All Sciences Proceedings

$1^{st}$ International Conference on Scientific and Innovative Studies

April 18-20, 2023 : Konya, Turkey

ICSIS

All Sciences Proceedings
http://as-proceeding.com/

https://as-proceeding.com/index.php/icsis

# A CAN Bus-Based Efficient Design for a Simple Autonomous Vehicle (AV)

Zeina Ali [*] and Qutaiba I. Ali [2]

[1] Computer and Information Engineering Department, Ninevah University, Iraq
[2] Computer Engineering Department, University of Mosul, Iraq

[*]zinah.mohammed@uoninevah.edu.iq

***Abstract*** **–** The development of autonomous vehicles (AVs) has gained significant attention in recent years due to their potential benefits in terms of increased safety, reduced traffic congestion, and improved energy efficiency. Various types of Controller Area Network (CAN) bus protocols in AVs are crucial for their proper functioning. This paper presents an efficient design of a basic autonomous vehicle based on various CAN bus types. The proposed design incorporates a multi-protocol CAN bus system that enables efficient communication between the different components of the AV. The system is designed to be scalable and adaptable to future advancements in autonomous driving. The paper also discusses the challenges faced during the design process and the solutions to overcome them. Simulation results demonstrate the effectiveness of the proposed design, highlighting its reliability and robustness in various driving scenarios. Overall, the proposed design can be a foundation for developing more advanced AVs and pave the way toward safer and more efficient transportation systems.

*Keywords – Autonomous Vehicles, CANoe, Controller Area Network, Electronic Control Unit, Matlab*

## I. INTRODUCTION

A vehicle that operates without human input or control is autonomous. Autonomous vehicles (AVs) can sense their surroundings using a combination of cameras, LiDAR, radar, and other electronic control units (ECUs), and then use these data to navigate and make decisions in real time [1][2].Several authors have proposed architectural models for the technical and functional design, development, and deployment of AV systems to facilitate a successful product development life cycle and eliminate the need for each AV development to reinvent the wheel. [3].

From a functional viewpoint, AVs are made up of functional or logical blocks, which are defined by the information flow and processing phases from data collecting to the vehicle control system, including the internal monitoring of the vehicle. Perception, planning and decision, motion and vehicle control, and system supervision are the four main functional blocks that can be found in the majority of the proposed architectures and solutions from both academia and industry[1][3][4][5][6].The four primary functional modules of the AV system process ECU-collected vehicular data, as shown in Fig. 1.

In the perception stage, ECUs collect data about the vehicle's environment, which is transmitted via the inra_vehicle_network (IVN) such as Controller Area Network (CAN) bus to the AV's central computing unit. In the planning and decision-making stage, the AV's computing unit uses artificial intelligence algorithms to analyze the data and make decisions about the vehicle's movements[7]. Finally, in the control stage, the AV's control system uses the CAN bus to execute the planned movements by sending commands to

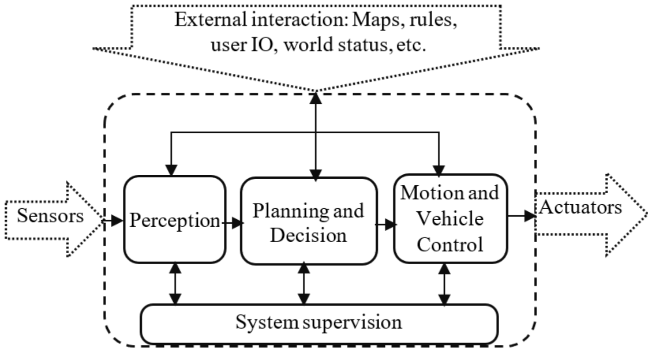


Fig. 1. Main functional modules of the AV system

various components, such as the engine, brakes, and steering system. The CAN bus plays a critical role in enabling real-time communication between these stages, allowing the AV to operate safely and efficiently.

The primary objective of the perception stage in AVs is to accurately and reliably perceive the surrounding environment through a combination of ECUs such as cameras, LiDAR, radar, and other ECUs. It is responsible for identifying and tracking objects and obstacles such as other vehicles, pedestrians, cyclists, road signs, lane markings, and traffic signals in real-time. It also helps in estimating the speed, distance, and direction of these objects to enable safe and effective decision making by the autonomous vehicle[3][5].

The CAN is a type of asynchronous serial bus multi-master protocol in which all ECUs have equal access rights and is widely used for in-vehicle communication [6]. It is the most famous bus standard for automotive applications . The protocol is on the lower two OSI layers (physical and data link protocol). The CAN bus has an arbitration method based on the CSMA/CD (Carrier Sense Multiple Access with Collision Deterction) with the AMP technique (supporting Arbitration on Message Priority) [8].

The CAN bus versions are classical CAN, CAN FD (Flexible Data Rate), and CAN XL(eXtra Large). Classical CAN (ISO11898-1), reliably connects ECUs at 1 Mbps. CAN FD, known as ISO 11898-2, is a newer CAN bus that supports 8 Mbps data transfer. The latest CAN bus, ISO 11898-3 or CAN XL, is under development. It supports data transfer rates up to 10 Gbps[9][10].

The CAN bus plays a crucial role in connecting and communicating with both the external and internal ECUs in an AV [11][12][13]. External ECUs such as cameras, lidars, radars, and GPS systems provide critical data about the vehicle's surroundings, while internal ECUs such as accelerometers and temperature ECUs provide information about the vehicle's internal environment. The CAN bus enables the real-time exchange of data between these ECUs and the AV's central computer. The AV's central computer uses machine learning algorithms and other technologies to analyze the information and make decisions about the vehicle's trajectory, speed, and other parameters [5][14][15]. This communication is critical for the safe operation of an AV, as it relies on the accuracy and timeliness of ECU data to make decisions about its movements.

As a result of the fact that a modern intra-vehicle communication system is almost always composed of a multiple subnetworks operating with a variety of communication protocols, automotive gateways, which serve as the interfaces between the various subnetworks [16], are vital to the intra-vehicle communication network overall and should under no circumstances be disregarded [17].

In a communication system, an automotive gateway can typically perform one of three different functions. In order to facilitate the transmission of data across various subnetworks, it may first work as a protocol bridge. The most traditional function of a gateway is also this one. Second, it can be used to "expand" network bandwidth, where the gateway is connected to other subnetworks of the same protocol in order to prevent one network section from becoming overloaded. Thirdly, a gateway can serve as a firewall, preventing unauthorized attempts to access the network from the outside and reducing undesirable disruptions. [17].

Relevant work examining the impact of CAN bus performance and comparing it with IVN is summarized in Table 1. The following are ways in which this paper contributes:

1) Designing a basic model of the AV based on different CAN bus standards by investigating the impact of the AV's internal and external ECUs on the CAN bus

2) Adapting the topology of the CAN network

Table 1. Summary of Existing Studies on CAN Bus Performance

| **Reference** | **Focus** | **Scenario** | **Verification** |
|---|---|---|---|
| [18] | Measure the performance of a CAN bus | CAN-FD-8-byte and CAN networks were examined. Four ECUs exchange more than 40 messages in the simulation network. CAN-FD outperformed CAN for network message busload and WCRT. | Simulation using CANoe |
| [19] | | The authors measured CAN and CAN FD bus transmission times for a network of two ECUs, one sending and one receiving data. Data bytes and messages were correlated. CAN FD outperformed CAN when flashing large amounts of data to the ECU. | Simulation using CANoe |
| [20] | | A CAN FD network that sent time- and event-based SAE benchmark message set was examined. The CAN FD protocol improves real-time control system message delay and bus utilization in simulations. | Simulation using MATLAB |
| [21] | | FSAE standards were used to measure busload and response time for four electric vehicle ECUs. | Simulation using CANoe with hardware |
| [22] | | J1939 and ISO 11783 were used to simulate and analyze an agricultural machine vehicle network. They compared CAN and CAN FD network performance (8 and 64 bytes). Three ECUs and a virtual terminal comprised the simulated network. CAN FD had lower busload, WCRT, and jitter than CAN. | Simulation using CANoe |
| [23] | | Queuing analysis-based CAN, CAN-FD, and Automotive Ethernet analytical models estimated response time. The 81-message CAN bus connects six ECUs in CAN and CAN FD. | Simulations using OMNET++ |
| [24] | Connecting with other IVN using the gateway | The authors created three communication protocols—LIN, classical CAN, and FlexRay—and a gateway protocol to transfer messages between them. | Simulation using CANoe with Hardware |
| [25] | | The authors investigated how CAN bus structure affects transmission performance. The multi-level-bus CAN network with a gateway reduced busload and message delay in simulations. | Simulation using CANoe |
| [26] | Performance comparison with other IVN | Ring and star network topologies were compared by busload for four ECUs. The star topology outperformed the ring topology. | Simulation using CANoe |
| [27] | | The authors compared FlexRay and CAN-Bus latency. CAN-Bus is better for hard-time systems, and FlexRay is good for low-priority deterministic data transfer. | Hardware using HSC12 microcontrollers |
| [28] | | Classical CAN network performance was analyzed. Compared to classical CAN+LIN, CAN has a 2.7% faster average transmission speed. | Simulation using CANoe |

based on the performance findings that were received, which led to an increase in performance and a decrease in delay

3) Giving the recommended characteristics of a multi-protocol CAN bus system for AV model development.

The following is the rest of this paper: Section II briefly explains the AV external ECUs and Research methodology is presented in Section III. simulation scenarios and results details in Section IV. Finally, concluding remarks are presented in Section V.

## II. AV EXTERNAL ECUs

ECUs quantify the sensed events or environmental changes for subsequent processing. ECUs must provide a perceptive and locational view of the environment so the vehicle can make real-time decisions.

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.

### A. *Camera*

Vision Detection Generator ECU blocks are composed of a camera ECU and a monocular camera ECU. The configuration data for a camera comprises both intrinsic and extrinsic parameters. (The camera's focal length and optical center) and extrinsic parameters (pitch, yaw, and roll) [29].

### B. *Radar*

Radio Detection and Ranging, or RADAR, is operated by producing electromagnetic waves within the area of interest and receiving scattered waves (or reflections) from targets to analyze signals and determine range information. It estimates the relative speed and location of known obstacles using the Doppler effect of electromagnetic waves [30].

Medium-Range Radar (MRR), Long-Range Radar (LRR), and Short-Range Radar (SRR) are the three main categories of vehicle radar systems .

### C. *INS*

An inertial navigation system (INS) is one of the positional ECUs used to calculate a vehicle's pose (position and orientation) and velocity.

### D. *LiDAR*

LiDAR is an acronym that stands for Light Detection and Ranging. LiDAR works by using rotating lenses to produce laser or infrared light pulses and measuring how long it takes for each light pulse to reflect.

## III. RESEARCH METHODOLOGY

The AV's external ECUs derive their data from the vehicle's surrounding environment, which is configured using Driving Scenario Designer application in Matlab. Then, these data is transferred to CANoe to study its combined effect with the data of the AV's internal ECUs on the CAN bus.

This section explains the requirements that need to be prepared for the proposed network, including the roads that will be simulated in Matlab and the proposed scheme for integrating the Matlab/CANoe programs, in addition to the specifications of the external ECUs and internal messages.

### A. *Roads Traffic Map*

Fig. 2 generally describes the situation in terms of the vehicles. It shows two roads that cross, the blue Ego vehicle and two other yellow and red vehicles. Each vehicle has its own set of waypoints, as well as the speed (in meters per second) and wait time (in seconds) for each one. As shown in the scene, the barriers are installed, and the pedestrians are scattered on both sides of the road.

Table 2 shows the types and numbers of external ECUs used in addition to the maximum number of detection objects for the camera and radar. Table 3 represents the delay deadline communication requirements of each ECU [9].

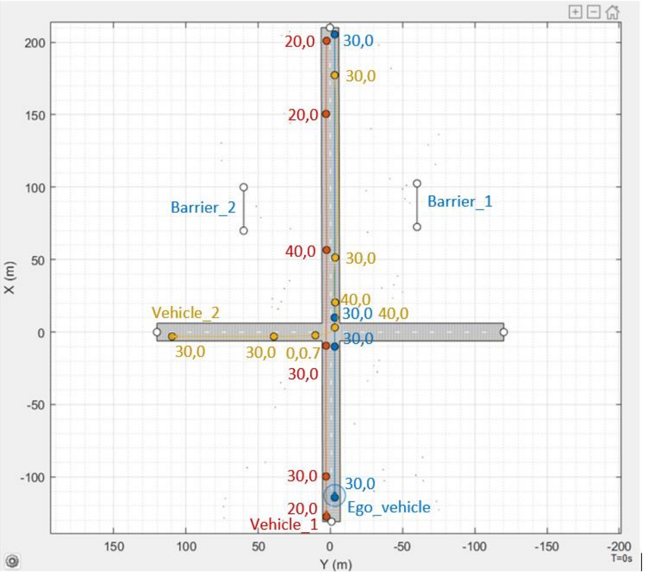


Fig. 2. General description of the scene

Two scenes will be simulated to test the system. Their specifications of them are shown in Table 4. In the first scene, the ego's vehicle travels along a busy road (like a mall road or something similar).

Table 2. 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 3. delay deadline requirements for AVs

| **ECU** | **Delay Deadline (ms)** |
|---|---|
| Lidar | 10 |
| Radar | 10 |
| Camera | 10 |
| Normal control | 5~50 |
| Critical control | 0.1 |

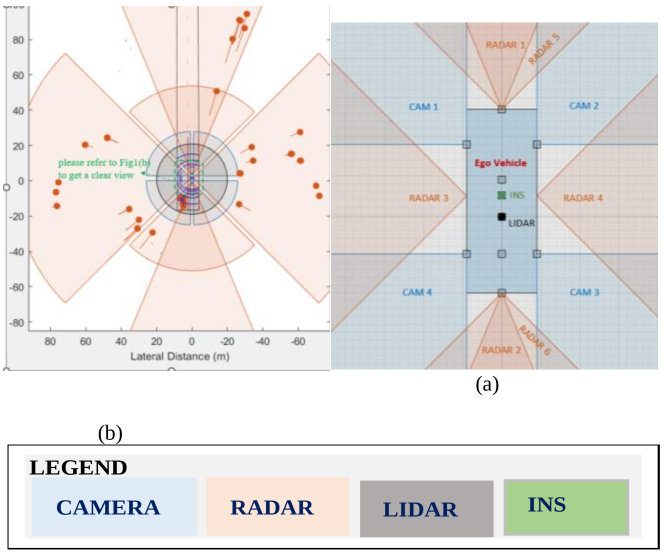
Table 4. More crowded road and less crowded road scenes

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

The second scene is similar to the first, but the road is less busy (fewer objects).

The simulation's two scenes will be created using Matlab's Driving Scenario Designer application. It enables us to identify the route, the ego vehicle that contains the decision-making agent, and the actors, which include all other participants in the situation. The probable actors are grouped into three categories: vehicles, pedestrians, and barriers.

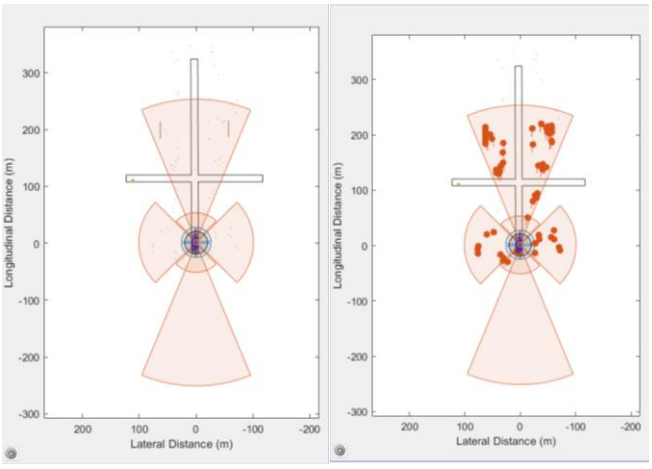
A Birds-eye plot of the selected ECU suite is shown in Fig. 3(a) (a close-up of Fig. 4 that will be illustrated later) that represents the general description of the scene, Fig. 3(b) is included to provide a clearer view of Fig 3 (a). ECU positioning and placement can be found on the Ego vehicle.



(a)

(b)

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



a)before detection

b)after detection

Fig. 4. Actors detected by ECUs

Fig. 4(a) shows how the road looked before the ECUs found the actors on the ego vehicle. Fig. 4(b) shows the actors' detection points within the ECUs' coverage area, as an example, at the moment of the beginning of running the driving scene.

## B. *Matlab/CANoe Integration*

When simulating the internal ECUs, only the canoe program will be used, whereas the CANoe/Matlab integration will be used when simulating the external ECUs.. Fig. 5 shows the scheme proposed in this work and the integration relationship between Matlab/CANoe, starting from creating the scene in Matlab in order for the vehicle's external ECUs to obtain their data from the surrounding environment, and ending with placing external and internal ECUs data on the CAN bus.

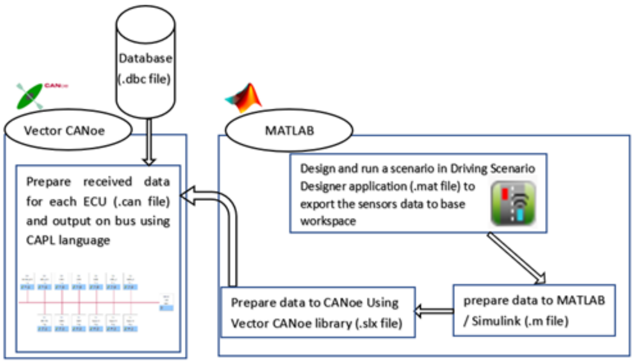


Fig. 5. The proposed scheme

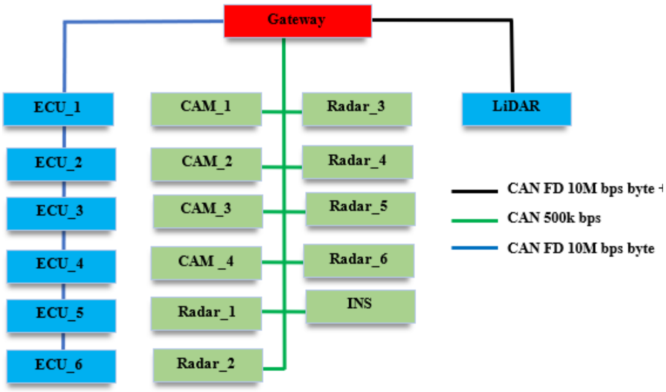


Fig. 6. Illustration of the simulated network

Fig. 6 shows the simulated network in the CANoe environment, which contains 18 different ECUs. Those ECUs exchange messages over three types of CAN buses.

TABLE 5. Specifications of the camera and radar that will be changed

| Camera parameters | value | Radar parameters | value |
|---|---|---|---|
| Detection Range(meter) | 25 | Azimuth field of view(◦)<br>Long range<br>Mid range<br>Short range | 45<br>90<br>90 |
| Horizontal Field of View (◦) | 90 | azimuth resolution (◦) | 4 |
| Bit Rate(bps) | 500 000 | Bit Rate(bps) | 500 000 |
| Update Interval (m sec) | 100 | Update Interval (m sec) | 100 |

Table 6. The specifications of LiDAR ecu

| LiDAR Sensor parameters | | | |
|---|---|---|---|
| Bit rate(Mbps) | 10 | AzimuthResolution(◦) | 0.4 |
| Max Range(m) | 20 | ElevationResolution(◦) | 2 |
| RangeAccuracy(◦) | 0.03 | ElevationLimits(◦) | [-15 15] |
| Frames per Second (FPS) | 5 | AzimuthLimits(◦) | [-180 180] |

### *C. Exteroceptive ECUs of Autonomous Vehicles's specifications*

Table 5 shows the characteristics of cameras and radars, which we will modify one by one and Table 6 shows the detailed specifications of the LiDAR . We set camera and radar specifications based on their changeable specifications [35,36]. We do this for the purpose of studying the effect of changing ECUs on the CAN bus. We set on what was

TABLE 7. CAN-FD 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 |

concluded from the previous results, which showed how each type of ECU independently affected the busload, The intranetwork of the AV will be configured to accommodate all ECU types with minimal complexity, the worst case response time (WCRT) will also be calculated.

### *D. Internal ECUs of Autonomous Vehicles's specifications*

In the experimental vehicle there are six internal ECUs[1] are connected to a single CAN bus in the experimental vehicle to support 81 message types [23] [37]. Table 7 shows the attributes of these messages, of which 27 are aperiodic and 54 are periodic.

## IV. SIMULATION SCENARIO AND RESULTS *(OVERALL ECUs EFFECT)*

The CAN network of the AV will be configured to accommodate all ECU types with minimal complexity, the WCRT will also be calculated. In this scenario, we will measure the WCRT and study the effect of the combined ECUs with various data rates .

The busload and FPS were obtained using the CANoe statistics tool. The Vector CAN programming language using CAPL , it is also feasible to acquire the message response time by collecting the time measurement when a message is queued to be delivered to the network by one ECU and another time measurement when another ECU receives this message in the network. First, the time was measured using the *timeNow*() function (which returns simulation time in 10 microseconds) and saved in a system variable. Then, when the ECU got the message, the time was measured again, and the time from when the message sent was subtracted. The code shown in Fig. 7 was used to set up the program to measure the WCRT.

```
// 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();
 differance_Time =Now Time - Past_Time;
if(differance_Time < worst_Time)
worst_Time = differance_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

We conclude that it is not possible to collect the effect of all ECUs on the vehicle on one bus network because the bus load exceeds its maximum capacity by 100%, which necessitated us to use subnets and connect them using the gateway, We noticed that the LiDAR external ECU, which had a 24-byte payload, worked well with the CAN FD network. The other external ECUs, such as the camera, radar, and INS, worked well with the traditional CAN. With a payload of 8 bytes, the CAN FD suits the internal ECUs. Thus, the three subnetworks above will be connected using the gateway.

Table 8 and Table 9 shows the busload and the FPS of the ECUs in each bus for two different bit

TABLE 8. Busload of all ecus over various CAN bus

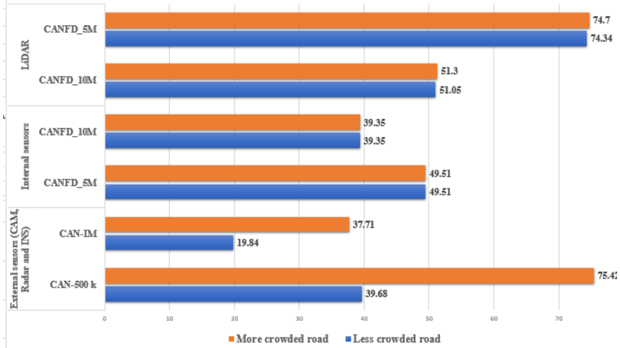
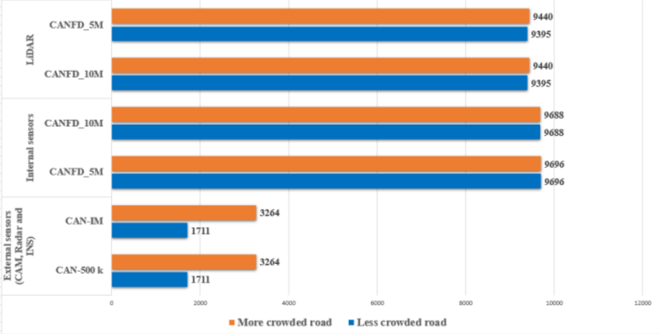


TABLE 9. FPS of all ECUs over various CAN bus

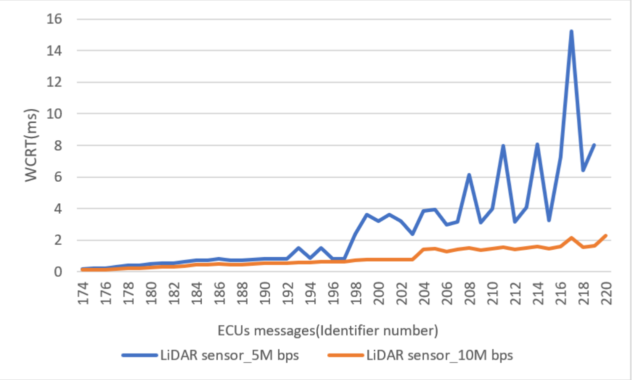


---

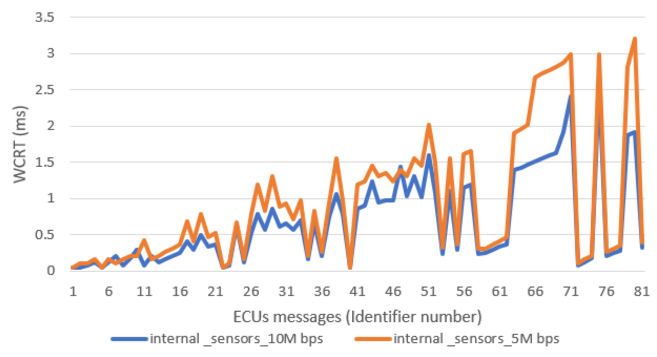
[1] ECU_1 to ECU_6 represent the different internal ECUs in electric vehicles, such as: the vehicle controller (V/C), the batteries (Battery), the inverter/motor controller (I/MC), brakes (Brakes), the instrument panel display (INS) and the transmission control (Trans)[38][39]

rates for both scenes, respectively. The CAN FD was appropriate for internal ECU messages. For the external ECUs, like the LiDAR, the CAN FD with the highest bit rate was the right choice, while the traditional CAN was appropriate for the other external units.

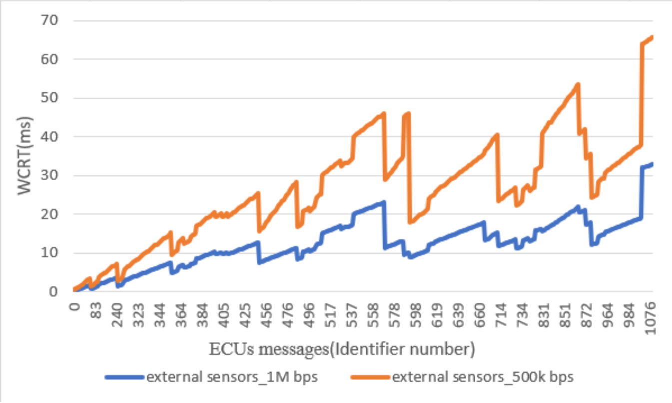
Finally, Fig. 8 illustrates the relationship between the priority of the ECUs' messages for three buses and two different bit rates and the WCRT. As shown in Fig. 8, the WCRT drops when a message with a higher priority than other messages on the network is sent. It emphasizes the significance of the the message's priority. By comparing the results obtained for the final simulation network, all ECUs were within the acceptable limits of busload or deadline delay (depending on Table 3), except the lidar, which breached the acceptable limit when using CAN FD with a 5 Mbps bit rate. However, in the case of using 10 Mbps, the delay was within acceptable limits.



(a) WCRT for LiDAR messages



(b) WCRT for internal ECUs messages



c)WCRT for external ECUs messages except LiDAR

Fig. 8. worst-case response time for all ECUs messages over CAN

## V. CONCLUSION

This paper presents an efficient design of a basic autonomous vehicle based on various CAN bus protocols. The proposed design incorporates a multi-protocol CAN bus system that enables efficient communication between the different components of the AV. The system is designed to be scalable and adaptable to future advancements in autonomous driving. Table 10 lists the recommended CAN network characteristics for AV model development. The comparison of our work with the most relevant previous research is shown in Table 11.

The simulations' results demonstrate the proposed design's effectiveness, highlighting its reliability and robustness in various driving scenarios. The use of a multi-protocol CAN bus system enhances the efficiency of communication between different components of the AV, leading to improved performance and reduced latency. The challenges faced during the design process were addressed by adopting appropriate solutions. These include using redundant CAN Bus systems, fine tuning the ECUs performance, and the CAN Bus load control mechanisms.

The proposed design can be a foundation for developing more advanced AVs, paving the way toward safer and more efficient transportation systems. It can be used as a basis for the development of future AVs with enhanced

TABLE 10. CAN network characteristics for AV model development

| Parameters | Network_1 | Network_2 | Network_3 |
|---|---|---|---|
| CAN , CAN_FD | CAN_FD | CAN | CAN_FD |
| Bit rate (bps) | 5 M | 500 K | 10 M |
| DLC (byte) | 8 | 8 | 24 |
| Number of ECUs | 6 | 11 | 1 |
| ECUs | Internal ECUs | Camera , Radar and INS | LiDAR |

TABLE 11. Comparison among this work and previous related works

| Issue | [18] | [20] | [21] | [22] | [23] | This work |
|---|---|---|---|---|---|---|
| Busload and WCRT | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Comparative with CAN FD | ✓ | ✓ | _ _ | ✓ | ✓ | ✓ |
| SAE Internal Message sets | _ _ | ✓ | ✓ | _ _ | ✓ | ✓ |
| Simulation tool | CANoe | Matlab | CANoe with hardware | CANoe | OMNET++ | Matlab/ CANoe |
| Simulation external ECUs | _ _ | _ _ | _ _ | _ _ | _ _ | ✓ |
| Number of ECUs (4 or more) | 4 | >5 | 4 | >5 | >5 | >5 |

functionalities, such as advanced driver assistance systems and full autonomy.

Overall, the design presented in this paper offers a significant contribution towards the development of AVs, highlighting the importance of the efficient use of various types of CAN bus protocols in designing autonomous driving systems. The proposed design provides a promising solution for achieving safe, efficient, and reliable autonomous transportation systems.

## REFERNCES

[1] Ignatious, Henry Alexander, and Manzoor Khan. "An overview of sensors in Autonomous Vehicles." Procedia Computer Science 198 (2022): 736-741.

[2] Hartstern, Maike, Viktor Rack, and Wilhelm Stork. "Conceptual Design of Automotive Sensor Systems: Analyzing the impact of different sensor positions on surround-view coverage." 2020 IEEE SENSORS. IEEE, 2020.

[3] Velasco-Hernandez, Gustavo, John Barry, and Joseph Walsh. "Autonomous driving architectures, perception and data fusion: A review." 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP). IEEE, 2020.

[4] Azam, Shoaib, Farzeen Munir, Ahmad Muqeem Sheri, Joonmo Kim, and Moongu Jeon. "System, design and experimental validation of autonomous vehicle in an unconstrained environment." Sensors 20, no. 21 (2020): 5999.

[5] Eskandarian, Azim, Chaoxian Wu, and Chuanyang Sun. "Research advances and challenges of autonomous and connected ground vehicles." IEEE Transactions on Intelligent Transportation Systems 22.2 (2019): 683-711.

[6] Wang, Jiadai, Jiajia Liu, and Nei Kato. "Networking and communications in autonomous driving: A survey." IEEE Communications Surveys & Tutorials 21.2 (2018): 1243-1274.

[7] Bathla, Gourav, Kishor Bhadane, Rahul Kumar Singh, Rajneesh Kumar, Rajanikanth Aluvalu, Rajalakshmi Krishnamurthi, Adarsh Kumar, R. N. Thakur, and Shakila Basheer. "Autonomous vehicles and intelligent automation: Applications, challenges, and opportunities." Mobile Information Systems 2022 (2022).

[8] Vyas, Manu, 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). IEEE, 2017.

[9] Choi, Eunmin, Hoseung Song, Suwon Kang, and Ji-Woong Choi. "High-Speed, Low-Latency In-Vehicle Network Based on the Bus Topology for Autonomous Vehicles: Automotive Networking and Applications." IEEE Vehicular Technology Magazine 17, no. 1 (2021): 74-84.

[10] Reindl, Andrea, Daniel Wetzel, Norbert Balbierer, Hans Meier, Michael Niemetz, and Sangyoung Park. "Comparative Analysis of CAN CAN FD and Ethernet for Networked Control Systems." In embedded world conference digital. 2021.

[11] Darr, Matthew J., Timothy S. Stombaugh, and Scott A. Shearer. "Controller area network based distributed control for autonomous vehicles." Transactions of the ASAE 48.2 (2005): 479-490.

[12] Wu, You, Lijun Fu, Yinan Xu, Fan Ma, and Yao Lu. "Controller area network modeling and its application in cyber-physical power system co-simulation." In 2018 37th Chinese Control Conference (CCC), pp. 6178-6183. IEEE, 2018.

[13] Changalvala, Raghu, and Hafiz Malik. "LiDAR data integrity verification for autonomous vehicle." IEEE Access 7 (2019): 138018-138031.

[14] Shahian Jahromi, Babak, Theja Tulabandhula, and Sabri Cetin. "Real-time hybrid multi-sensor fusion framework for perception in autonomous vehicles." Sensors 19.20 (2019): 4357.

[15] Moulahi, Tarek, Salah Zidi, Abdulatif Alabdulatif, and Mohammed Atiquzzaman. "Comparative performance evaluation of intrusion detection based on machine learning in in-vehicle controller area network bus." IEEE Access 9 (2021): 99595-99605.

[16] Ben Lakhal, Nadhir Mansour, Othman Nasri, Lounis Adouane, and Jaleleddine Ben Hadj Slama. "Controller area network reliability: overview of design challenges and safety related perspectives of future transportation systems." IET Intelligent Transport Systems 14, no. 13 (2020): 1727-1739.

[17] Zeng, Weiying, Mohammed AS Khalid, and Sazzadur Chowdhury. "In-vehicle networks outlook: Achievements and challenges." IEEE Communications Surveys & Tutorials 18.3 (2016): 1552-1571.

[18] B. Cheon and J. Jeon, “The CAN FD network performance analysis using the CANoe,” in Proc. IEEE Int. Symp. Robot., Seoul, South Korea, 2013,pp. 1–5.

[19] T. H. Nguyen, B. M. Cheon, and J. W. Jeon, “CAN FD performance analysis for ECU re-programming using the CANoe,” in Proc. 18th IEEE Int. Symp. Consum. Electron., JeJu Island, South Korea, 2014, pp. 1–4.

[20] Tenruh, Mahmut, Panagiotis Oikonomidis, Periklis Charchalakis, and Elias Stipidis. "Modelling, simulation, and performance analysis of a CAN FD system with SAE benchmark based message set." Proc. 15th Int. CAN Conf. 2015.

[21] Vemparala, Manoj Rohit, Shikhara Yerabati, and Gerardine Immaculate 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 (RTEICT). IEEE, 2016.

[22] Zago, Guilherme Marcon, and Edison Pignaton de Freitas. "A quantitative performance study on CAN and CAN FD vehicular networks." IEEE Transactions on Industrial Electronics 65.5 (2017): 4413-4422.

[23] Kim, Haeri, Wonsuk Yoo, Seoncheol Ha, and Jong-Moon Chung. "In-Vehicle Network Average Response Time Analysis for CAN-FD and Automotive Ethernet." IEEE Transactions on Vehicular Technology (2023).

[24] 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 1.2 (2011): 40-45.

[25] Yong, Shujun, et al. "Analysis of the Influence of CAN Bus Structure on Communication Performance." IoT as a Service: 5th EAI International Conference, IoTaaS 2019, Xi’an, China, November 16-17, 2019, Proceedings 5. Springer International Publishing, 2020.

[26] Hegde, Rajeshwari, Siddarth Kumar, and K. S. Gurumurthy. "The impact of network topologies on the performance of the in-vehicle network." International Journal of Computer Theory and Engineering 5.3 (2013): 405.

[27] Hafeez, Azeem, Hafiz Malik, Omid Avatefipour, Prudhvi Raj Rongali, and Shan Zehra. Comparative study of can-bus and flexray protocols for in-vehicle communication. No. 2017-01-0017. SAE Technical Paper, 2017.

[28] Ishak, Mohamad Khairi, Omer Ali, Emma Ahmad Sirajuduin, and Lee Shea Qi. "Vehicle Sensors Programming Based On Controller Area Network (CAN) Bus Using Canoe." 2019 16th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON). IEEE, 2019.

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

[30] Shahian Jahromi, B.; Tulabandhula, T.; Cetin, S. Real-Time Hybrid Multi-Sensor Fusion Framework for Perception in Autonomous Vehicles. Sensors 2019, 19, 4357

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

Available online: https://www.rand.org/news/press/2016/04/12.htm (accessed on 1 June 2020).

[32] Kalra, N.; Paddock, S.M. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? Transp. Res. Part A Policy Pract. 2016, 94, 182–193. [CrossRef]

[33] Yao, Linlin, Jian Wu, Yu Wang, and Chuanfu 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). IEEE, 2014.

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

[35] Ignatious, Henry Alexander, and Manzoor Khan. "An overview of sensors in Autonomous Vehicles." Procedia Computer Science 198 (2022): 736-741.

[36] Yeong, De Jong, Gustavo Velasco-Hernandez, John Barry, and Joseph Walsh. "Sensor and sensor fusion technology in autonomous vehicles: A review." Sensors 21, no. 6 (2021): 2140.

[37] 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, Apr. 2014.

[38] Tindell, Ken, and Alan Burns. "Guaranteeing message latencies on control area network (CAN)." Proceedings of the 1st International CAN Conference. Citeseer, 1994.

[39] I Ali, Qutaiba. "Tele-operated Vehicles System Using WLAN and Industrial Ethernet Techniques." Al-Rafidain Engineering Journal (AREJ) 16.5 (2008): 33-42.