A Study on Operating Systems for Autonomous Vehicles and real-time systems

1 Tallapureddy Sai Kiran Reddy, 2Attili Divya Sree, 3Pagadala Risheeth Reddy,4Gurubinder Singh Brar

Computer Science, School of Computer Science and Engineering

Lovely Professional University, Phagwara, India

## Abstract: This paper dives into the world of autonomous vehicles (AVs) and the crucial role operating systems (OS) play in their development. Autonomous vehicles hold immense promise for revolutionizing transportation, offering a future with enhanced safety, efficiency, and accessibility. However, significant technical challenges remain, such as real-time data processing and ensuring system reliability. The paper explores how different OS options, including QNX, Linux-based AGL, AUTOSAR, ROS, and NVIDIA Drive OS, address these challenges. It delves into specific features like real-time performance, resource management, and communication, highlighting how they empower AVs to function effectively. Looking ahead, research focuses on advancements in sensor integration, harnessing the power of AI for complex decision-making, and mitigating cybersecurity threats. The paper concludes by emphasizing the transformative potential of OS-powered AVs. These intelligent vehicles have the power to reshape the future of mobility, creating a safer, more efficient transportation landscape.

## Keywords**:** Autonomous Vehicles, Operating Systems, Safety, Efficiency, Accessibility, Real-Time Performance, Resource Management, Communication, Sensor Integration.

I.INTRODUCTION

Autonomous vehicles are swiftly shifting from fiction to reality, thanks to advancements in operating systems. These systems serve as the backbone for integrating sensors, executing complex algorithms, and coordinating vehicle functions. They enable self-driving cars to perceive their surroundings and navigate safely without human intervention, marking a revolutionary shift in transportation.

Operating systems in autonomous vehicles must meet stringent standards for reliability, real-time performance, safety, and scalability. They process vast sensor data in real time, make split-second decisions, and execute complex algorithms for tasks like object detection and path planning. Ensuring passenger safety, they guard against cyber threats and system failures.

Prime choices for autonomous vehicles include real-time systems like QNX and Linux-based platforms such as Automotive Grade Linux (AGL). Each offers unique features, like deterministic performance and flexibility, shaping the landscape of autonomous driving. As this technology evolves, it holds promise for enhancing road safety, reducing congestion, and revolutionizing transportation across industries.

**Operating systems in autonomous vehicles are crucial for:**

1. **Resource Management**: They efficiently allocate CPU, memory, and peripherals, optimizing software performance.

2**. Task Scheduling**: Real-time OS prioritizes critical tasks like sensor data processing, ensuring timely execution.

3. **Communication**: They facilitate seamless data exchange between sensors, actuators, and software modules.

4. **Fault Tolerance**: OS implements error detection, recovery, and fault isolation to ensure system reliability.

5. **Security**: They enforce access controls, data protection, and encryption to safeguard against cyber threats.

**II.NEED OF THE STUDY:**

Research on autonomous vehicles using operating systems has the potential to create significant impact across multiple domains.

From a technological perspective, advancements in OS design and implementation can enhance the performance, efficiency, and

robustness of autonomous driving systems, ultimately leading to safer and more reliable transportation solutions. Firstly, AV technology has the potential to revolutionize transportation by enhancing safety, efficiency, and accessibility. However, numerous technical challenges need to be addressed to realize this potential fully. Operating systems serve as a critical component of AVs, providing the foundation for managing hardware resources, executing software applications, and facilitating communication between different subsystems. By investigating the intersection of AVs and operating systems, researchers can contribute to overcoming these challenges and advancing the state-of-the-art in autonomous driving technology. **Research Key areas of investigation may include:** What role do different **operating systems** play in the development and deployment of **autonomous vehicles**, and how can they impact the performance and reliability of these self-driving cars?

**III.RESEARCH METHODOLOGY**

**3.1. Levels of Autonomous Vehicle (AV’s)**

Autonomous cars are classified into five levels based on the degree of automation, as defined by the **Society of Automotive Engineers (SAE).** Here's a breakdown of each level:

Figure 1: Levels of Autonomous Vehicle

A diagram of a car level

Description automatically generated

1. **Level 1 Autonomous Cars**:

Level 1, the vehicle starts to receive support for steering assistance or acceleration/deceleration system functions, while the human driver still retains full authority over driving the vehicle.

2. **Level 2 Autonomous Cars**:

Level 2 represents a stage where autonomous vehicles provide support for steering and acceleration/deceleration functions, but humans remain responsible for judgment and driving-related decisions.

3. **Level 3 Autonomous Cars**:

At Level 3, autonomous vehicle systems take on the responsibility of driving, but the driver must be ready to intervene when requested, assuming control of the vehicle when necessary.

4.**Level 4 Autonomous Cars**:

In Level 4, the autonomous driving system can manage all aspects of driving, including core control functions, monitoring the driving environment, and responding to emergencies. However, driver intervention may still be required in certain situations.

5. **Level 5 Autonomous Cars**:

Finally, Level 5 represents full autonomy, where the autonomous vehicle system handles all aspects of judgment and control under any road conditions and environments, without the need for human intervention.

Table 1:Level of classification vs function

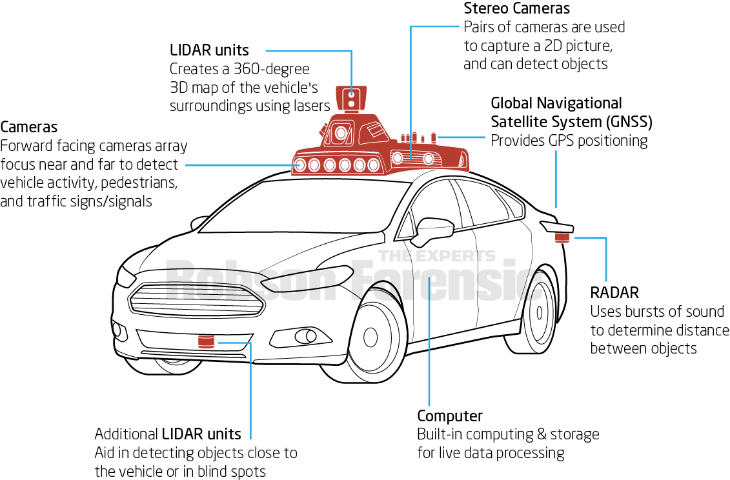
A table of text on a white background

Description automatically generated

**3.2. AUTONOMOUS VEHICLE WORK USING OPERATING SYSTEM**

A variety of sensors and computer systems are required for a vehicle to replicate the traditional human process of understanding its current location, desired location, and how to safely navigate the routes and hazards in between. Among the sensors currently available, no single unit is capable of deciphering its environment in a manner that provides sufficient information for autonomous driving. Instead, engineers design a system which incorporates a combination of sensors, strategically leveraging specific capabilities of one system, while accommodating its deficiencies through the use of others. In the area below we describe the most common sensors used in driver aids and autonomous systems:

Figure 2:AUTONOMOUS VEHICLE



1. **RADAR**- Uses bursts of sound to measure distance by measuring the time it takes for the sound to return to the sensor.
2. **LIDAR**- Similar to radar but uses lasers to measure distance instead of sound, with a range of up to 200m.
3. **Video Cameras** - Records multiple still shots (frames) to capture a 2 dimensional motion picture.
4. **Ultrasonic Range Sensors** - Uses high frequency sound waves to measure distance between objects.
5. **Inertial Measuring Unit (IMU)** - Combination of accelerometers and gyroscopes that determine a vehicle’s linear and angular motion.
6. **Global Navigation Satellite System (GNSS)** - Uses satellites to provide autonomous geo-spatial positioning. Global Positioning System (GPS) is a subset of GNSS.

An autonomous vehicle is a complex and diversified environment in which multiple operating systems can coexist. More than that, they can even run on the same hardware as one another. An ECU controls every component, from the engine to the windshield wipers, as well as every automotive function, from steering to brakes.

A small device in the body of a vehicle called an electronic control unit (ECU) is in charge of controlling a specific function.

Today’s vehicles may include 100 or more ECUs, which manage operations ranging from the basic (engine and power steering control) to the luxurious (power windows, seats, and HVAC), to security and access (such as door locks and key less entry). ECUs are also in charge of passive safety systems like airbags, as well as fundamental active safety features like automated emergency braking.

Each ECU has its own dedicated chip that runs its own software or firmware and requires power and data connections to function.

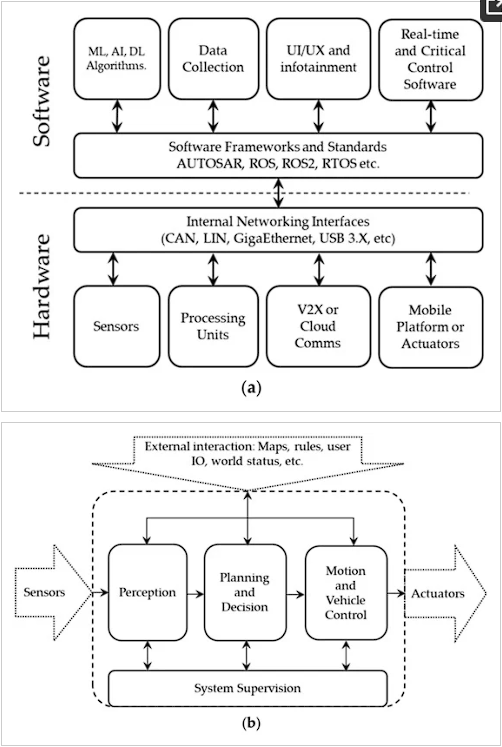
All ECUs in autonomous vehicles can be roughly divided into two categories:

1. ECUs responsible for ADAS and controlling car operations such as steering, shifting gears, braking, and fuel consumption, and ECUs responsible for ADAS and controlling car operations such as steering, switching gears, braking, and fuel consumption. Various real-time operating systems can control these ECUs (RTOS).
2. ECUs control A/V entertainment systems and run a variety of applications. These ECUs can be controlled by operating systems that are more akin to those found on personal computers. Not just RTOSs, but also general-purpose operating systems, can be used (GPOS).

The art of marrying the data from multiple sensors in real time enables a ECU to determine the state of the vehicle and its surroundings. Individual shortcomings of each sensor type cannot be overcome by using more of the same sensor, so data from different sensor types must be combined to mask the shortcomings in the individual sensors. In short, the whole is greater than the sum of its parts. By fusing the sensors into an integrated system, engineers enable the vehicle to identify its surroundings and threats with greater accuracy.

Autonomous vehicles rely on a sophisticated combination of hardware and software systems to navigate, perceive their surroundings, make decisions, and control vehicle functions. Operating systems play a crucial role in managing these complex tasks. Here's a simplified overview of how autonomous vehicles work using an operating system:

Figure 3:WORK USING OPERATING SYSTEM



# **3.2.1 SOFTWARE(**RTOS **FOR AUTONOMOUS VEHICLE)**

1. **QNX**

QNX, renowned for its reliability, determinism, and scalability, is a leading choice for safety-critical applications like autonomous vehicles. Its microkernel architecture, message passing, memory protection, and real-time scheduling are crucial for supporting mission-critical systems. With a modular design and fault-tolerant architecture, QNX excels in distributed computing environments, ensuring high availability.

The hypervisor's minimal performance overhead, typically under 2%, and rapid boot times for guests, reduced to tens of milliseconds, underscore its efficiency. Supporting hardware optimization on Intel x86\_64 VT-x and ARMv8 AArch64 hardware, it caters to various automotive reference boards, including Intel Atom processor C3000 product family, Renesas R-Car H3, and Qualcomm Snapdragon 820A.2.

Figure 4: QNX Neutrino RTOS

A screenshot of a computer

Description automatically generated

2.**Linux**

Automotive Grade Linux (AGL) is a specialized version of the Linux operating system tailored for use in automobiles. It serves as a collaborative platform where car manufacturers, tech companies, and other stakeholders come together to enhance and standardize car software development. AGL facilitates the creation of innovative features such as music systems, interactive screens, and even components for autonomous driving. By providing a common set of tools and standards, AGL accelerates the development of new automotive technologies.

Supported by the Linux Foundation, AGL ensures that car software operates seamlessly and efficiently. It acts as a robust foundation, enabling various stakeholders to contribute their unique ideas without compromising system integrity. In essence, AGL serves as the catalyst for making cars smarter and more enjoyable to drive with each passing day.

Figure 5: AGL(Automotive Grade Linux)

A computer application form with text

Description automatically generated

3. **AUTOSAR (Automotive Open System Architecture**)

AUTOSAR, short for Automotive Open System Architecture, stands as a standardized framework designed by automotive industry stakeholders to address the complexities associated with software in contemporary vehicles. It introduces a structured approach by organizing software into layers, each featuring standardized interfaces and protocols. This fosters the development of modular and reusable components, thereby streamlining software development processes.

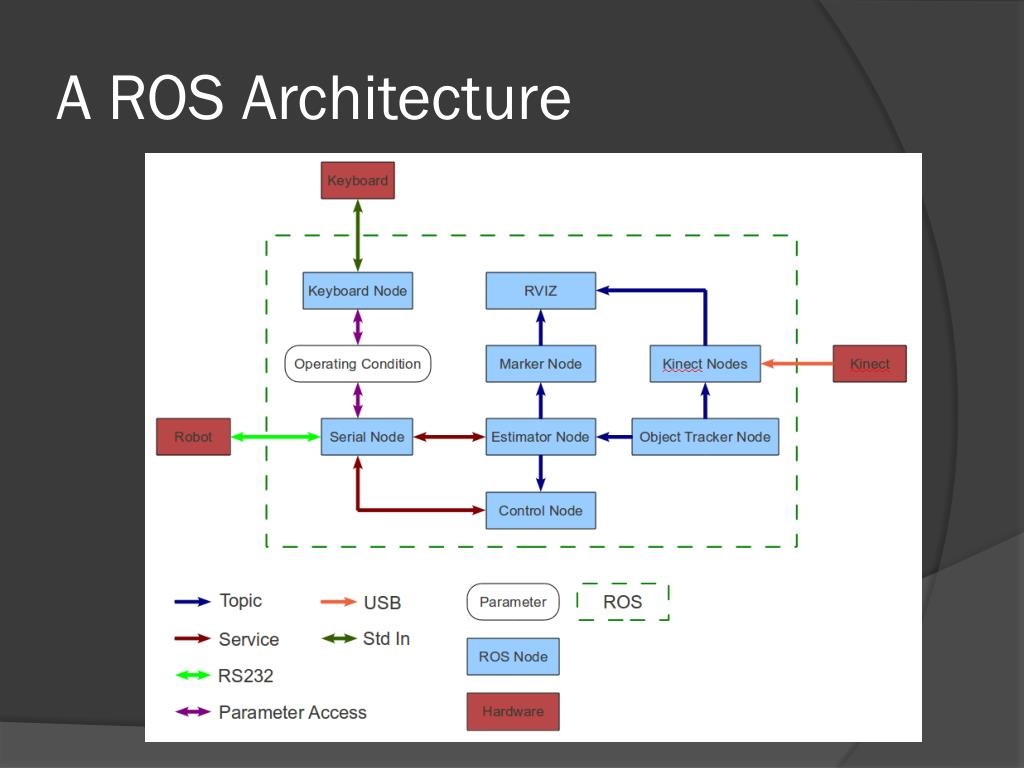
Typically, systems built upon AUTOSAR utilize operating systems such as Linux or specialized real-time operating systems (RTOS) to effectively manage tasks and resources within the vehicle's electronic control units (ECUs). The architecture of AUTOSAR comprises several layers, including the application layer, runtime environment (RTE), and basic software (BSW). The BSW layer is further subdivided into services, electronic control unit (ECU) abstraction, and microcontroller abstraction.

One of the key aspects of AUTOSAR is the Virtual Functional Bus (VFB), which facilitates communication among various software components while ensuring hardware-independent development. This feature plays a pivotal role in enabling seamless interoperability among different components within the vehicle's system.

4. **ROS (Robot Operating System)**

Prominent open-source middleware framework extensively used in robotics and the development of autonomous systems. Its flexible and modular architecture is well-suited for creating sophisticated robotic systems, spanning from autonomous vehicles to drones and industrial robots. With a comprehensive suite of libraries, tools, and communication mechanisms, ROS simplifies the development, testing, and deployment of robotic applications, earning popularity among researchers and developers in the autonomous vehicle field.

FIGURE 6: Architecture Of ROS(Robot Operating System)



5.**NVIDIA's DRIVE**

AGX platform, in conjunction with ROS-based platforms like Clear path Robotics' Warthog, leverages the Robot Operating System (ROS) for prototyping, testing, and deploying autonomous vehicle applications. The NVIDIA DRIVE OS Software Development Kit (SDK) furnishes developers with all the essential tools required for constructing and deploying applications for autonomous vehicles across a spectrum of hardware components.

The DRIVE AGX Developer Kit, powered by this SDK, empowers developers to craft Advanced Driver Assistance Systems (ADAS) and in-car AI applications, thereby enhancing both safety and convenience in automotive environments. Additionally, NVIDIA DRIVE Infrastructure furnishes substantial computational power essential for training and validating self-driving algorithms, thereby accelerating the pace of development and propelling advancements in autonomous vehicle technology.

Figure 7:Nvidia’s Drive Software

.A screenshot of a computer

Description automatically generated

**6.Hybrid architecture** is like blending different ingredients to create a recipe that suits your needs perfectly. It combines various elements like network setups, cloud models, or computing systems to make systems more flexible and adaptable. For example, in cloud computing, it's like mixing different types of clouds to meet specific goals. Similarly, in autonomous vehicles, hybrid software architectures mix open-source and proprietary systems to balance flexibility and safety.

**3.2.1 HARDWARE**

1. **Sensors and Perception**: Autonomous vehicles are equipped with various sensors such as LiDAR, radar, cameras, and ultrasonic sensors to perceive their environment. These sensors collect data about the vehicle's surroundings, including road conditions, obstacles, traffic signs, and other vehicles.

2. **Data Processing**: The data collected by sensors are processed in real-time by onboard computer systems. Operating systems manage these processes efficiently, allocating resources to different tasks such as sensor data fusion, object detection, and environment mapping.

3. **Decision Making:** Based on the processed data, the autonomous vehicle's software makes decisions about navigation, speed control, lane changes, and other driving manoeuvrers. Operating systems provide the framework for decision-making algorithms to run effectively, ensuring safety and efficiency.

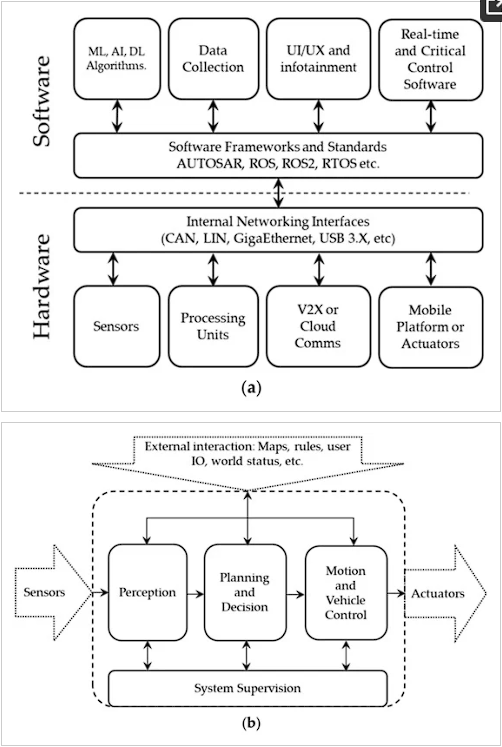
4. **Control Systems:** Once decisions are made, control systems actuate various vehicle components such as steering, braking, and acceleration. Operating systems facilitate communication between decision-making algorithms and control modules, ensuring precise execution of commands.

5. **Localization and Mapping:** Autonomous vehicles continuously update their internal maps and determine their precise location within the environment. Operating systems manage localization algorithms that integrate sensor data with pre-existing maps to achieve accurate positioning.

6. **Communication and Connectivity**: Autonomous vehicles often communicate with other vehicles, infrastructure, and cloud-based services to exchange information about road conditions, traffic patterns, and potential hazards. Operating systems handle communication protocols and networking tasks, enabling seamless interaction between the vehicle and external systems.

7. **Redundancy and Fail-Safe Mechanisms**: Operating systems play a critical role in implementing redundancy and fail-safe mechanisms to ensure the reliability and safety of autonomous vehicles. They manage redundant sensor inputs, backup systems, and emergency protocols to mitigate the impact of system failures or unexpected events.

Figure 8:harware steps of AV(autonomous vehicle)



**3.3Challenges Encountered:**

**Developing autonomous vehicles using an operating system (OS) presents several formidable challenges, necessitating a comprehensive approach to address the intricate demands of this cutting-edge technology**:

1.Real-time Processing: The OS must adeptly handle the exigencies of real-time data processing, encompassing tasks such as **sensor fusion**, object detection, and rapid decision-making. Ensuring the expeditious execution of critical functions is imperative for safe autonomous driving.

2. **Safety and Reliability**: Autonomous vehicles demand unwavering reliability across diverse environmental scenarios, including inclement weather, variable road conditions, and unforeseen events. The OS must furnish robust mechanisms for fault tolerance, error mitigation, and swift recovery to safeguard passengers and pedestrians alike.

3**. Security**: In an era of heightened cybersecurity threats, autonomous vehicles are prime targets for malicious intrusions, encompassing malware incursions, hacking endeavours, and surreptitious data breaches. The OS must fortify the vehicle's systems, communication channels, and sensitive data with formidable security protocols to stave off unauthorized access and manipulation.

4. **Resource Management**: Optimal allocation and utilization of computational resources—comprising CPU cycles, memory bandwidth, and network capacity—are pivotal for ensuring peak performance and scalability. The OS should orchestrate resource allocation dynamically, prioritizing critical tasks and adapting seamlessly to evolving computational demands.

5. Integration with Hardware and Middleware: Seamless integration with a diverse array of sensors, actuators, and **communication** interfaces is indispensable for enabling autonomous vehicles to perceive and interact with their surroundings effectively. The OS must seamlessly interface with hardware components and middleware frameworks, facilitating streamlined sensor data processing, localization, mapping, and control.

6. Interoperability and Standardization: Embracing industry-standard protocols and fostering interoperability between disparate vehicle platforms and components are essential imperatives for catalyzing collaboration, innovation, and ecosystem development within the autonomous driving domain. Adherence to standardized protocols augments compatibility, propelling the industry towards a harmonized and cohesive future.

**IV. RESULTS AND DISCUSSION**

An essential element in the race toward self-driving automobiles is the operating system (OS), which The journey towards self-driving cars hinges on a critical element: the operating system (OS). This software maestro orchestrates the complex tasks of perception, decision-making, and control. Selecting the right OS from contenders like Linux, AGL, NVIDIA's DRIVE OS, ROS, RTOS, and QNX is a crucial decision. Each offers unique strengths, but also carries limitations when considering the challenges faced by autonomous vehicles, such as data management, sensors, security, communication, and safety.

* **Linux:** Open-source and highly customizable, Linux is a popular choice due to its flexibility. However, ensuring guaranteed real-time processing and safety can be a challenge.
* **AGL (Automotive Grade Linux):** Built on the foundation of Linux, AGL caters specifically to automotive needs. It prioritizes safety, security, and standardized interfaces, offering improved compatibility for car applications. However, compared to vanilla Linux, it may lack some flexibility.
* **NVIDIA's DRIVE OS:** Leverages NVIDIA's expertise in AI and high-performance computing, making it ideal for advanced functionalities like deep learning and sensor fusion. However, its reliance on NVIDIA hardware restricts its broader ecosystem compatibility.
* **ROS (Robot Operating System):** As a popular open-source middleware platform, ROS excels in rapid prototyping due to its focus on sensor data management and decision-making. However, achieving real-time performance and safety certification for critical applications in autonomous vehicles can be complex.
* **RTOS (Real-Time Operating System):** Prioritizes deterministic real-time processing, a fundamental aspect of safety-critical functions. Options like FreeRTOS and VxWorks offer guaranteed response times, but may lack the versatility and ecosystem support found in general-purpose OSes.
* **QNX:** Renowned for reliability, security, and real-time capabilities, QNX has a strong presence in automotive embedded systems. While ideal for safety-critical applications, its licensing costs and potentially closed ecosystem could be drawbacks.

**Challenges and Considerations for Autonomous Vehicle Functions:**

* **Data Management:** The sheer volume of sensor data demands efficient processing, fusion, and distribution for accurate perception and decision-making. All OSes need robust solutions to handle this data deluge.
* **Sensor Integration:** Seamless integration with diverse sensors (cameras, LiDAR, radar) is crucial for accurate environmental understanding and precise vehicle control. The chosen OS should interact smoothly with this complex sensor network.
* **Security:** Cybersecurity threats loom large. The OS must safeguard vehicle systems, communication channels, and sensitive data from unauthorized access or manipulation. Robust authentication, encryption, and secure communication protocols are essential.
* **Communication:** Autonomous vehicles rely on communication with their surroundings. The OS should support V2V (vehicle-to-vehicle) and V2I (vehicle-to-infrastructure) communication protocols. Quality-of-service (QoS) mechanisms can prioritize critical messages, ensuring minimal latency and reliable delivery.
* **Safety:** Safety is paramount. The OS can contribute significantly by incorporating real-time scheduling, fault detection/recovery mechanisms, and redundancy in sensor inputs and algorithms. Rigorous testing, simulation, and validation processes are crucial for ensuring safety compliance.

Figure 9:Comparison of Functions and Operating System

A screen shot of a computer

Description automatically generated

**VI. Future Works**

The future of autonomous vehicles, powered by sophisticated operating systems (OS), holds immense promise across various domains. Advancements in sensor integration will enable vehicles to gather and process data from diverse sources with unprecedented accuracy and speed, enhancing their perception capabilities. Integrating artificial intelligence (AI) and machine learning into OS will empower vehicles to make complex decisions in real-time, ensuring safe and efficient navigation through challenging environments. Furthermore, improved connectivity and communication technologies will enable seamless interaction between vehicles, infrastructure, and the cloud, enhancing safety and optimizing traffic flow. Addressing cybersecurity and data privacy concerns will be paramount, ensuring the integrity and protection of sensitive information exchanged within the vehicle ecosystem. Compliance with evolving regulatory standards and ethical considerations will guide the development of OS, fostering trust and acceptance of autonomous driving technology. Additionally, enhancing user experience through intuitive interfaces and advanced driver assistance features will play a crucial role in increasing adoption. Collaboration with infrastructure providers will be essential to develop the necessary infrastructure to support autonomous vehicles' widespread deployment. Continued research and innovation will drive advancements in OS technology, enabling vehicles to meet evolving demands for real-time processing, fault tolerance, energy efficiency, and scalability. In essence, the future of autonomous vehicles using OS is characterized by transformative advancements that will revolutionize transportation and shape the future of mobility.

**VII. CONCLUSION**

The development of autonomous vehicles (AVs) has been fueled by the potential to revolutionize transportation, promising increased accessibility, reduced crashes, energy consumption, pollution, and congestion. Advancements in communication and robotics technologies have accelerated this development, with the operating system (OS) serving as a critical component influencing the performance, safety, and reliability of AVs. In addressing the challenges of developing autonomous vehicles, various operating systems have been explored, including QNX, Linux (AGL), AUTOSAR, Nvidia Drive Os, ROS, and RTOS. Each OS offers unique functionalities and capabilities, contributing to the overall efficiency and effectiveness of AVs. By slimming down kernel functions and providing real-time operating system (RTOS) functionality, OSs like QNX play a crucial role in optimizing AV performance.

Beyond technical challenges, the adoption of AVs presents policy and societal hurdles, such as regulatory frameworks, public acceptance, and ethical considerations. Research efforts have been directed towards overcoming these challenges, aiming to ensure a smooth and successful integration of AVs into our transportation system. The research on autonomous vehicles using various operating systems has made significant strides in addressing both technical and non-technical challenges. The advancements achieved in this field hold promise for shaping the future of transportation, making it safer, more efficient, and more accessible for all. As research continues to evolve, the transformative potential of autonomous vehicles in revolutionizing transportation systems will become increasingly apparent.

.

# **References**

**[1]** RTOS for Automotive vehicles, <https://blackberry.qnx.com/en/ultimate-guides/software-defined-vehicle/automotive-rtos> , 04-03-2024.

**[2]** The Privacy Implications of Autonomous Vehicles, <https://www.dataprotectionreport.com/2017/07/the-privacy-implications-of-autonomous-vehicles/> , 05-03-2024.

**[3]** Operating System Technology Trends for Autonomous Vehicles, <https://www.idpublications.org/wp-content/uploads/2022/12/Full-Paper-OPERATING-SYSTEM-TECHNOLOGY-TRENDS-FOR-AUTONOMOUS-VEHICLES.pdf> ,05-03-2024.

**[4]** Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review, <https://www.dataprotectionreport.com/2017/07/the-privacy-implications-of-autonomous-vehicles/> **,** 06-03-2024.

**[5]** Autonomous vehicles: challenges, opportunities, and future implications for transportation policies, <https://link.springer.com/article/10.1007/s40534-016-0117-3> , 07-03-2024.

**[6]** NVIDIA DRIVE OS, <https://developer.nvidia.com/drive/os> , 08-03-2024.

**[7]** QNX Hypervisor,<https://blackberry.qnx.com/content/dam/qnx/products/hypervisor/hypervisor-product-brief.pdf> , 08-03-2024.

**[8]** Automotive Grade Linux (AGL) documentation, <https://docs.automotivelinux.org/en/pike/> ,09-03-2024.

**[9]** Automotive open system Architecture, <https://www.autosar.org/> , 10-03-2024.

**[10]** Robot Operating System, <https://en.wikipedia.org/wiki/Robot_Operating_System> , 10-03-2024.

**[11]** Hybrid Architecture, <https://www.sciencedirect.com/topics/computer-science/hybrid-architecture> ,10-03-2024.

**[12]** QNX Real Time Safety-Certified OS, <https://www.directinsight.co.uk/qnx-real-time/> ,11-03-2024.

**[13]** Evaluating perceived safety of autonomous vehicle: The influence of privacy and cybersecurity to cognitive and emotional safety, <https://www.sciencedirect.com/science/article/pii/S0386111223000262> ,15-03-2024.

**[14]** An Intelligent Task Scheduling Mechanism for Autonomous Vehicles via Deep Learning, <https://www.mdpi.com/1996-1073/14/6/1788> ,16-03-2024.

**[15]** Autonomous Vehicles Enabled by the Integration of IoT, Edge Intelligence, 5G, and Blockchain, <https://www.mdpi.com/1424-8220/23/4/1963> ,17-03-2024.

**[16]** Efficient Data Privacy and Security in Autonomous Cars, <https://pubs.sciepub.com/jcsa/7/1/5/index.html> ,18-03-2024.

**[17]** Security and Privacy Issues in Autonomous Vehicles: A Layer-Based Survey, <https://ieeexplore.ieee.org/document/9762777> ,19-03-2024.

**[18]** ROS Architecture Image, <https://image1.slideserve.com/1575646/a-ros-architecture-l.jpg> ,22-03-2024.