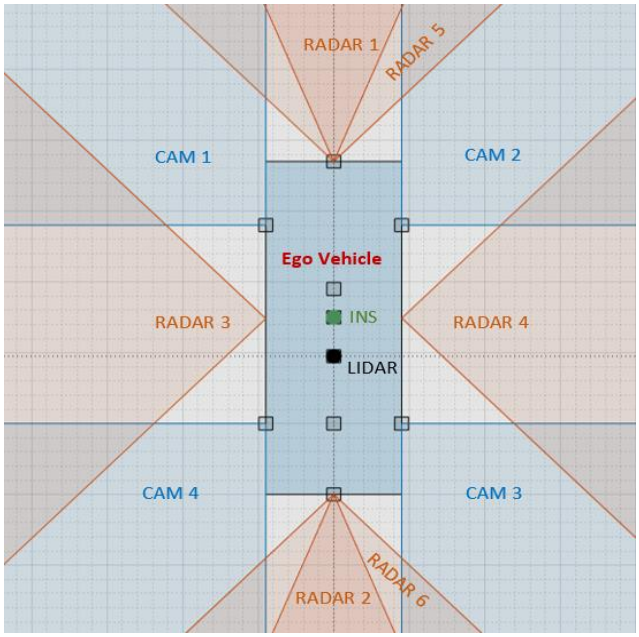
Table. 2: Delay deadline requirements for AVs

ECU	Delay dead line (ms)
Lidar	**10**
Radar	**10**
Camera	**10**
Control	**50**

Table. 1 lists the maximum camera and radar detection objects and external ECU types and numbers. Table. 2 shows each ECU's delay deadline communication requirements [9].



(a)



(b)

Fig. 3: (a) A Birds-eye plot (b) ECU positioning and placement on Ego vehicle

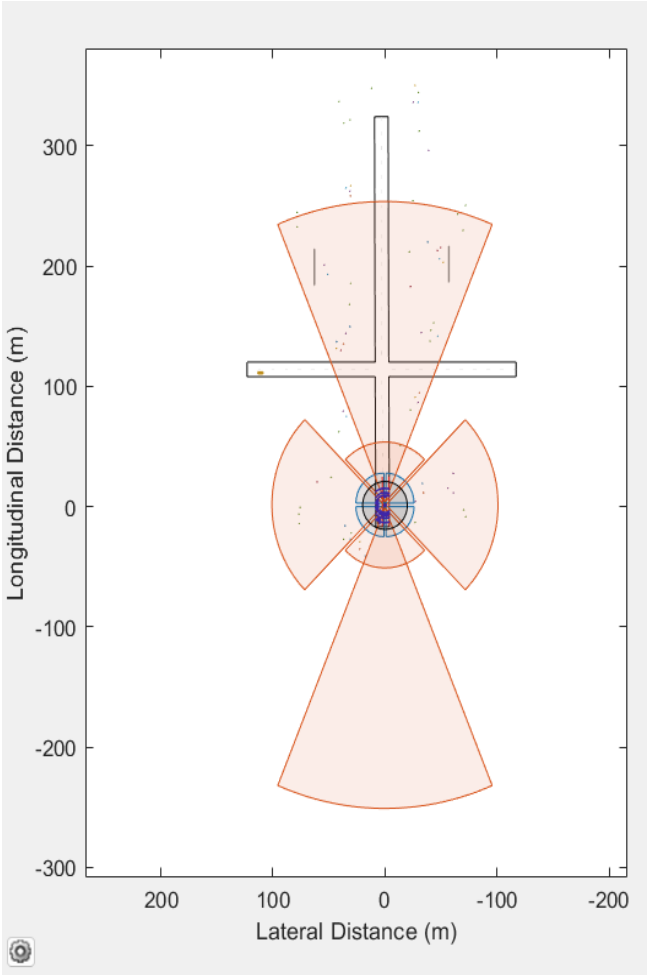
The simulation encompasses two distinct scenarios, the particulars of which are detailed in Table. 3. The first scenario involves the Ego vehicle navigating a typically congested roadway, reminiscent of a shopping district thoroughfare. The second scenario is akin to the first, yet features a road with reduced traffic and fewer obstacles. These two scenarios are crafted using Matlab's Driving Scenario Designer application, which allows for the plotting of routes, the designation of the Ego vehicle as the central decision-making entity, and the inclusion of other elements such as vehicles, pedestrians, and barriers, categorized as actors in the simulation. A birds-eye view of the selected suite of ECUs is presented in Fig. 3 (a), which zooms in on a segment of the later Fig. 4 for more detailed visualization. Fig. 3 (b) is offered as an enhanced view to clarify the details in Fig. 3 (a), showing the positioning of ECUs on the Ego vehicle. Fig. 4 (a) captures the road scene prior to the activation of the ECUs on the Ego vehicle, depicting an unpopulated roadway. Conversely, Fig. 4 (b) illustrates the detection of various actors by the ECUs at the onset of the simulation, mapping the coverage area and pinpointing the initial detection moments.

Table. 3: More crowded road and less crowded road scenes

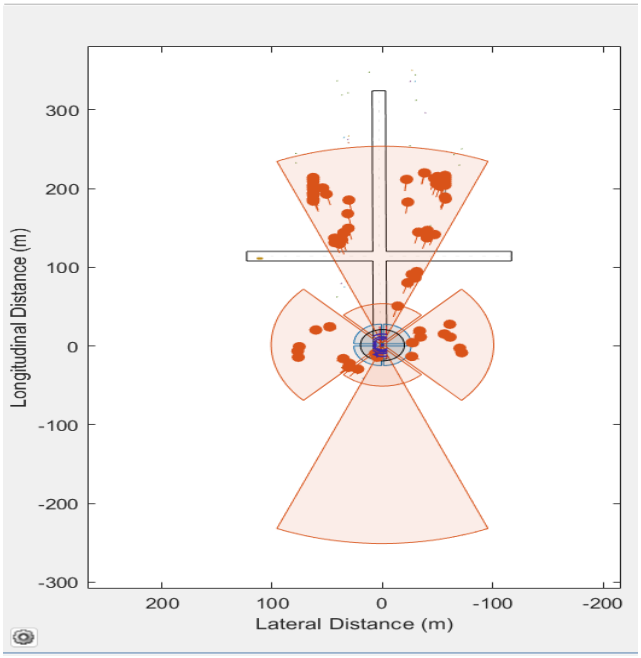
Scene Name	Number of Pedestrian	Number of Barrier	Number of vehicles
More Crowded	70	2	3
Less Crowded	24	0	2

4.2 Matlab/CANoe Integration

For the simulation of the internal ECUs, the CANoe program will be utilized exclusively. In contrast, the integration of CANoe with Matlab will facilitate the simulation of the external ECUs. Fig. 5 illustrates the integration strategy adopted in this research, tracing the process from the initial scene creation in Matlab—where the vehicle's external ECUs gather data about their surroundings—to the finale of the simulation where data from both external and internal ECUs are mapped onto the CAN bus.



(a) Before detection



(b) After detection

Fig. 4: Actors detected by ECUs

Table. 4: Specifications of the camera and radar that will be changed

Camera parameters	value	Radar parameters	value
Detection Range(meter)	25	Azimuth field of view (◦) Long range Mid range Short range	 45 90 90
Horizontal Field of View (◦)	90	azimuth resolution (◦)	4
Bit Rate(bps)	1M	Bit Rate(bps)	1M
Update Interval (m sec)	100	Update Interval (m sec)	100

Table. 5: The Specifications of Lidar ECU

LiDAR Sensor parameters			
Bit rate (Mbps)	10	Azimuth Resolution (◦)	0.4
Max Range(m)	50	Elevation Resolution (◦)	2
Range Accuracy (◦)	0.03	Elevation Limits (◦)	[-15 15]
Frames per Second (FPS)	5	Azimuth Limits (◦)	[-180 180]

Fig. 6 displays the network as simulated within the CANoe environment, featuring a network of 18 distinct ECUs. These units communicate with each other, exchanging messages across three different CAN bus varieties.

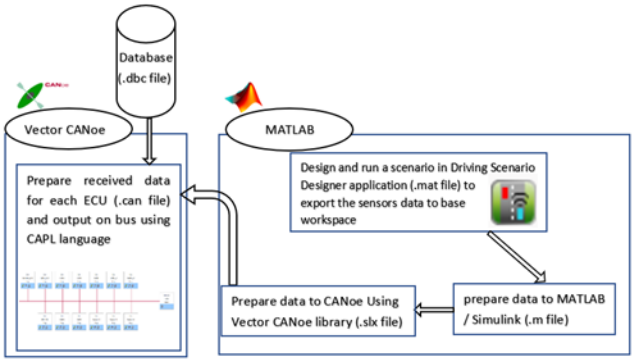


Fig. 5: The proposed scheme

4.3. Exteroceptive ECUs of AV's specifications

Table. 4 outlines the specifications of cameras and radars, each of which will be individually adjusted. Meanwhile, Table. 5 details the specific features of the LiDAR. The specifications for both the camera and radar are set based on their variable characteristics as referenced in [39-40]. This approach is taken to explore how alterations in ECUs impact the CAN bus. These adjustments are informed by prior results that demonstrated the individual effects of each ECU type on busload. With these insights, the internal network of the AV will be configured to efficiently support a variety of ECU types, ensuring simplicity in design. Additionally, the worst-case response time (WCRT) for the network will be determined.

Table. 6: Busload and FPS for camera ecus over classical can bus

		More Crowded		Less Crowded	
		Busload (%)	FPS	Busload (%)	FPS
Coverage Range (meter)	20	2.35	280	2.2	260
	25	2.4	285	2.25	270
	30	2.7	320	2.4	285
HFOV (deg)	60	1.3	155	1.15	135
	70	2.3	275	2.15	260
	90	2.4	285	2.25	270
Bit Rate (kbps)	125	19.15	285	17.8	270
	500	4.8	285	4.45	270
	1000	2.4	285	2.25	270
Update Interval (ms)	20	8	940	7.9	925
	50	7.6	890	7.05	830
	100	2.4	285	2.25	270

Table. 7: CAN simulation message set

ID	Type	[messages /ms]	Payload [bytes]	sender	ID	Type	[messages /ms]	payload [bytes]	sender	ID	Type	[messages / ms]	payload [bytes]	sender
1	p	0.02	8	3	28	p	0.01	8	1	55	p	0.0078	8	6
2	s	0.03	8	3	29	s	0.03	8	1	56	p	0.01	8	1
3	p	0.02	8	3	30	p	0.01	8	1	57	s	0.03	8	4
4	s	0.03	8	3	31	s	0.03	8	1	58	p	0.004	3	4
5	s	0.03	8	3	32	p	0.05	8	4	59	p	0.002	8	4
6	p	0.02	8	3	33	s	0.03	8	4	60	p	0.002	8	4
7	s	0.03	8	1	34	p	0.002	8	4	61	p	0.002	7	4
8	s	0.03	8	3	35	p	0.05	8	4	62	p	0.002	8	4
9	p	0.02	8	3	36	p	0.002	8	4	63	s	0.03	2	4
10	s	0.03	8	3	37	p	0.05	8	4	64	p	0.01	8	4
11	s	0.03	8	3	38	s	0.03	8	3	65	p	0.01	8	4
12	s	0.03	8	6	39	p	0.05	8	4	66	p	0.01	8	4
13	p	0.01	8	1	40	p	0.005	8	3	67	p	0.01	8	4
14	p	0.01	8	3	41	p	0.01	8	4	68	p	0.01	8	4
15	p	0.01	8	1	42	p	0.01	8	5	69	p	0.01	6	4
16	p	0.01	8	3	43	s	0.03	8	5	70	s	0.03	8	4
17	s	0.03	8	3	44	p	0.01	8	5	71	s	0.03	8	4
18	p	0.01	8	1	45	s	0.03	8	5	72	p	0.005	8	4
19	s	0.03	8	1	46	p	0.02	8	5	73	p	0.005	8	4
20	p	0.01	8	1	47	s	0.03	8	5	74	p	0.005	8	4
21	p	0.01	8	2	48	p	0.02	8	5	75	s	0.03	8	4
22	p	0.002	8	2	49	s	0.03	8	6	76	p	0.005	8	4
23	p	0.002	8	2	50	p	0.01	8	6	77	p	0.005	8	4
24	s	0.03	8	2	51	s	0.03	8	6	78	p	0.005	2	4
25	p	0.002	8	2	52	p	0.01	8	6	79	p	0.02	1	3
26	p	0.01	8	3	53	p	0.0078	8	6	80	p	0.01	2	3
27	S	0.03	8	3	54	s	0.03	8	6	81	p	0.005	2	3

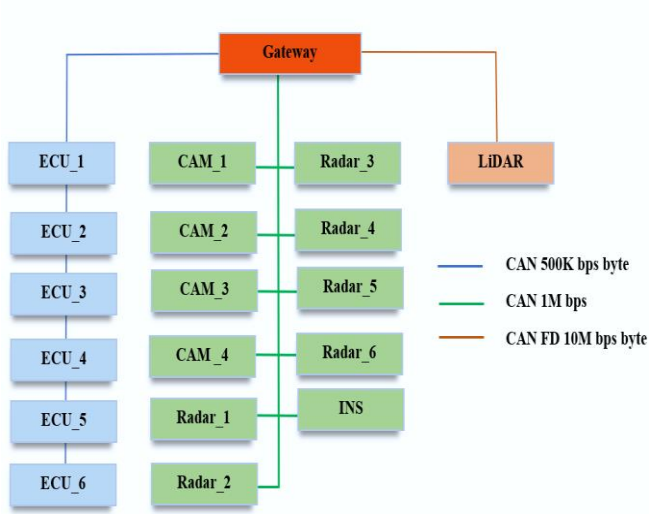


Fig. 6: Illustration of the simulated network

4.4. Internal ECUs of Autonomous Vehicles's specifications

Within the experimental vehicle setup, six internal ECUs are connected to a singular CAN bus, accommodating a total of 81 different message types as noted in [27], [41-43]. Table. 7 details the characteristics of these messages, distinguishing between the 27 aperiodic and 54 periodic ones.

5. Simulation scenarios and results

The performance of the ECUs (camera, radar, and INS) and internal ECUs on the classic CAN bus in terms of busload and frame per second (FPS) will be shown in separate tables, as will the effect of the

LiDAR on the CAN FD bus. In the final, based on what was concluded from the previous results, which showed how each type of ECU independently affected the busload, The CAN network of the AV will be configured to accommodate all ECU types with minimal complexity, the WCRT will also be calculated.

5.1 First scenario

In the first section, we will discuss the effect of four camera ECUs on CAN bus; in that scenario, each camera ECU will detect a maximum of 11 objects, which will be used to find objects. Each object detected by the camera ECU is determined by 6 variables (position (x, y, z), speed (x, y, z)), in addition to the 2 lanes, and each lane consists of 7 variables. Thus, the maximum number of messages per each camera is 80 frames per burst.

Table. 6 shows the effects of changing four camera properties these mentioned in Table. 4 on the busload and the number of FPS for less and more crowded scenes. In the this section, we will also discuss the effect of six radar ECUs; a radar ECU will detect a maximum of 20 objects, and 6 variables determine each detected object (position (x, y, z), speed (x, y, z)). Thus, the maximum number of messages per radar is 120 frames per burst. Table. 8 shows the effects of changing the four radar properties on the busload and the number of FPS.

The lidar ECU sends back the point as an array of positive [x, y, z] points with real values of m by n by 3. The number of elevation (vertical) channels and azimuthal (horizontal) channels in the point are m and n, respectively. m and n, as shown in the following equation, determine the number of points in the point set [33].

$$m \times n = \frac{V_{FOV}}{V_{RES}} \times \frac{H_{FOV}}{H_{RES}} \quad (1)$$

where

V_{FOV} is the vertical field of view (HFoV), in degrees.
V_{RES} is the vertical angular resolution, in degrees.
H_{FOV} is the horizontal field of view (VFoV), in degrees.
H_{RES} is the horizontal angular resolution, in degrees.

Table. 8: Busload and FPS for radar ECUS over classical CAN bus

		More crowded		Less crowded	
		Busload (%)	FPS	Busload (%)	FPS
Azimuth Resolution (deg)	2	41.51	5075	23.73	2898
	4	40.18	4907	22.12	2702
	6	39.9	4879	21.77	2660
HFOV (deg)	70	43.96	5376	29.26	3577
	90	40.18	4907	22.12	2702
	110	29.12	3542	14.07	1715
Bit Rate (kbps)	125	99.96	1085	99.96	1069
	500	68.36	4907	40.31	2702
	1000	40.18	4907	22.12	2702
Update Interval (ms)	20	99.96	4345	99.96	4177
	50	99.96	4334	59.92	2589
	100	40.18	4907	22.12	2702

Each m-by-n element in the array gives the x, y, and z coordinates of a detected point in the ego vehicle coordinate system. If the LiDAR fails to detect a point at a given coordinate, NaN values are returned for x, y and z, an 12-byte DLC value is used in the CAN FD of LiDAR.

Table. 9: Busload and FPS for lidar ECU over can FD bus

		More crowded		Less crowded	
		Busload(%)	FPS	Busload(%)	FPS
Coverage Range (meter)	30	56.70	10266	56.43	10216
	50	59.68	10803	59.13	10703
	70	59.86	10837	59.16	10728

Table. 10: Busload and FPS for internal ECU over CAN bus

	Bit Rate	Busload [%]	Std. Data [FPS]
CAN Bit Rate	125 Kbps	91.34	853
	500 Kbps	24.54	784
	1 Mbps	11.75	812

Table.11: CAN network characteristics for AV model development

Parameters	Network_1	Network_2	Network_3
CAN, CAN_FD	CAN	CAN	CAN_FD
Bit rate (bps)	500 k	1 M	10 M
DLC (byte)	1,2,3,6,7,8	4	12
ECUs count	6	11	1
ECUs	Internal ECUs	Camera, Radar and INS	LiDAR

```
// Save the current time whenever a message is transmitted
Now_Time = timeNow();
@cam1rt::rt.m1= Now_Time
// Whenever a message is received, calculate WCRT
if(this.id == ID)
{ Past_Time =@cam1rt::rt.m1
Now_Time = timeNow();
 difference_Time =Now Time - Past_Time;
if(difference_Time < worst_Time)
worst_Time = difference_Time;}
if (worst_Time!=0)
 write("msg_ID=xx worst_case_response_time=%d  ms ",
worst_Time /100);
```

Fig. 7: The code instrumentation to obtain the WCRT

Depending on Table. 5 and (1) , the LiDAR ECU will get about 13500 points per 200 milliseconds (ms), which cannot be transferred through the CAN bus, which is why the switch to CAN FD was made. Moreover, different coverage ranges were taken, as were the busload and the number of frame per second (FPS), as shown in Table. 9. When the range is 30 meters, we notice that both scenarios match the busload and the number of FPS. The difference appears slightly (because the range of the LiDAR cannot exceed 70 meters due to the characteristics of the bus) as the coverage of the LiDAR increases.

As for the INS ECU, at each time, it carries 5 parameters, and each parameter has (x, y, z) variables, so the maximum number of messages for the INS ECU (Orientation, Position, Velocity, Acceleration, and Angular Velocity) is 15 frames per burst. INS ECU depends on the vehicle's location, regardless of the environment around it. Therefore, we note that the busload is 0.73% and had 90 FPS in both scenarios.

5.2 Second Scenario (Internal ECUs)

In the second scenario, we will shows the effect of internal ECUs consisting of 6 ECUs to have 81 messages, Table. 10 shows the effect when three different CAN bit rates are used.

5.3 Third Scenario (Overall ECUs Effect)

Before, we looked at how the CAN bus was affected by the different features of each type of ECU. We will find the WCRT and look at what happens when we combine ECUs with different data rates. The CANoe data tool was used to get the busload and FPS. By measuring the time when a message is queued to be sent to the network by one ECU and the time when another ECU receives this message in the network, the Vector CAN programming language using CAPL can also be used to get the message reaction time. First, the time was recorded in a system variable after being measured with the time Now method. After the ECU got the message, the time was recorded again, and the time before the message was sent was taken away. It was set up to measure the WCRT with the code shown in Fig. 7.

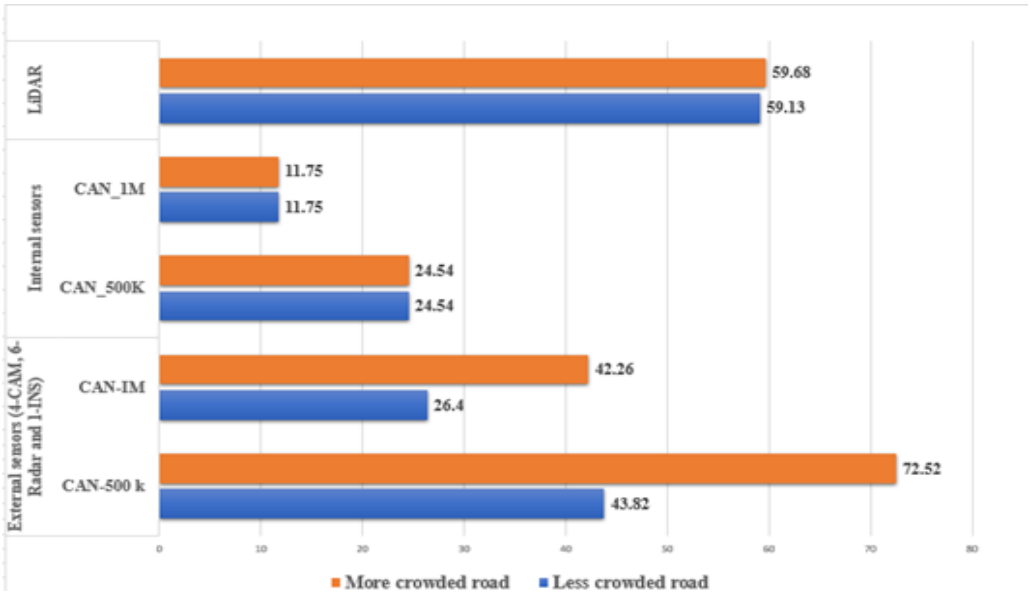


Fig. 8: Busload of the ECUs

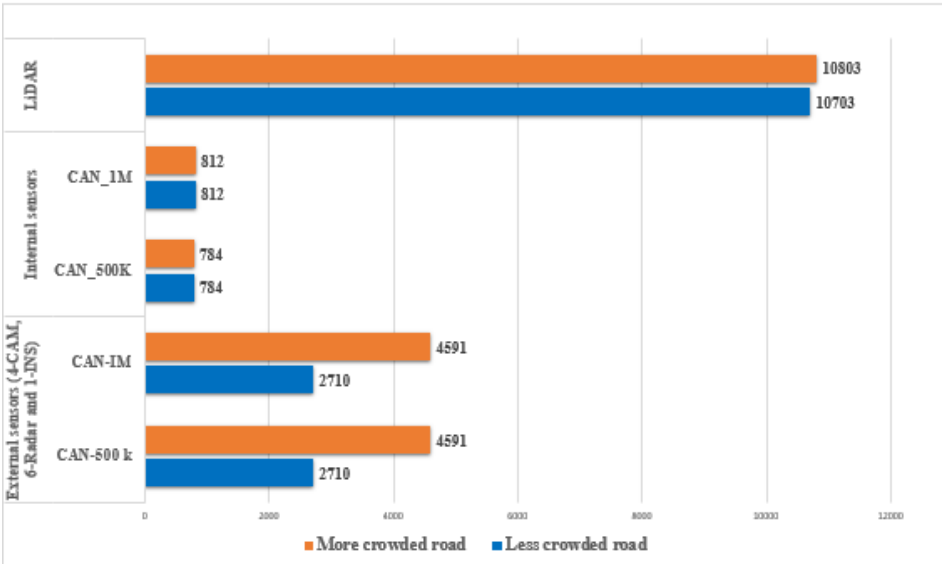
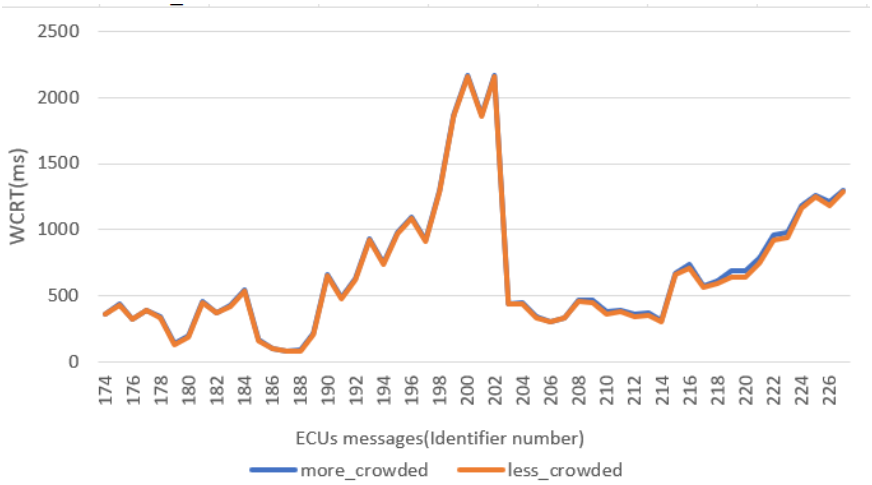
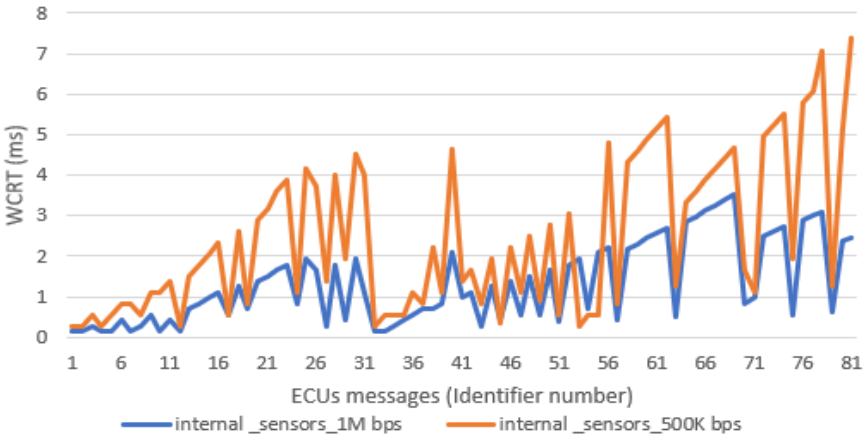


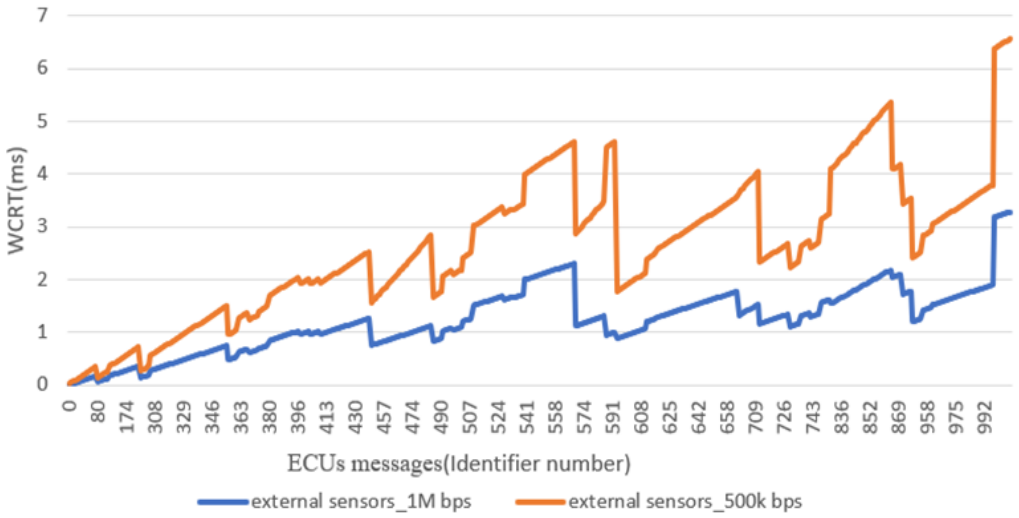
Fig. 9: FPS of the ECUs



(a) WCRT for LiDAR messages



(b) WCRT for internal ECUs messages



c) WCRT for external ECUs messages except LiDAR

Fig. 10: Worst-case response time for all ECUs messages over CAN buses

Fig. 8 and Fig. 9 shows the busload and the FPS of the ECUs in each bus for two different bit rates for both scenes, respectively. We looked at the effects of each type of ECU on its own and came to the conclusion that it is not possible to collect all of these effects on the car on a single bus network. This is because the bus load is 100% higher than its maximum capacity, so we had to use subnets and connect them with a gateway. It was easy to see that the 12-byte LiDAR external ECU worked well with the CAN FD network. With the old CAN, the camera, radar, and INS, among other external ECUs, worked well. The CAN FD works with the internal ECUs because it has an 8-byte message. In this way, the gateway will join the three subnetworks listed above. The CAN FD was good for sending messages inside the ECU. The CAN FD with the highest bit rate worked best for external ECUs like the LiDAR. The standard CAN worked best for the other external units. In the end, Fig. 10 shows how the WCRT is related to the importance of the ECUs' messages for three buses and two different bit rates. Fig. 10 shows that the WCRT drops when a message is sent that is more important than other messages on the network. It stresses how important the message's importance is. By looking at the final test network's results side by side, we saw that all but one of the ECUs fell within the acceptable ranges for busload or deadline delay (see Table. 2). The lidar, on the other hand, despite its few specifications, validated the data transfer rates via CAN FD; however, they failed to satisfy the access delay requirements.

Table. 12: Comparison among this work and previous related works

Issue	[24]	[25]	[26]	[27]	This work
Busload and WCRT	✓	✓	✓	✓	✓
Comparative with CAN FD	✓	– –	✓	✓	✓
SAE Internal Message sets	✓	✓	– –	✓	✓
Simulation tool	Matlab	CANoe with hardware	CANoe	OMNET++	Matlab/ CANoe
Simulation external ECUs	– –	– –	– –	– –	✓
Number of ECUs (4 or more)	>5	4	>5	>5	>5

6. Conclusion

This research introduces a basic autonomous vehicle design utilizing diverse CAN bus protocols. The design features a multi-protocol CAN bus system, fostering effective communication among the AV's components. Its scalable nature ensures compatibility with future advancements in autonomous vehicle technology. Table. 11 provides the suggested CAN network specifications for the development of this AV model, while Table. 12 compares this study's findings with relevant prior research. The results from the simulations confirm the effectiveness of the proposed design, underscoring its dependability and durability across a range of driving conditions. The

implementation of a multi-protocol CAN bus system significantly enhances communication efficiency within the AV, resulting in better overall performance and minimized delays. Throughout the design process, various challenges were encountered and overcome. Solutions implemented include the integration of redundant CAN Bus systems, optimization of ECU performance, and effective control of CAN Bus load. The design presented here lays a solid groundwork for the creation of more sophisticated AVs, leading the charge towards safer and more efficient transport solutions. It serves as a foundational model for the future development of AVs with more complex features like advanced driver assistance and complete autonomy. In summary, this paper makes a substantial contribution to the progression of AV technology. It underscores the critical role played by the adept utilization of various CAN bus protocols in the design of autonomous driving systems. The design proposed here offers a viable pathway to achieving autonomous transportation systems that are safe, efficient, and reliable

Conflict of Interest

The authors declared "No conflict of interest".

References

[1] I. H. Alexander, and Manzoor Khan "An overview of sensors in Autonomous Vehicles." Procedia Computer Science, Vol. 198, pp, 736-741, 2022.
https://doi.org/10.1016/j.procs.2021.12.315

[2] M. Hartstern, V. Rack and W. Stork, "Conceptual Design of Automotive Sensor Systems : Analyzing the impact of different sensor positions on surround-view coverage", *2020 IEEE Sensors*, Rotterdam, Netherlands, pp. 1-4, 2020.
doi: 10.1109/SENSORS47125.2020.9278638

[3] G. Velasco-Hernandez, D. J. Yeong, J. Barry and J. Walsh, "Autonomous Driving Architectures, Perception and Data Fusion: A Review," *2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing*, Cluj-Napoca, pp. 315-321, 2020.
doi: 10.1109/ICCP51029.2020.9266268

[4] A. Shoaib, F. Munir, A. M. Sheri, J. Kim, and M. Jeon "System, design and experimental validation of autonomous vehicle in an unconstrained environment", *Sensors*, Vol. 20, No. 21, art. no. 5999, 2020.
https://doi.org/10.3390/s20215999

[5] E. Azim, C. Wu and C. Sun "Research advances and challenges of autonomous and connected ground vehicles", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 22, No. 2, pp. 683-711, 2019.
https://doi.org/10.1109/TITS.2019.2958352

[6] W. Jiadai, J. Liu and N. Kato "Networking and communications in autonomous driving: A survey", *IEEE Communications Surveys & Tutorials*, Vol. 21, No. 2, pp. 1243-1274, 2018.
https://doi.org/10.1109/COMST.2018.2888904

[7] B. Gourav, K. Bhadane, R. K. Singh, R. Kumar, R. Aluvalu, R. Krishnamurthi, A. Kumar, R. N. Thakur, and S. Basheer "Autonomous vehicles and intelligent automation: Applications, challenges, and opportunities", *Mobile Information Systems*, Vol. 2022, pp. 1-36, 2022.
https://doi.org/10.1155/2022/7632892

[8] M. Vyas, H. Sarath, K. Smitha and A. Bagubali, "Modern automotive embedded systems with special mention to radars", *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, Bangalore, India, pp. 1618-1625, 2017.
https://doi.org/10.1109/RTEICT.2017.8256873

[9] C. Eunmin, H. Song, S. Kang and J. W. Choi. "High Speed, Low Latency In Vehicle Network Based on the Bus Topology for Autonomous Vehicles: Automotive Networking and Applications", *IEEE Vehicular Technology Magazine*, Vol. 17, No. 1, pp. 74-84, 2021.
https://doi.org/10.1109/MVT.2021.3128876

[10] R. Andrea, D. Wetzel, N. Balbierer, H. Meier, M. Niemetz, and S. Park "Comparative Analysis of CAN CAN FD and Ethernet for Networked Control Systems", In embedded world conference digital, 2021. [Cross Ref]

[11] J. D. Matthew, T. S. Stombaugh and S. A. Shearer "Controller area network based distributed control for autonomous vehicles", *Transactions of the ASAE*, Vol. 48, No. 2, pp. 479-490, 2005.
http://dx.doi.org/10.13031/2013.18312

[12] W. You, L. Fu, Y. Xu, F. Ma, and Y. Lu "Controller area network modeling and its application in cyber-physical power system co-simulation", *2018 37th Chinese Control Conference (CCC*), pp. 6178-6183, 2018.
https://doi.org/10.23919/ChiCC.2018.8483930

[13] C. Raghu and H. Malik "LiDAR data integrity verification for autonomous vehicle", *IEEE Access*, Vol. 7, pp. 138018-138031, 2019.
https://doi.org/10.1109/ACCESS.2019.2943207

[14] S. Jahromi, B. T. Tulabandhula and S. Ceti "Real-time hybrid multi-sensor fusion framework for perception in autonomous vehicles", *Sensors*, Vol. 19, No. 20, art.no. 4357, 2019.
https://doi.org/10.3390/s19204357

[15] M. Tarek, S. Zidi, A. Alabdulatif and M. Atiquzzaman "Comparative performance evaluation of intrusion detection based on machine learning in in-vehicle controller area network bus", *IEEE Access*, Vol. 9, pp. 99595-99605, 2021.
https://doi.org/10.1109/ACCESS.2021.3095962

[16] B. Lakhal, N. Mansour, O. Nasri, L. Adouane, and J. B. H. Slama "Controller area network reliability: overview of design challenges and safety related perspectives of future transportation systems", *IET Intelligent Transport Systems*, Vol. 14, No. 13, pp. 1727-1739, 2020.
https://doi.org/10.1049/iet-its.2019.0565

[17] Z. Weiying, M. A. S Khalid and S. Chowdhury "In-vehicle networks outlook: Achievements and challenges", *IEEE Communications Surveys & Tutorials*, Vol. 18, No. 3, pp. 1552-1571, 2016.
https://doi.org/10.1109/COMST.2016.2521642

[18] M. Hyeran, K. Han, and D. H. Lee "Ensuring safety and security in CAN-based automotive embedded systems: A combination of design optimization and secure communication", *IEEE Transactions on Vehicular Technology*, Vol. 69, No. 7, pp. 7078-7091, 2020.
https://doi.org/10.1109/TVT.2020.2989808

[19] Z. Zhang, Y. Cao, Z. Cui, W. Zhang and J. Chen "A Many-Objective Optimization Based Intelligent Intrusion Detection Algorithm for Enhancing Security of Vehicular Networks in 6G", *IEEE Transactions on Vehicular Technology*, Vol. 70, No. 6, pp. 5234-5243, 2021.
https://doi.org/10.1109/TVT.2021.3057074

[20] A. J. Michaels *et al*., "CAN Bus Message Authentication via Co-Channel RF Watermark", *IEEE Transactions on Vehicular Technology*, Vol. 71, No. 4, pp. 3670-3686, 2022.
https://doi.org/10.1109/TVT.2022.3143708

[21] G. Bruno and N. Navet "Fault confinement mechanisms on CAN: analysis and improvements", *IEEE transactions on vehicular technology*, Vol. 54, No. 3, pp. 1103-1113, 2005.
https://doi.org/10.1109/TVT.2005.844652

[22] C. Wonsuk, et al. "Identifying ecus using inimitable characteristics of signals in controller area networks." IEEE Transactions on Vehicular Technology, Vol. 67, No. 6, pp. 4757-4770, 2018.
https://doi.org/10.1109/TVT.2018.2810232

[23] L. Hansang, et al. "Quantitative analysis of ringing in a controller area network with flexible data rate for reliable physical layer designs", *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 9, pp. 8906-8915, 2019.
https://doi.org/10.1109/TVT.2019.2930110

[24] T. Mahmut, P. Oikonomidis, P. Charchalakis, and E. Stipidis "Modelling, simulation, and performance analysis of a CAN FD system with SAE benchmark based message set." *Proceedings of 15th International CAN Conference*, 2015. [Cross Ref]

[25] M. R. Vemparala, S. Yerabati and G. I. Mary, "Performance analysis of controller area network based safety system in an electric vehicle," *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology*, Bangalore, India, pp. 461-465, 2016.

https://doi.org/10.1109/RTEICT.2016.7807863

[26] Z. G. Marcon and E. P. de Freitas "A quantitative performance study on CAN and CAN FD vehicular networks", *IEEE Transactions on Industrial Electronics*, Vol. 65, No. 5, pp. 4413-4422, 2018.
https://doi.org/10.1109/TIE.2017.2762638

[27] H. Kim, W. Yoo, S. Ha and J. -M. Chung "In-Vehicle Network Average Response Time Analysis for CAN-FD and Automotive Ethernet", *IEEE Transactions on Vehicular Technology*, Vol. 72, No. 6, pp. 6916-6932, 2023.
https://doi.org/10.1109/TVT.2023.3236593

[28] Rishvanth, D. Valli, and K. Ganesan. "Design of an in-vehicle network (Using LIN, CAN and FlexRay), gateway and its diagnostics using vector CANoe", *American Journal of Signal Processing*, Vol. 1, No. 2, pp. 40-45, 2011.
https://doi.org/10.5923/j.ajsp.20110102.07

[29] K. Sukhwan, J. Seong, and M. Lee "Controller area network with flexible data rate transmitter design with low electromagnetic emission", *IEEE Transactions on Vehicular Technology*, Vol. 67, No. 8, pp. 7290-7298. 2018.
https://doi.org/10.1109/TVT.2018.2832659

[30] T. Chong, T. Liu, Y. Zhang, C. Ma, X. Jia and Z. Wu, "Analysis of the influence of CAN bus encryption and decryption on real time performance", *2021 2nd International Conference on Computer Communication and Network Security (CCNS)*, Xining, China, pp. 38-44, 2021.
https://doi.org/10.1109/CCNS53852.2021.00016

[31] G. Xie *et al*., "WCRT Analysis of CAN Messages in Gateway-Integrated In-Vehicle Networks," *IEEE Transactions on Vehicular Technology*, Vol. 66, No. 11, pp. 9623-9637, 2017.
https://doi.org/10.1109/TVT.2017.2737035

[32] J. Walrand, M. Turner and R. Myers "An Architecture for In-Vehicle Networks", *IEEE Transactions on Vehicular Technology*, Vol. 70, No. 7, pp. 6335-6342, July 2021.
https://doi.org/10.1109/TVT.2021.3082464

[33] Matlab. Automated driving toolbox. https://www.mathworks.com/products/automated-driving.html

[34] Y. D. Jong, G. V. Hernandez, J. Barry, and J. Walsh "Sensor and sensor fusion technology in autonomous vehicles: A review", *Sensors*, Vol. 21, No. 6, art. no. 2140, 2021.
https://doi.org/10.3390/s21062140

[35] Autonomous Vehicles Cannot Be Test-Driven Enough Miles to Demonstrate Their Safety; Alternative Testing Methods Needed. [Cross Ref]

[36] K. Nidhi, and S. M. Paddock "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?", *Transportation Research Part A: Policy and Practice*, Vol. 94, pp. 182-193, 2016.
https://doi.org/10.1016/j.tra.2016.09.010

[37] L. Yao, J. Wu, Y. Wang and C. Liu, "Research on vehicle integrated control algorithm based on MATLAB and CANoe co-simulation", *2014 IEEE Conference and Expo Transportation Electrification Asia-Pacific (ITEC Asia-Pacific)*, Beijing, China, pp. 1-5, 2014.
https://doi.org/10.1109/ITEC-AP.2014.6941219

[38] https://www.vector.com/int/en/products/products-az/software/canoe/#

[39] I. H. Alexander, and M. Khan "An overview of sensors in Autonomous Vehicles", *Procedia Computer Science*, Vol. 198, pp. 736-741, 2022.
https://doi.org/10.1016/j.procs.2021.12.315

[40] A. Kumar, K. Stephen and A. S. Sabitha, "A Systematic Review on Sensor Fusion Technology in Autonomous Vehicles," *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, pp. 42-48, 2023.
doi: 10.1109/ICESC57686.2023.10193038

[41] S. Mubeen, J. Maki-Turija, and M. Sjodin, "Extending Worst Case Response-Time Analysis for Mixed Messages in Controller Area Network With Priority and FIFO Queues", *IEEE Access*, Vol. 2, pp. 365-380, 2014.
https://doi.org/10.1109/ACCESS.2014.2319255

[42] K. Tindell, A. Burns, A Wellings "Guaranteeing message latencies on control area network (CAN)", *IFAC Proceedings Volumes*, Vol. 27, No. 15, pp. 35-40, 1994.
https://doi.org/10.1016/j.tra.2016.09.010

[43] I. A. Qutaiba "Tele-operated Vehicles System Using WLAN and Industrial Ethernet Techniques", *Al-Rafidain Engineering Journal (AREJ),* Vol. 16, No. 5, pp. 33-42, 2005.
https://doi.org/10.33899/rengj.2008.44856

***