A Survey on Embedded Systems in Autonomous Vehicles

Isabella Paperno
Electrical & Computer Engineering Department
University of Nevada, Las Vegas
Las Vegas, United States
paperi1@unlv.nevada.edu

Daniela Nikoloska
Electrical & Computer Engineering Department
University of Nevada, Las Vegas
Las Vegas, United States
nikolosk@unlv.nevada.edu

***Abstract*—Autonomous vehicles (AVs) represent a fundamental change in transportation, enabled by the integration of embedded systems. This comprehensive survey delves into the embedded hardware and software architectures that are the groundwork for self-driving capabilities. It further researches the computational substrates, from high-performance heterogeneous system-on-chips (SoCs) to parallelized graphics processing unit (GPU) architectures. Critical design considerations such as functional safety, cyber-security and performance constraints are examined. The purpose of this survey is to review the emerging utilization of embedded systems in autonomous vehicles by observing their history, comparing the three main contenders today, then looking into specific case studies such as Alphabet Waymo, Tesla Autopilot, and a few others to provide practical perspectives on how different companies implement embedded systems into their autonomous driving platforms. The final portion of this survey will predict the future direction of autonomous vehicles and discuss how current advancements such as artificial intelligence (AI) or vehicle-to-everything (V2X) communication would impact the embedded systems.**

***Index Terms*—Artificial intelligence, autonomous vehicles, autonomy, centralized architecture, cyber-physical systems, cybersecurity, deep learning, distributed architecture, electronic control unit, embedded systems, hybrid architecture, machine learning**

I. INTRODUCTION

Over the past few decades, embedded systems have become integral components in realizing autonomous vehicle functionality. Moore's Law has enabled increases in transistor densities and semiconductor capabilities leading to the current era. Autonomous vehicles, often seen as the future of transportation, depend heavily on embedded systems to attain the advanced features required for safe and efficient self-driving. Safety features include real-time monitoring of components like sensors and actuators to detect anomalies and initiate corrective measures. Reference [1] emphasizes ensuring the safety of autonomous vehicles is a paramount concern, and embedded systems are integral to implementing the necessary safety measures. In addition to safety measures, embedded systems contribute to navigation, maintenance, energy management, human-machine interaction (HMI), and environmental awareness capabilities within autonomous vehicles. The main scope and objectives of this survey are:

1) To analyze the embedded computing architectures enabling autonomous vehicle operations.
2) Review case studies and observe their successes and failures.
3) Look at the current developments and predict the future direction of autonomous vehicles.

II. BACKGROUND

A. History

The history of embedded systems in self-driving cars stretches back over 30 years. The first car to

incorporate an embedded system was the 1977 General Motors Oldsmobile Toronado by using an electronic control unit (ECU) for electronic spark timing [2]. The idea of using specialized hardware to accelerate neural network computations in embedded systems can be traced back to Intel's electrically trainable artificial neural network (ETANN) in the late 1980s. This analog circuit implementation, with its 64 neurons and 10,240 synapses, showcased the potential benefits of dedicated accelerators for improving efficiency in embedded neural network applications [3]. However, it was not until the early 2000s that significant breakthroughs were made toward autonomous driving capabilities.

The Defense Advanced Research Projects Agency (DARPA) Grand Challenges in 2004, 2005, and 2007 provided major advances [4]. The challenges attracted teams from top universities and research institutions worldwide. The 2007 DARPA Urban Challenge marked the most ambitious event yet in the Defense Agency's trilogy of Grand Challenges. While the preceding 2004 and 2005 Grand Challenges involved navigating self-driving cars through static desert courses, the Urban Challenge raised the difficulty level. Self-driving cars now had to demonstrate situational awareness by navigating dynamic urban settings with moving traffic and obstacles. As further discussed in [5] this new challenge required a more complex system referred to as cyber-physicals systems (CPS). The cyber-physical systems approach has been pivotal for propelling embedded autonomous driving systems forward. CPS techniques like co-design, distributed real-time architectures, formal modeling, and human-system integration ensured embedded autonomous driving systems operated safely.

Waymo (formerly the Google Self-Driving Car project), launched in 2009, made a significant development by creating a vertically integrated embedded computing platform purposely built for autonomous vehicles [4]. Waymo's real world testing across millions of road miles generated critical data for refinement and validation.

Throughout the 2010s, numerous automakers and technology companies formed partnerships to advance autonomous driving capabilities. Notable examples include the partnership between BMW, Intel, and Mobileye. This cross-industry approach combined the automotive expertise of traditional manufacturers with the computing and artificial intelligence (AI) knowledge of tech companies. Over the course of the decade, the cross-industry partnerships produced breakthroughs shaping the embedded computing foundations for production self-driving cars.

B. Fundamentals of Autonomous Vehicle Systems

When we discuss autonomous vehicles, there are different levels of autonomy that we could refer to. The Society of Automotive Engineers (SAE) has defined six levels of automation to describe how much human-to-machine dependence the vehicle requires, which has become adopted as the "de facto global standard," the J3016 automated driving classification standard [6]. Each level, described below, utilizes embedded systems that grow in complexity alongside each defined level of autonomy to accommodate for the growing responsibilities being handled by the machines. Beginning with level 0, "No Driving Automation," this describes standard vehicles where the driver has full control of all driving tasks and no automation is used whatsoever. Next is level 1, "Driver Assistance," where the vehicle contains a single automated feature such as cruise control or a collision warning system but the driver remains in control of the majority of components. Then level 2, "Partial Driving Automation," describes a vehicle that can perform steering and acceleration on its own but is still dependent on the driver being alert and ready to take over at any time, especially in emergency situations. Moving into the more autonomous levels, level 3, "Conditional Driving Automation," has the vehicle controlling most driving tasks but the driver still needs to be alert as an emergency response and correct any erroneous maneuvers. Level 4, "High Driving Automation," states that the vehicle can perform all driving tasks, under certain conditions. The driver should be in control under certain conditions such as heavy storms, merging into extremely congested traffic on a highway, or closed-campus operations, otherwise the vehicle is able to traverse by itself with little issues, even in emergency situations. Finally, there is level 5, "Full Driving Automation," where the driver can sit back as the vehicle has full control of all driving tasks under all conditions [6]-[8].

An autonomous vehicle contains five basic

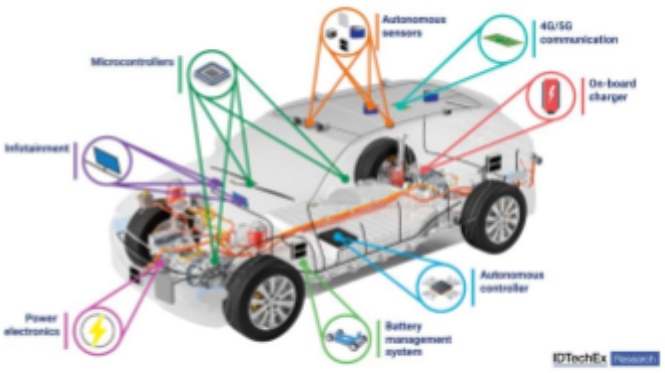

Fig. 1. Different semiconductor devices that could be found on autonomous vehicles [10].

functions that allow the vehicle to maneuver within the differing levels of automation. Those five functions are: perception, localization, planning, control, and system management. Perception is the ability for the vehicle to see its environment which can be achieved through the use of various sensors and sensing techniques such as light detecting and ranging (LiDAR) and computer vision. Localization functions allow the vehicle to accurately determine its position, usually with the techniques of a Global Positioning System (GPS), triangulation, or similar tracking methods. The planning functions act as the brain of the vehicle, analyzing the information provided from all the other functions and determining the appropriate motion to take or multiple possibilities to pursue. On the other hand, the control functions are the muscles of the vehicle which provide movement such as steering, accelerating, and braking. These movements are based on the results from the planning functions and the control functions follow every command given to them. Finally, the system management, these functions supervise everything that happens within the autonomous vehicle with key functions such as logging, fault management systems, and human-machine interface for interactions. The system management would have all the information if the vehicle acted up and could assist in pinpointing the errors [10].

Within each of those five functions are many different sub functions that can differ based on the complexity and needs of the autonomous vehicle. The sub functions could include RADAR, infrared sensor, or any other autonomous sensor for perception, autonomous controllers for control, 4G/5G communication for planning functions to communicate with necessary components, screens or some other infotainment devices for human-machine interaction within the system management and many more (Fig. 1) [10].

Some sub functions could consist of just one component but most times there is a fusion of two or more components working together. For instance, perception could be handled by just a single LiDAR or camera, each with their own advantages and disadvantages. Using a single camera can allow for clearer images with more data to read but lacks depth perception and loses value in the dark as it cannot see as much or as far. Meanwhile a single LiDAR provides a point cloud map of anything that the laser can hit which could provide a further view of the surrounding area even in the dark but is missing key details such as road markings or the colors of a stop light. Both can also fail under bad weather conditions, the LiDAR more than cameras as the weather can greatly impact a LiDAR's reading but it only distorts a camera's view if it is large or heavy enough to obstruct it. By fusing these two visions together the vehicle can obtain a complete picture of its surroundings with a large radius filled with details (Fig. 2) [11].

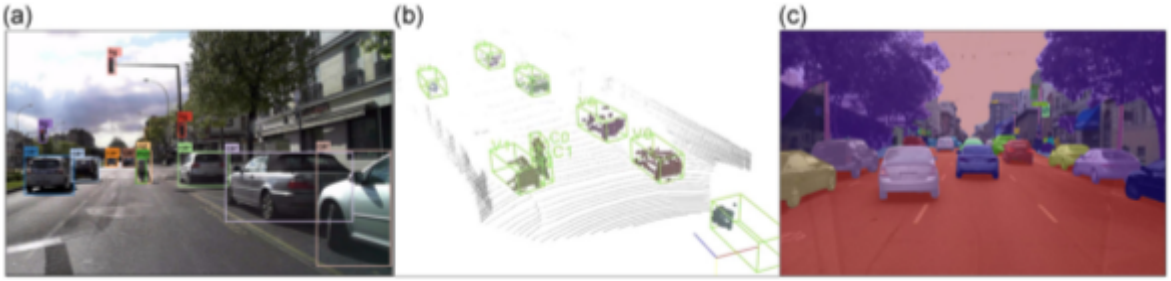

Fig. 2. Scene perception result samples. (a) 2D object detection in images captured from camera systems, (b) 3D bounding-box detector applied on LiDAR data, and (c) semantic segmentation results on images [11].

III. Embedded System Architecture

Throughout all these references, there is a growing reliance on the software used to make cars autonomous while also increasing the complexity of the software. The increased complexity comes from creating new code to solve the many issues that arise. In 2009, low-end cars had 30 to 50 ECUs all throughout the car to be able to control everything within it and monitor it for safety [2]. Now, there are even more systems such as the Advanced Driver Assistance Systems (ADAS), intelligent driving model (IDM), Trusted Platform Modules (TPM), and Root of Trust (RoT) that have been developed to help secure the safety of these vehicles and ensure that they would be able to effectively drive without needing the assistance of a person.

The growing reliance on software requires an efficient architecture to manage the complexity and ensure reliable operations. There are three main architecture types—centralized, distributed, and hybrid—each with their own unique benefits and trade-offs.

A. Centralized System Architecture

In a centralized system architecture, a majority of the components are contained within a single, centralized, computational unit. The vision and movement are external components that connect to the computational unit containing everything else through multiple wires (Fig. 3); this single unit allows for an ease of shareability as no additional networks need to be added, allowing for minimum delay and loss of information [9].

However, centralized systems lack scalability as the number of components would be limited to the size of the computational unit and increasing its size could also increase the length of connections between the sensors or actuators to the unit, adding more weight and noise to the overall system [9]. Not only that but collecting all the computational aspects into a centralized unit forces all the functions in one place, preventing tests or developments from any submodules from occurring at the same time. It forces the modules to perform within a certain sequence which requires high computing power and makes it difficult to detect any faults before they occur or even maintain a backup module for those purposes [12].

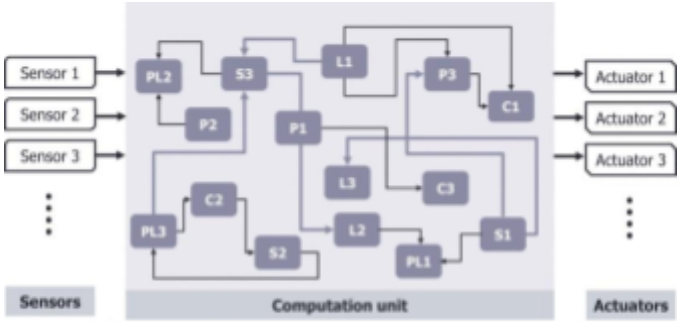

Fig. 3. Centralized architecture for autonomous cars [11].

B. Distributed System Architecture

In distributed system architectures, there could be multiple local computation units with all the sensors and actuators distributed between the units and each unit is connected through a common communication system (Fig. 4). By breaking into multiple computational units, the overall complexity of the autonomous system is reduced with the individual units being as complex as their objective needs it to be. It also increases computational performance of the system as the units can run different processes in parallel [9].

This concept is further explored in [13] where an open platform is introduced for autonomous vehicles using distributed systems. Their basic control and data flow diagram (Fig. 5) illustrates how the system can distribute computational tasks across different modules allowing data to be processed from multiple inputs in parallel with all communication between these units relying on a Controller Area Network (CAN) bus [13].

Essentially, the distributed system architecture resolves any issues from the centralized system architecture, however it brings some issues of its own. Due to having multiple control units, the distributed system requires a central communication system to distribute data to the necessary locations,

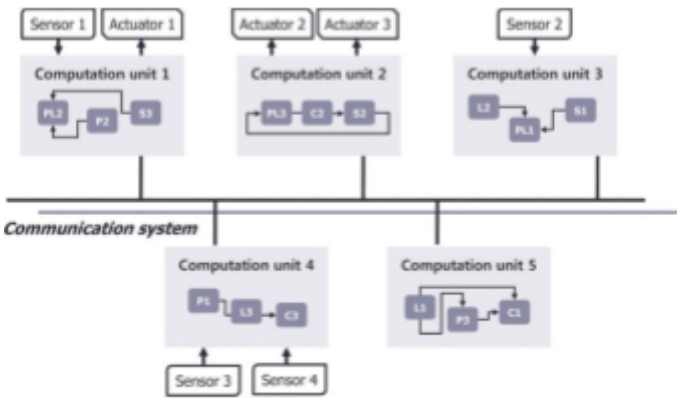

Fig. 4. Distributed architecture for autonomous cars [11].

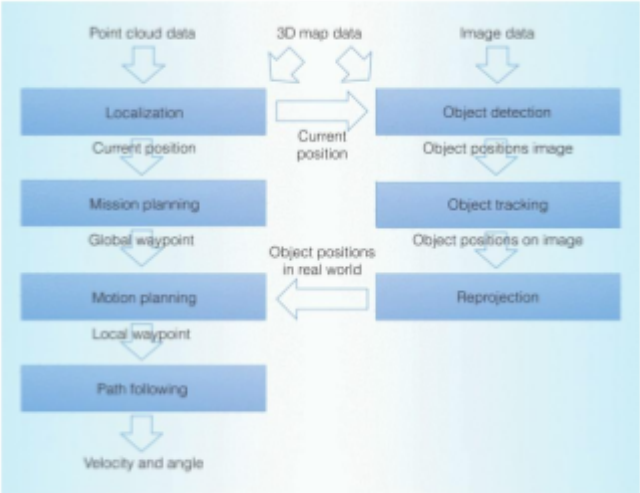

Fig. 5. Basic control and dataflow of algorithms [13].

but that causes a loss of performance as some data may be delayed or collect noise between functions. This is a major issue in time sensitive cases that could occur when traversing at high speeds in an autonomous vehicle. In an attempt to resolve this, time-triggered communication protocols such as FlexRay or the Automotive Data and Time-triggered Framework (ADTF) were introduced to be activated for specific applications [12].

C. Hybrid System Architecture

Hybrid system architectures combine the best parts of the centralized and distributed models discussed above but focus more on optimizing performance and flexibility.

One notable example is the open-source software Autoware. Autoware is a hybrid system due to its main application being a centralized unit for itself but working with other units to allow the entire vehicle to function as seen in Fig. 6 [14]. A vehicle utilizing Autoware could still have multiple processes running simultaneously to figure out the localization through different methods such as creating 3D maps from LiDAR scans while also using mapping algorithms like the Iterative Closest Point (ICP) algorithm, and detecting surrounding objects all within their own computational units just like the distributed system architecture. However, once all that data is obtained, it gets fed to the Autoware software which then processes all the gathered information, comparing the accuracy of the mapping algorithms with the LiDAR maps and checks if the objects detected were captured on some other visual sensor, in order to produce the most accurate response in their current situation. The way it collects all the data to produce the final outcome is similar to the centralized system architecture [14].

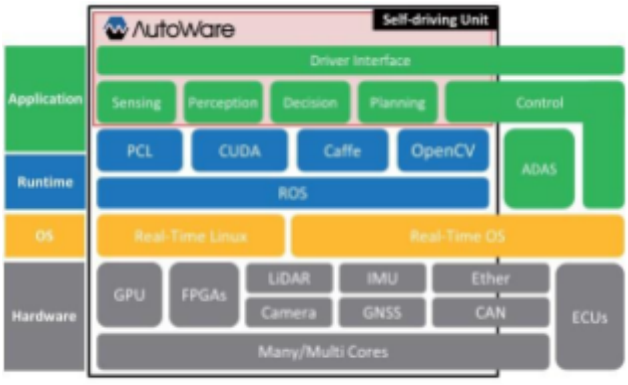

Fig. 6. Complete software stack for autonomous vehicles using Autoware [14].

IV. Specific Case Studies

In the rapidly evolving autonomous vehicle industry both automobile manufacturers and newly created companies are all racing to create a level 5 autonomous vehicle. However, the "self-driving cars" of today are only reaching level 2 autonomy with the leaders achieving level 3 autonomy [8]. As of 2018, companies like Waymo, General Motors (GM), and Volkswagen led the development of autonomous vehicles with many others following in their footsteps (Fig. 7). This portion of the survey will be looking at some of these companies to observe certain advancements in the technological developments for the autonomous vehicles industry.

Fig. 7. Navigant Research Leaderboard Report for Automated Driving in 2018 [8].

A. *Alphabet's Waymo*

Waymo's embedded autonomy stack demonstrates a sensor fusion approach combining multiple data streams through deep learning (DL) architectures. The core perception model is regarded in [15] as RoadNets, a two stage three-dimensional (3D) convolutional neural network processing raw inputs from LiDAR (light detection and ranging, cameras, and radars). In the initial stage, 3D region proposals are generated from different sensor modalities [15]. For example, a LiDAR identifies 3D regions of interest from the point cloud data, while image-based fusion heads propose regions from the camera inputs. The second state is where visual segmentation "RoadMask" subnetworks classify and refine the regions. Enabling precise detection and tracking of obstacles such as pedestrians, cyclists, and vehicles.

Comparators are fundamental components in Waymo's embedded systems. The study by Rosenband explains how these comparators are built from a few dozen transistors and are replicated thousands of times across the self-driving system. For instance, Waymo's 1080p image sensors contain 1920 comparators, one for each pixel column, and must perform over 510 billion comparisons every second [16]. The custom LiDARs described previously rely on Waymo comparators to precisely measure the time it takes for a laser pulse to reflect off an object and return to the sensor, enabling distance measurements accurate to within 1 cm. Waymo's custom LiDAR system and extensive use of comparators in their embedded systems have been crucial in advancing autonomous vehicle technology.

B. *Tesla Autopilot*

Tesla's Autopilot system is powered by a custom embedded hardware platform, the Full Self-Driving (FSD) Computer. The latest generation of the FSD Computer, introduced in 2019, features a custom SoC. As stated by Talpes et. al, the latest FSD computer is a 21x improvement over its predecessor and is capable of processing up to 2300 frames per second [17]. The SoC includes two neural processing networks, each capable of delivering 144 trillion operations per second (TOPS). This custom chip accelerates the neural networks used in Autopilot and Full Self-Driving while reducing cost and power consumption. It further exemplifies how purpose-built hardware can greatly enhance the performance of autonomous driving systems.

Tesla's approach to Autopilot development is also characterized by a "tethered" architecture.

TABLE I
TESLA'S AUTOPILOT HARDWARE GENERATIONS

	Period 0	Period 1	Period 2		Period 3
Hardware Generation	No-AP	First (AP1)	Second (AP2)	Second (AP2.5)	Third (AP3)
Production Period	Jun 2012 to Sep 2014	Sep 2014 to Oct 2016	Oct 2016 to Aug 2017	Aug 2017 to Mar 2019	Mar 2019 to Dec 2020
Car Models	S	S, X	S, X	S, X, 3	S, X, 3, Y
Main Autopilot Computer	N/A	Mobileye EyeQ3, capacity 0.256 TOPS	Nvidia Drive PX2, capacity ~10 TOPS	Nvidia Drive PX2, capacity ~10 TOPS, with added redundancy	Tesla FSD, capacity ~144 TOPS
Camera	N/A	1 forward facing, monochrome, maximum distance not known	3 forward facing, max. distances of 250, 150, and 60 meters, depending on the field of vision 2 sides forward facing max. distance 80 m. 2 sides rearward facing max. distance 100 m. 1 rearview camera, max. distance 50 m. *(AP2.5 changed the color filters for forward and side cameras from RCCC to RCCB)*		
Radar	N/A	1 forward facing, max. distance 160 meters		1 forward facing, max. distance 170 m.	
Ultrasound	N/A	12 around the car, max. distance 5 m.	12 around the car, max. distance 8 m.		

Table from [18] that was created from the following sources: Tesla, Mobileye, Nvidia, Continental, and reverse engineering reports.

According to [18], the architecture enables continuous improvement of Autopilot's behavior through software updates and data collection [18]. Between 2012 and 2020, Tesla introduced four car models, three Autopilot hardware generations and 47 software based features, releasing software updates every five days on average (Table I). This tethered architecture allows for modular hardware variations and enables a new trajectory for software based development. Examining this "tethered" architecture we observe how Tesla challenged the traditional notion of fixed hardware-software configurations in embedded systems. Tesla has shown the potential for embedded systems to evolve over time based on real-world data and user feedback.

C. Baidu Apollo

Baidu, a Chinese technology company, has developed a real-time operating system called the Apollo OS for autonomous driving. In the examination of the Baidu Apollo platform, [19], emphasizes how the development of a specialized operating system differentiates Apollo from other platforms [19]. Unlike general-purpose operating systems, Apollo OS is designed to meet the real-time and safety-critical requirements of self-driving systems. One of the key components of the Apollo platform is the Apollo Computing Unit (ACU). Another component is Apollo Dreamland, a self-driving simulator. Dreamland provides a realistic and immersive simulation environment that accurately replicates real-world driving scenarios. Building upon this, [19], further explains that the simulator saves developers time and effort by providing a virtual testing environment [19]. Dreamland's integration with Apollo OS ensures that the simulated behavior of the autonomous vehicle closely matches its real-world performance.

Reference [20], utilized the Apollo Open Platform and Apollo Dreamland similarly to develop and test a novel path planning approach for autonomous vehicles. Their strategy involved a two stage process: first, generating a smooth driving guideline from map data, and second, optimizing the path in the Frenet frame using a piecewise-jerk formulation [20]. The Frenet frame, also known as the Frenet-Serret frame, is a coordinate system that is used in path planning for autonomous vehicles. By leveraging the Apollo Open Platform and Apollo Dreamland simulator, [20] demonstrated significant improvements in path planning. The two stage optimization approach resulted in efficient computation of kinematically possible paths in cluttered environments.

D. NVIDIA DRIVE Platform

NVIDIA's primary hardware for AVs is updated every 3 years. At the hardware level, NVIDIA's DRIVE series has set new benchmarks for computational power in the industry. The latest DRIVE AGX ORIN Developer Kit offers 254 TOPS of performance to support multiple AI inference pipelines. Reference [21] provides additional context by explaining that the latest version was integrated into production vehicles in 2023, while its successor, DRIVE Atlan, is scheduled for release in 2026 [21].

Looking ahead, NVIDIA's DRIVE Hyperion 9 is a platform for software-defined autonomous vehicles, scheduled for 2027. Hyperion 9 aims to achieve automated driving with 8x the performance of the current DRIVE Orin-based architecture within the same power envelope. Additionally, Hicok offers a more detailed explanation, stating that the platform incorporates redundancy in its compute architecture, with the DRIVE Thor superchip delivering 2,000 teraflops of performance [22]. On the software side, NVIDIA's DRIVE OS and DriveWorks SDK have streamlined the development of AVs applications. DRIVE OS is an operating system tailored for autonomous vehicle applications running on DRIVE AGX-based hardware. According to NVIDIA, it offers security features like secure boots, security services, and over-the-air updates [23]. DriveWorks SDK, on the other hand, provides a library of modules, tools, and reference applications that enable developers to make full use of the DRIVE platform's computing power. NVIDIA has consistently raised the bar for performance and scalability in embedded systems for autonomous vehicles. From achieving unprecedented levels of computational power with platforms like DRIVE AGX Orin and the upcoming DRIVE Hyperion 9 to software innovations such as the DRIVE OS and DriveWorks SDK.

E. GM Cruise

In 2016, General Motors acquired Cruise Automation and used their knowledge to develop

one of the first level 4 autonomous vehicles utilizing a custom embedded system called the Cruise Self Driving Computer (CSDC). The CSDC features a modular architecture that allows for scalability and flexibility. One innovation in Cruise's embedded system is the use of a real-time operating system (RTOS) called the Cruise AV OS. The use of this RTOS allows for tight coordination between different components of the autonomous driving system.

Unlike the previous case studies, there is little information regarding the embedded system, instead the company website provides videos describing how artificial intelligence has been integrated with their system for better control. Still utilizing the RTOS but combining it with deep learning algorithms, Cruise AVs are able to track sudden kinematic changes and accurate behavioral predictions through miniscule details, allowing the autonomous vehicle to determine the difference between a right turn and a wide U-turn where other systems would fail [24].

V. Challenges

There are many things to consider when creating autonomous vehicles. Not only do we need to consider how the components work together in the autonomous vehicle but we also need to view the autonomous vehicle as a whole and observe its impact on society. With this perspective, safety and security are crucial objectives to perfect in these vehicles, more so than the current automobiles being driven.

A. Safety

The main safety concern stems from faults within the automated system. A fault is at least one deviation from the system's usual condition and can be classified as sensor (input) faults, actuator (output) faults, and component or process faults which are faults from anywhere else in the system [25]. Sensor faults can be caused from various issues such as hardware failures, environmental factors like weather conditions, or adversarial attacks such as spoofing or jamming. Reference [26] further explains that jamming attacks disrupt wireless communication channels by sending jamming signals to block V2X network communication. Spoofing attacks, on the other hand, replace actual sensor data in existing communication protocols, such as misleading GPS receivers with stronger spoofed signals. Actuator faults may result from mechanical breakdowns, software bugs, or malicious interference like ECU software flashing attacks aimed at gaining control over the vehicle. These faults can potentially result in unintended vehicle movements or complete loss of control. Incorporating fault tolerance through redundant hardware and software architectures is essential in the presence of faults. Sensor fusion techniques help overcome individual sensor limitations and improve environmental perception capabilities.

International standards, such as ISO 26262, are fundamental in outlining the safety requirements for road vehicles, encompassing the entire lifecycle from development to decommissioning. This standard assigns Automotive Safety Integrity Levels (ASIL) to hazards, ranging from ASIL D (highest hazard) to ASIL A (lowest hazard). The ASIL class is determined by factors such as severity, exposure, and controllability of the hazard. Verification processes can help identify potential safety hazards before deployment. This includes simulation-based testing and real-world trials.

B. Cybersecurity

A significant cybersecurity challenge for autonomous vehicles arises from their reliance on open communication systems like V2X (Vehicle-to-Everything) networks. Reference [26], propose a systematic classification of security threats derived from the principles of confidentiality, integrity, and availability (CIA). This includes false data injection attacks that feed erroneous sensor data, information theft targeting vehicle and user data, privilege escalation attacks to gain unauthorized control, communication blocking through denial-of-service attacks, and time delay attacks that disrupt real-time performance. These cyberattacks The range of potential cyberattacks targeting the embedded systems of autonomous vehicles is vast, necessitating a comprehensive defensive approach. Efforts are underway to develop cybersecurity standards like ISO 21434 that present risk management requirements for the entire vehicle life cycle. As well as IEEE 1609.2 that defines secure data structures, formats, and processing for vehicle security [26].

VI. Future Directions

Many of the papers we reviewed explain the use of embedded systems in safety protocols or monitoring systems and provide simulations or comparisons of the software through multiple embedded systems. Many articles also detail the importance of sensors and how they can be used to gather information about the environment around the vehicle that the embedded systems and code can then react to. However, even with newer sensors or other components, the main development involves newer technologies such as artificial intelligence (with all its levels), newer simulation methods, and enhanced algorithms for tasks such as model prediction or fault detection, all aimed to improve the safety of the autonomous vehicles.

A. Integration of Artificial Intelligence

The emergence of artificial intelligence has greatly impacted the way things work today. There are many methods that could be improved by using AI software or its many branches such as machine learning (ML), deep learning (DL), reinforcement learning (RL) and many more. Through these, algorithms could be created to have the machines test and learn on their own which provides better opportunities for safety testing as tests could occur in quicker successions and the algorithms would be learning from each test instead of having to wait for humans to feed the knowledge to improve the system.

Reference [11] describes how newer technologies such as deep learning, a form of artificial intelligence, utilize a 3D representation of the vehicle's environment from the use of LiDAR sensors which allow the vehicle to obtain a 360° horizontal field of view to react to. They also detail how current deep learning methodologies can be implemented in two ways, either as a modular perception-planning-action pipeline (Fig. 8a) where there are four main components working together to control the vehicle or as an End2End learning fashion (Fig. 8b) where information regarding the environment collected from the sensors is mapped directly to the outputs, causing a more sudden reaction to any changes [11]. Both options would be able to utilize a new system called "learning controllers" which, just like traditional controllers, make use of prior models with fixed parameters to control the movement of the vehicle but also learn from and train based off their current models, providing more accurate responses and controls each test. This is achievable with the deep learning technology that allows machines to learn through repetition and fine-tune their algorithms on the fly. However, there are some risks with this, for instance, there could be a fault during one of the

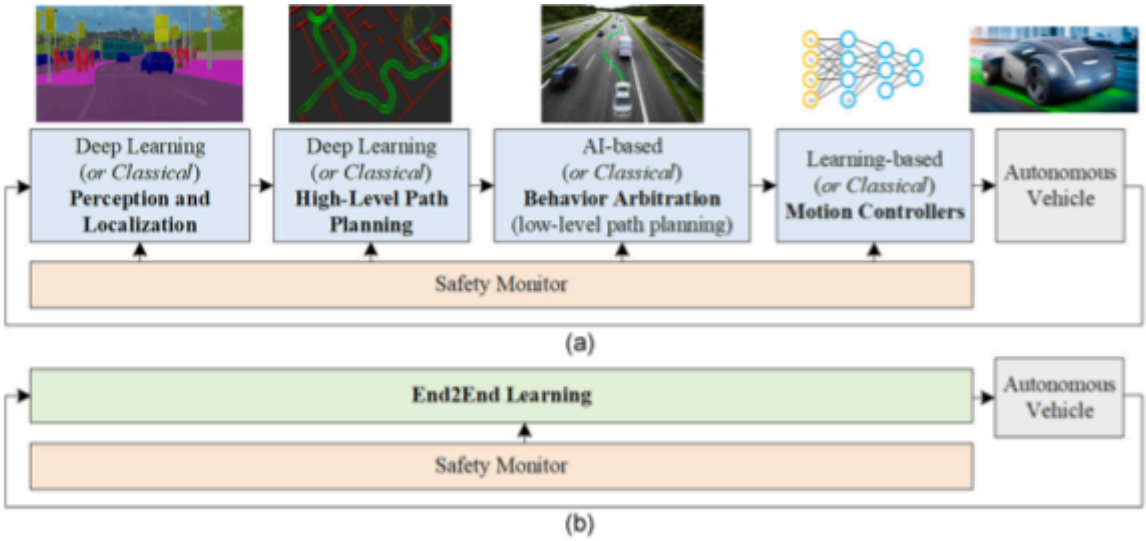

Fig. 8. Utilizing deep learning in autonomous cars. The architecture can be implemented either as a sequential perception-planning-action pipeline (a) or as an End2End system (b). In the sequential pipeline case, the components can be designed either using AI and deep learning methodologies, or based on classical nonlearning approaches. End2End learning systems are mainly based on deep learning methods. A safety monitor is usually designed to ensure the safety of each module [11].

TABLE II
PERCENTAGE OF SUCCESS FOR GOAL-DIRECTED NAVIGATION EXPERIMENTS

	Training conditions			New town			New weather			New town & weather		
Task	MP	IL	RL	MP	IL	RL	MP	IL	RL	MP	IL	RL
Straight	**98**	95	89	92	**97**	74	**100**	98	86	50	**80**	68
One turn	82	**89**	34	**61**	59	12	**95**	90	16	**50**	48	20
Navigation	80	**86**	14	24	**40**	3	**94**	84	2	**47**	44	6
Nav. dynamic	77	**83**	7	24	**38**	2	**89**	82	2	**44**	42	4

Experiment results of three autonomous driving systems on goal-directed navigation tasks. The three methods evaluated are: modular pipeline (MP), imitation learning (IL), and reinforcement learning (RL) with the table reporting the percent of successfully completed drives in each condition [28].

tests that negatively affects the results, without someone catching that and fixing the issue the algorithm can continue to train itself to exploit those vulnerabilities leading to an unsafe performance [11].

B. Improved Simulations

Utilizing more simulation platforms could also assist in ensuring safety as more tests could be conducted without endangering anyone and the software could fine-tune the autonomous features.

Bouncing off the AI integration, companies have taken to using AI to enhance the amount of testing they can obtain without needing to go out in the real-world and possibly endanger those on the roads. Utilizing AI allows simulations to test a number of scenarios such as specific locations, various weather conditions, or reactions of motorists that may take "thousands of road miles to test it properly" in the real-world either due to rarity of the occurrence or costly conditions. These test scenarios also make use of a system called non-playable character (NPC) AI to simulate how pedestrians, other vehicles, and even emergency vehicles may react at any given time to properly test the autonomous features in a secure environment. This method of testing is very scalable as any interaction or experience can be generated with high accuracy rather than having to go create a specific scenario to test in the real-world [27].

One specific example is Car Learning to Act (CARLA), an open-source simulator for autonomous driving research. CARLA was created for the sole purpose of supporting development, training, and validation of autonomous driving systems [28] while providing flexible communication between the simulation environment and the experimenter. Being open-source, CARLA has access to and provides digital assets such as urban layouts, buildings, and vehicles that are designed specifically to simulate real-world conditions. In [27], the platform was used to evaluate three approaches to autonomous driving: a modular pipeline (MP), a deep network trained with imitation learning (IL), and a deep network trained with reinforcement learning (RL). To accurately evaluate these methods, forty-eight different simulations were tested utilizing four increasingly difficult driving maneuvers, two town environments (with Town 1 used for training and Town 2 used for testing), and six weather conditions. The results in Table II present the success rate of each condition (the higher the better) with the MP and IL methods greatly outperforming the RL method however, it is unclear which is better between MP and IL as both had their wins in different scenarios [28]. Overall, the simulations were able to provide difficult and lifelike scenarios for the autonomous vehicles to traverse which provided more insight on what improvements could be made for each system. Through more simulations like this we can confirm the safety virtually, ensuring the software will act and react properly, before taking the real vehicles onto the road. These simulations also provide an easier way for manufacturers to catch on if their system is exploiting any loopholes and adjust them accordingly.

C. Model Predictive Control Improvement

A similar development for the future would be the improved application of model predictive control (MPC), which is intended to assist drivers and

TABLE III
COMPARISON OF COMPUTATIONAL EFFICIENCY

Strategy in use	"Only MB"	"Only CSC"	"MB+CSC"
Comp. time	B/A	C/B	C/A
Mean C.T.	34.4%	30.7%	12.2%
Max. C.T.	52.95%	35.0%	18.8%

This table compares the computational efficiency of the move blocking (MB) and constraint-set-compression (CBC) strategies individually then together when controlling a vehicle in a constant acceleration scenario [29].

improve road safety. MPC has actually been used in vehicles already through the adaptive cruise control (ACC) model but due to its computational issues researchers wish to improve it. In order to reduce the computational intensity, researchers are combining the MPC with move blocking (MB) and constraint-set-compression (CSC) strategies while still allowing it to maintain real-time implementation. Through MB, the degrees of freedom on the control input would be reduced which makes the output a smaller value with a more accurate approximation. However, if the computational intensity is still high after that, the CBC strategy will be implemented to set constraints on the output significantly reducing the intensity as it has limited the data that the MPC needs to review. Two cases were tested to ensure this method worked, the first test was with a vehicle that was constantly accelerating and the second was a vehicle in a more natural traffic flow, with both tests three algorithms were tested: A - the baseline or basic MPC, B - only the MB strategy, and C - using both the MB and CSC strategies. Table III showing the overall computational performance showing that algorithm C (utilizing both the MB and CSC strategies) was the most effective in improving the computational speed as the maximum computational time dropped from 52.95% with only the MB strategy to 18.8% when utilizing both. Table IV compares the loss of optimality through the standard deviations while it does appear that utilizing only MB would have fewer computation errors than both MB and CSC, all errors are less than 0.1% which means that the strategies have negligible effect on the control optimality and would still be the best option out of the three algorithms [29].

D. Fault Detection Methods

To deal with a safety issue, a new fault detection method has been proposed called a hybrid fault detection and diagnosis (FDD) system which will

TABLE IV
COMPARISON OF LOSS OF OPTIMALITY

Algorithm	A	B "Only MB"	C "MB+CSC"
	Std. derivation	Std. deriv. of computation errors	Std. deriv. of computation errors
Δv (m/s)	1.09	3.4×10^{-4} (0.03%)	9.1×10^{-4} (0.08%)
Δd (m)	14.8	6.1×10^{-3} (0.04%)	7.2×10^{-3} (0.048%)

This table compares the effectiveness of the move blocking (MB) and constraint-set-compression (CBC) strategies of control in a naturalistic traffic flow scenario. It records the relative speed (v) and distance error (d) found from each of the three tested algorithms where algorithm A is the baseline, algorithm B is the MB strategy, and algorithm C combines the MB and CSC strategies.

Note: The bracket is the percentage of computation errors compared with that in algorithm A [27].

combine a standard autonomous vehicles fault detection system with techniques such as fuzzy logic and support vector machines (SVMs) to be able to diagnose the fault. In order to diagnose the fault though, three tasks must be completed: fault detection to check if there is any fault or malfunction in the system, fault isolation to locate the faulty component, and fault identification to determine the type of fault it is. One-Class SVM becomes the fault detector and through simulations proves that it can detect abrupt faults and slowly changed faults in real-time by creating boundary curves that classify safe and unsafe regions. Then a Kalman filter observer is used to predict the location of the fault with the Jarque-Bera test being used to confirm the location by checking for noise. Finally a neural network fuzzy system is used to classify the fault types and through its neural network nature it is able to learn and update its calculations to accurately classify the faults [25]. Experiments were conducted resulting in the One-Class SVM having a high detection rate and the Kalman filter observer having a high location accuracy, providing excellence in this new FDD system and a new method to ensure safety.

VII. CONCLUSION

This survey has provided an extensive overview of the role embedded systems play in the development of autonomous vehicles. By examining the historical context, fundamental concepts, and architectural choices, we have shown the complex interplay between hardware and software components that enable self-driving capabilities.

The case studies of industry leaders such as

Waymo, Tesla, Baidu, NVIDIA, and GM Cruise demonstrate the diverse approaches and innovations in embedded system design. Tesla's centralized FSD computer exemplifies the pursuit of reduced latency and simplified integration, while General Motors' CSDC showcases the advantages of a distributed architecture in terms of scalability and flexibility. Baidu's Apollo platform demonstrates the potential of hybrid architectures, combining the benefits of centralized and distributed approaches. As the case studies highlight, the choice of architecture depends on the specific requirements and priorities of each company.

The common misconception many people have is that the current autonomous vehicles are level 5, fully autonomous, when our survey shows that they are much closer to level 2 or 3. This survey has provided a comprehensive exploration of the different levels of autonomy and the roles of embedded systems in autonomous vehicles. We have reiterated the importance of embedded systems in enabling the core functionalities of autonomous vehicles such as perception, localization, planning, control, and system management.

However, the path to fully autonomous vehicles is not without challenges. Safety and cybersecurity remain the foremost concerns. As research and development in these areas continue to progress, we can expect to see significant advancements in the level of autonomy achieved by self-driving vehicles.

As we look towards the future of autonomous vehicles, it is evident that embedded systems will continue to shape the trajectory of this technology. The rapid advancements in computing power, sensor technologies, artificial intelligence algorithms, coupled with the increasing investments in research and development, paint a picture of a future where autonomous vehicles become an integral part of our daily lives.

References

[1] S. Sonko *et al.*, "The evolution of embedded systems in automotive industry: A global review," *World Journal of Advanced Research and Reviews*, vol. 21, no. 2, pp. 096–104, Feb. 2024. doi:10.30574/wjarr.2024.21.2.0420

[2] R. N. Charette, "This car runs on code," *IEEE Spectrum*, vol. 46, no. 3, pp. 3, Feb. 2009, doi: 10.1109/MSPEC.2009.4770101.

[3] M. Holler, S. Tam, H. Castro, and R. Benson, "An electrically trainable artificial neural network (ETANN) with 10240 'floating gate' synapses," *International Joint Conference on Neural Networks*, 1989. doi:10.1109/ijcnn.1989.118698

[4] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: *common practices and emerging technologies*," *IEEE Access*, vol. 8, pp. 58443–58469, 2020. doi:10.1109/access.2020.2983149

[5] C. Berger, "From Autonomous Vehicles to Safer Cars: Selected Challenges for Software Engineering," in F. Ortmeier and P. Daniel (eds.), Computer Safety, Reliability, and Security, SAFECOMP 2012, Lecture Notes in Computer Science, vol. 7613, Springer, Berlin, Heidelberg, 2012, pp. 207-220.

[6] D. Hopkins and T. Schwanen, "Talking about automated vehicles: WHAT DO levels of automation do?," *Technology in Society*, vol. 64, p. 101488, Feb. 2021. doi:10.1016/j.techsoc.2020.101488

[7] *Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles*, 2018. doi:10.4271/j3016_202104

[8] J. H. Hyun, "The Strategy of GM for the Development of Autonomous Driving Technology and Related Policies," *Journal of the Korea Academia-Industrial cooperation Society*, vol. 21, no. 3, pp. 51–56, Mar. 2020. doi:https://doi.org/10.5762/KAIS.2020.21.3.51

[9] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of autonomous car—part I: Distributed system architecture and development process," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 12, pp. 7131–7140, Dec. 2014. doi:10.1109/tie.2014.2321342

[10] N. Dahad, "Buying autonomous electric vehicles will be just like buying a laptop," Embedded.com,

https://www.embedded.com/buying-autonomous-electric-vehicles-will-soon-be-just-like-buying-a-laptop/ (accessed Apr. 30, 2024).

[11] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of Deep Learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, Nov. 2019. doi:10.1002/rob.21918

[12] O. S. Tas, F. Kuhnt, J. M. Zollner, and C. Stiller, "Functional system architectures towards fully automated driving," *2016 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2016. doi:10.1109/ivs.2016.7535402

[13] S. Kato *et al.*, "An open approach to Autonomous Vehicles," *IEEE Micro*, vol. 35, no. 6, pp. 60–68, Nov. 2015. doi:10.1109/mm.2015.133

[14] S. Kato *et al.*, "Autoware on board: Enabling Autonomous Vehicles with embedded systems," *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, Apr. 2018. doi:10.1109/iccps.2018.00035

[15] P. Sun et al., "Scalability in perception for autonomous driving: Waymo Open Dataset," arXiv.org, https://arxiv.org/abs/1912.04838 (accessed May 1, 2024).

[16] D. L. Rosenband, "Inside Waymo's self-driving car: My favorite transistors," *2017 Symposium on VLSI Circuits*, Jun. 2017. doi:10.23919/vlsic.2017.8008500

[17] E. Talpes *et al.*, "Compute Solution for Tesla's full self-driving computer," *IEEE Micro*, vol. 40, no. 2, pp. 25–35, Mar. 2020. doi:10.1109/mm.2020.2975764

[18] A. Lyyra, K. Koskinen, C. Sorensen, and T. Marion, "Tethered architectures in Cyber-Physical System Development: The case of Tesla's autopilot system," *SSRN Electronic Journal*, Jul. 2023. doi:10.2139/ssrn.4515061

[19] J. Xu *et al.*, "An automated learning-based procedure for large-scale vehicle dynamics modeling on Baidu Apollo Platform," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov. 2019. doi:10.1109/iros40897.2019.8968102

[20] Y. Zhang *et al.*, "Optimal vehicle path planning using quadratic optimization for Baidu Apollo Open Platform," *2020 IEEE Intelligent Vehicles Symposium (IV)*, Oct. 2020. doi:10.1109/iv47402.2020.9304787

[21] P. Schafhalter, S. Kalra, L. Xu, J. E. Gonzalez, and I. Stoica, "Leveraging cloud computing to make Autonomous Vehicles Safer," *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2023. doi:10.1109/iros55552.2023.10341821

[22] G. Hicok, "Introducing nvidia drive hyperion 9: Next-generation platform for software-defined autonomous vehicle fleets," NVIDIA Blog, https://blogs.nvidia.com/blog/drive-hyperion-9-thor/ (accessed May 4, 2024).

[23] "Nvidia Drive OS," NVIDIA Developer, https://developer.nvidia.com/drive/os (accessed May 4, 2024).

[24] C. Automation, "Autonomous Vehicle Technology: Driverless Cars," Cruise, https://www.getcruise.com/technology/ (accessed May 6, 2024).

[25] Y. Fang, H. Min, W. Wang, Z. Xu, and X. Zhao, "A fault detection and diagnosis system for autonomous vehicles based on hybrid approaches," *IEEE Sensors Journal*, vol. 20, no. 16, pp. 9359–9371, Aug. 2020. doi:10.1109/jsen.2020.2987841

[26] J. Han *et al.*, "Secure operations of connected and Autonomous Vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 11, pp. 4484–4497, Nov. 2023. doi:10.1109/tiv.2023.3304762

[27] R. Bellan, "Cruise lays out its plan for 'how' it will make robotaxis a reality," TechCrunch, https://techcrunch.com/2021/11/05/cruise-lays-out-its-plan-for-how-it-will-make-robotaxis-a-reality/ (accessed May 6, 2024).

[28] A. Dosovitskiy, G. Ros, F. Codevilla, A.

Lopez, and V. Koltun, “Carla: An open urban driving simulator,” PMLR, https://proceedings.mlr.press/v78/dosovitskiy17a.html (accessed Mar. 25, 2024).

[29] S. E. Li, Z. Jia, K. Li, and B. Cheng, “Fast online computation of a model predictive controller and its application to fuel economy–oriented adaptive cruise control,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1199–1209, Jun. 2015. doi:10.1109/tits.2014.2354052

[30] Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic robotics. MIT press.

[31] Korb, K. B., & Nicholson, A. E. (2010). Bayesian artificial intelligence. CRC press.

[32] D. Feng *et al.*, “Deep Multi-Modal Object Detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1341–1360, Mar. 2021. doi:10.1109/tits.2020.2972974

[33] C. Cadena *et al.*, “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age,” *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016. doi:10.1109/tro.2016.2624754

[34] H. G. Seif and X. Hu, “Autonomous driving in the icity—HD maps as a key challenge of the automotive industry,” *Engineering*, vol. 2, no. 2, pp. 159–162, Jun. 2016. doi:10.1016/j.eng.2016.02.010

[35] LaValle, S. M. (2006). Planning algorithms. Cambridge university press.

[36] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” arXiv.org, https://arxiv.org/abs/1105.1186 (accessed Mar. 25, 2024).

[37] B. R. Kiran *et al.*, “Deep Reinforcement Learning for Autonomous Driving: A Survey,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, Jun. 2022. doi:10.1109/tits.2021.3054625

[38] Coulter, R. C. (1992). Implementation of the pure pursuit path tracking algorithm (No. CMU-RI-TR-92-01). Carnegie-Mellon UNIV Pittsburgh PA Robotics INST.

[39] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, “A survey of motion planning and control techniques for self-driving urban vehicles,” *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, Mar. 2016. doi:10.1109/tiv.2016.2578706

[40] R. Rajamani, *Vehicle Dynamics and Control*. New York, NY: Springer, 2012.

[41] ISO. (2018). ISO 26262-1:2018 Road vehicles — Functional safety. International Organization for Standardization.

[42] G. Rong *et al.*, “LGSVL simulator: A high fidelity simulator for autonomous driving,” *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, Sep. 2020. doi:10.1109/itsc45102.2020.9294422

[43] L. Liu *et al.*, “Computing systems for autonomous driving: State of the art and Challenges,” *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6469–6486, Apr. 2021. doi:10.1109/jiot.2020.3043716

[44] E. Parisi et al., “TitanCFI: Toward enforcing control-flow integrity in the root-of-trust,” arXiv.org, https://arxiv.org/abs/2401.02567 (accessed Mar. 25, 2024).

[45] M. Duan *et al.*, *Research on vehicle lane keeping capability in low-speed scenarios based on autonomous driving technology*, Jan. 2024. doi:10.21203/rs.3.rs-3865516/v1

[46] A. Irshayyid, J. Chen, and G. Xiong, “A review on Reinforcement Learning-based Highway Autonomous Vehicle Control,” *Green Energy and Intelligent Transportation*, p. 100156, Jan. 2024. doi:10.1016/j.geits.2024.100156

[47] C. Conrad, S. Al-Rubaye, and A. Tsourdos, “Intelligent embedded systems platform for vehicular Cyber-Physical Systems,” *Electronics*, vol. 12, no. 13, p. 2908, Jul. 2023. doi:10.3390/electronics12132908

[48] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin, “Efficient iterative linear-quadratic approximations for nonlinear

multi-player general-sum differential games," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020. doi:10.1109/icra40945.2020.9197129

[49] H. Ning, R. Yin, A. Ullah, and F. Shi, "A survey on hybrid human-artificial intelligence for autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6011–6026, Jul. 2022. doi:10.1109/tits.2021.3074695

Daniela Nikoloska was born in Chicago, Illinois in 1999. She received her B.S. in Electrical Engineering from the University of Nevada, Las Vegas, in May 2023. Continuing her academic pursuits at the same institution, Daniela is currently pursuing a Master of Science in Electrical Engineering, with an expected graduation date in May 2025.

As a recipient of the NASA Space Grant Graduate Fellowship, Daniela conducts research using sonic-powered microgravity tissue chips to simulate space radiation effects on the human immune system. Previously, she served as an undergraduate research assistant at the University of Nevada, Las Vegas. In addition to her research endeavors, Daniela was an intern at VisionAid, where she designed the prototype of CaneIQ. She also actively engages in academic activities, serving as the Graduate & Professional Student Association (GPSA) Council Representative for the Electrical and Computer Engineering department at UNLV.

Daniela is an active member of professional organizations, including the IEEE Engineering in Medicine and Biology Society, Biomedical Engineering Society, and IEEE Institute of Electrical and Electronics Engineers.

Isabella Paperno was born in Las Vegas, NV, USA, in 2001. She received a B.S. in computer engineering from the University of Nevada, Las Vegas, USA, in 2024, and is pursuing a M.S. in electrical engineering at the same institution.

She is currently a Graduate Teaching Assistant in the Department of Engineering, University of Nevada, Las Vegas, NV, USA as well as a Co-Director of the Las Vegas Scholars Program in the same department. In 2022, she was an intern at Aldec, Henderson, NV where she reviewed the current tutorial system for their software and assisted in modernizing them. For the graduate thesis, she is researching methods of detecting arthritis in the hopes to create an algorithm that can catch the signs early and assist in prevention methods.