Development and Validation of an Open Architecture for Autonomous Vehicle Control

1st Alfredo Valle Barrio
University Institute for Automobile Research, INSIA
Universidad Politécnica de Madrid, UPM
Madrid, Spain
alfredo.valle@upm.es

2nd Walter Morales Alvarez
Intelligent Transport Systems,
ITS-Sustainable Transport Logistics 4.0.
Johannes Kepler Universität Linz, JKU
Linz, Austria
walter.morales_alvarez@jku.at

3rd Cristina Olaverri-Monreal
Intelligent Transport Systems,
ITS-Sustainable Transport Logistics 4.0.
Johannes Kepler Universität Linz, JKU
Linz, Austria
cristina.olaverri-monreal@jku.at

4th José Eugenio Naranjo Hernández
University Institute for Automobile Research, INSIA
Universidad Politécnica de Madrid, UPM
Madrid, Spain
joseeugenio.naranjo@upm.es

***Abstract*—Teams dedicated to research in the field of autonomous vehicles can be found in many universities and research centers, but they often face the challenge of finding a suitable platform for their work. The primary reason for this challenge is the inaccessibility and high cost of commercial autonomous vehicles, leading researchers to rely on simulators.**

This paper introduces a new software architecture designed to automate vehicles, providing all the necessary capabilities of an autonomous vehicle in a more cost-effective and efficient manner. The architecture is designed to be modular, universal, and with a public interface, making it easy to modify, adapt to any type of vehicle, and accessible to any researcher.

The new software architecture has been implemented in two platforms: a vehicle integrated with OpenPilot via ROS2 without any external hardware, and a last-mile robot. A validation test was conducted with volunteers to assess the reaction of passengers while the car was driving autonomously. The results of this implementation demonstrate the potential of this new software architecture to provide a comprehensive and accessible platform for the advancement of autonomous vehicle research.

***Index Terms*—ROS2, openpilot, ROSbridge, Autonomous vehicle, Low Level Control**

I. Introduction

The field of autonomous vehicles has seen tremendous growth in recent years [1], with a large number of universities and research centers dedicating teams to exploring this cutting-edge technology. However, despite the progress made in the field, researchers are still encountering significant challenges to find suitable platforms for their work. One of the most common obstacles faced by researchers is the lack of access to commercially available automated vehicles or the high cost of these vehicles, which makes them unaffordable for many research teams. As a result, many researchers are limited to simulators, which lack the real-world capabilities of autonomous vehicles.

To overcome these challenges, this paper presents a new software architecture designed specifically for automating vehicles in a more efficient and cost-effective manner. The architecture is designed to be modular, universal, and easily adaptable to different vehicles, making it an ideal platform for further innovation. The software architecture has been devised with a modular approach, providing ease of modification and upgradability, and a universal design, enabling rapid adaptation to various vehicle types. The public interface of the architecture also makes it accessible to any researcher, providing a platform for further innovation and development.

The implementation of the proposed software architecture was carried out on a platform at the Johannes Kepler Universität (JKU) in Linz. The platform is a vehicle that utilizes OpenPilot for actuator control and Robot Operating System (ROS)2 for communication [2].

Openpilot is an open-source software platform developed by Comma.ai that provides basic autonomous driving capabilities for compatible vehicles. It enables drivers to experience advanced driver-assistance systems such as adaptive cruise control, lane keeping assistance, and automatic emergency braking using sensors such as cameras, radars, and GPS. Its open-source nature allows developers to modify and enhance its code, making it a popular option in the autonomous driving community for its low cost, flexibility, and ease of use.

ROS2 is a software framework for building and developing robot applications that provides a set of tools, libraries, and conventions. It supports multiple programming languages, enabling the creation of distributed systems that can communicate with each other using a publish-subscribe model. ROS2's architecture emphasizes modularity, reusability, and interoperability, making it a widely adopted platform in the robotics community for various applications.

The implementation in the vehicle was validated through a field test with volunteers, who were seated as co-pilots while it drove autonomously. The results of the validation demonstrated the feasibility of the proposed software architecture and its potential to serve as a platform for further innovation in the field of autonomous vehicles. With this new software architecture, researchers can now pursue their work with greater confidence, knowing that they have access to a cost-effective and efficient platform for their research.

In addition to the benefits outlined above, the proposed software architecture offers several other advantages over existing platforms. Firstly, it provides researchers with a flexible and scalable to different and multiple platforms that can be easily adapted to their specific needs. This is particularly important in a rapidly evolving field such as autonomous vehicles, where new technologies and techniques are constantly being developed. The modular design of the architecture also allows an easy integration of new components, which can be added or updated as needed. Furthermore, the presence of a public interface allows for multiple vehicles equipped with this architecture to be controlled as a platoon or individually, even if they have not been automatized in the same way.

Another key advantage of the proposed software architecture is its cost-effectiveness. Many commercially available autonomous vehicles are prohibitively expensive for most research teams, making it difficult for them to access the hardware they need for their work. By providing a cost-effective platform for autonomous vehicle research, the proposed software architecture helps to level the playing field and make it easier for smaller research teams to participate in this exciting and important field.

In conclusion, the proposed software architecture for autonomous vehicles is a major step forward in the field, providing researchers with a flexible, scalable, and cost-effective platform for their work.The integration of Openpilot into the software control architecture enables the automation of more than two hundred vehicle models without requiring additional hardware.

With its modular design and public interface, the architecture contributes to driving further innovation in this exciting field of autonomous vehicles. The results of the validation test demonstrate the feasibility of the architecture and its potential to serve as a platform for further research and development.

II. Platforms

The JKU-ITS research team has provided a platform as the basis for the work presented in this article, with the aim of demostrating the versatility and adaptability of the proposed architecture. By utilizing this platform, the team demonstrates the architecture's capability to be applied to diverse scenarios and configurations, highlighting its robustness and flexibility. This platform serves to demonstrate the utility of the architecture, emphasizing its ability to be easily adapted to platforms with diverse specifications and requirements. The proposed solution is anticipated to provide valuable insights into the potential of the proposed architecture and emphasize its suitability for implementation in a wide range of real-world applications.

A. *Vehicle*

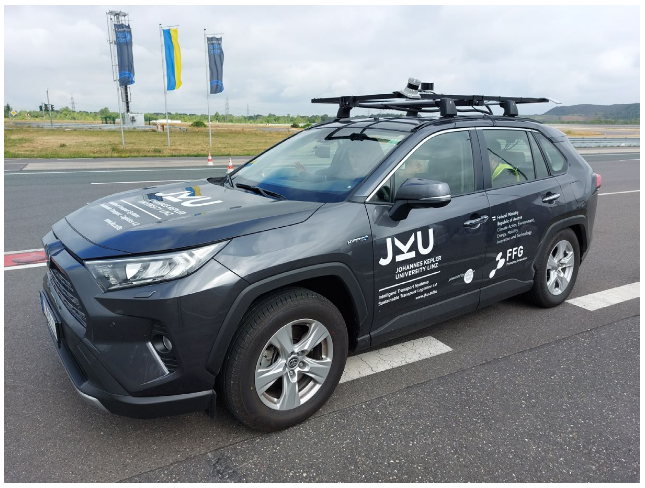

Fig. 1. Vehicle Toyota RAV4 from the JKU-ITS team

The Toyota RAV4 vehicle [2], shown in Fig.1, has been equipped with Openpilot, an open-source software developed by Comma.AI, that provides to the vehicle the autonomous capabilities. The software accomplishes this feat by reverse engineering the Toyota CAN database and supplanting the acceleration and torque commands, thus providing the vehicle with SAE Level 2 functionality. The implementation of Openpilot causes the vehicle to perceive it as the original Adaptive cruise control (ACC) and Lane Keeping System (LKS), thereby facilitating control of the actuators. Additionally, Openpilot provides the user with insightful information regarding the state of the vehicle, such as acceleration command, applied torque on the steering mechanism, vehicle velocity, steering wheel angle, and alert notifications.

In the context of the JKU-ITS vehicle, the Openpilot code responsible for sending the acceleration and torque commands was modified to integrate with other modules through the utilization of ROS2 [4]. The code underwent adaptation to create two ROS2 topics, the first of which provide an interface with ROS2 to transmit the acceleration and torque commands to the vehicle through Openpilot. The second topic transmits state information obtained by Openpilot from the vehicle's CAN network through ROS2.

III. Development

The purpose of this work is to enhance the software architecture, as outlined in a previous publication [5], and integrate Openpilot as the control interface for the vehicle's actuators. The previous architecture enabled the automation of vehicles through the use of standardized external actuators [6], being in-vehicle actuators only used in rare exceptions and even in this case, without directly interacting with the electronic control unit (ECU) and vehicle control system.

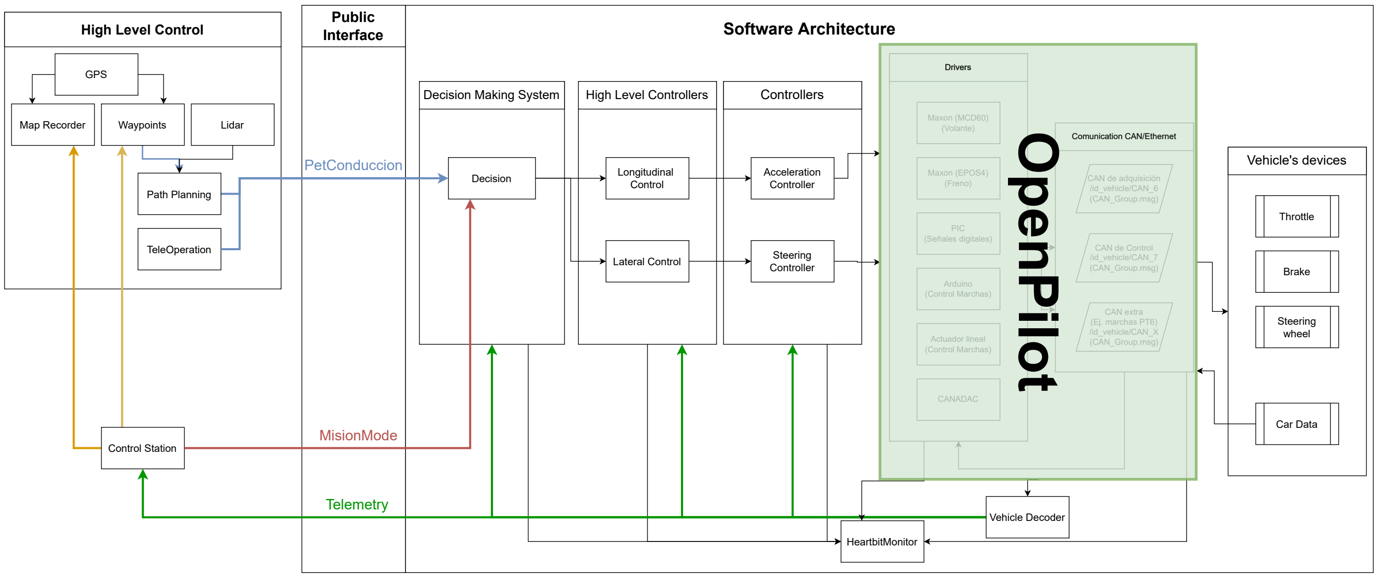

Fig. 2. Software architecture

The present work describes the integration of Openpilot into vehicle control systems through the Advanced Driver Assistance Systems (ADAS). By using ROS2 to connect Openpilot, the lowest levels of the control architecture proposed have been successfully substituted, resulting in a versatile software solution that can be adapted to a variety of vehicles without any physical alterations. It provides a development platform for signal control software and for the integration with other vehicles [7] and research centers.

A. *Software architecture*

The architecture employed in this study is depicted in Fig. 2. As demonstrated in the illustration, the architecture comprises five levels, of which the two lowest levels, the driver management and communication with devices, have been substituted by Openpilot.

Sitting above these levels is the controllers layer, which must be adapted for each vehicle in the event of a change to the actuator system. In this layer, standardized control messages from the higher levels are transformed into messages that are specific to the installed controllers. Given that gear control and any device other than acceleration and steering wheel rotation is not feasible, the architecture has been simplified.

The next layer contains the longitudinal and lateral signal controllers, which utilize inputs and outputs that are normalized to ± 1. This normalization enables the controllers to be quickly swapped with other controllers, such as those based on fuzzy logic or proportional-integral-derivative (PID) controllers, for testing in a real-world environment.

Finally, at the core of the architecture is the low-level decision layer, which is intended to accommodate both a teleoperation signal and an autonomous control signal. The chosen mode is transmitted to the lower level layers, passing through a security system that check valid messages. If a disconnection or node failure occurs, the vehicle will be stopped and the error will be reported. The security system also includes a manual override feature for individual devices, allowing for greater control during field tests.

All communication with the control architecture takes place through this decision node, owing to the standardized public interface that is implemented for all vehicles. This interface consists of only three custom ROS2 messages, PetConduccion, MissionMode, and Telemetry as shown in Table I.

TABLE I
PUBLIC INTERFACE CUSTOM MSG

PetConduccion.msg		ModoMision.msg		Telemetry.msg	
Type	**Name**	**Type**	**Name**	**Type**	**Name**
string	id_plataform	string	id_plataform	string	id_plataform
std_msgs/Header	header	uint8	MANUAL=00	float64	speed
float64	steering	uint8	AUTONOMO=01	float64	steering
float64	speed	uint8	TELE_OPERADO=02	float64	steering_deg
float64	gear	uint8	modo_mision	uint8	throttle
bool	b_brake			uint8	brake
bool	b_throttle			string	gears
bool	b_steering				
bool	b_gear				

The PetConduccion message serves to transmit control commands to the vehicle. It enables each device individually with fields starting with ”b_” and sends the commands, including the selected gear, although this does not have an effect on this particular car.

The MisionMode message selects the vehicle's operating mode, which can be Manual, TeleOperated, and Autonomous.

The Telemetry message contains the current parameters of the vehicle, such as speed, steering angle, engaged gear, and accelerator and brake signals.

In this work, the high-level control system, shown in the left part of Fig. 2, has been integrated into the control architecture to provide the vehicle with autonomous capabilities. This system comprises a teleoperation node connected to a control controller, which allows remote control of the vehicle through the use of ROS2, eliminating the need for a direct wireless connection such as Bluetooth.

The autonomous branch of the system consists of two nodes that generate driving commands. One of these nodes, the waypoint node, uses an Real-time kinematic positioning (RTK) inertial Global Position System (GPS), while the other, the lidar node, enables obstacle detection and driving in enclosed environments. These commands are integrated in the PathPlanning node, which makes high-level decisions that are then transmitted to the control architecture.

Additionally, a map recorder node has been implemented to record the routes to be followed in autonomous mode using the GPS signal. To complete the waypoint node, a control station has been set up to centralize all signals, including the messages that manage the waypoints. While it has been designed as a single entity for this project, the control station can also be distributed into multiple nodes that manage the waypoints, decision making, and teleoperation separately.

B. *Openpilot integration*

1) Advantages of use Openpilot: The integration of Openpilot into the software control architecture offers a significant advantage by enabling automation of more than two hundred vehicle models without the requirement for additional hardware. The only necessary component is the installation of a CAN data acquisition device. Allowing fast prototyping on all these vehicles. [8]

2) Limitations with respect to external automation: In spite of all the advantages, the execution of this work has encountered several significant limitations, especially when utilized in an uncontrolled environment. This is due to its reliance on the proper functioning of the vehicle's factory control systems, which can result in hazardous scenarios. As exemplified in the vehicle utilized in this study, the activation of the traction control system in the presence of a pothole or a slippery surface such as salt on an icy road can result in the automatic deactivation of the traction control system without any prior warning or opportunity for the driver to take the control.

IV. Validation

To validate the work presented we conducted an experiment that aim at simultaneously observing the performance of the autonomous vehicle in operation and quantifying, in an objective manner, the reactions and perceptions of individuals serving as passengers in the vehicle.

The results of this experiment provided valuable insights into the performance and acceptability of autonomous vehicles, thereby contributing to the ongoing discourse on this rapidly evolving technology.

A. *Testing Track*

The experiment was conducted on a track behind the Johannes Kepler Universität at Linz in Austria, that is approximately 250 meters in length and takes about 1 minute to travel. The route is shown in Fig. 3 and, although it may seem simple, it includes all the essential control elements that an autonomous vehicle must have as ability to stay in the lane with very limited lateral error and pronounced turns to perform obstacle avoidance. During the experiment, the vehicle's performance in controlling acceleration and deceleration, maintaining lane alignment through steering control, and responding to obstacles was evaluated. This was accomplished by simulating an obstacle avoidance scenario, marked in red, where the vehicle was required to safely change lanes and quickly return to its original lane.

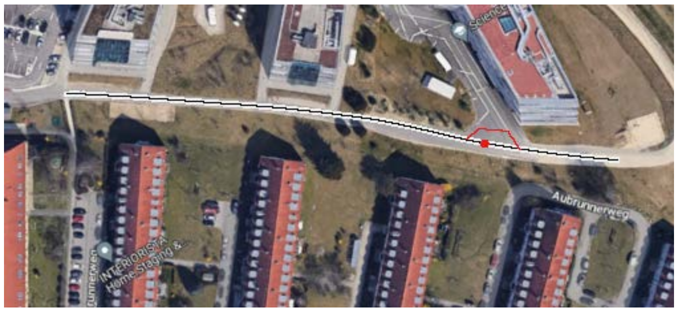

Fig. 3. Location of the testing track at the JKU in Linz, Austria

B. *Experimental Design*

The sample consisted on participants that were recruited at the campus and that included students, faculty, researchers and also citizens from Linz that were walking in the vicinity of the track. A total of 26 individuals expressed interest, but only 24 of them participated due to technical difficulties. To ensure a representative sample, efforts were made to select participants with diverse demographic backgrounds, including age, years of driving experience, and nationality, while maintaining the confidentiality and anonymity of the participants. For example, the diversity of nationalities among the participants is shown in Fig. 4.

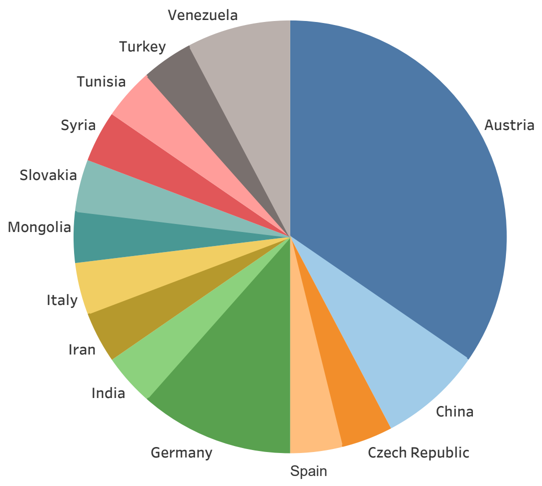

Fig. 4. Nationalities of Participants: Pie Chart Representation

1) Non related driving task: The participants in the experiment were asked to perform a non-driving-related task (NDRT) on a mobile phone while they occupied the co-driver's seat and the car had the autopilot activated to drive autonomously without a driver. The objective of the experiment was to find out if there was any kind of reaction from the side of the participants when the car maneuvered to avoid an obstacle on the road. For example, something that would denote nervousness or lack of confidence in the automation, such as looking up from the phone on which they were performing the tasks. The task assigned to them consisted on playing a game that demanded a consistent level of attention, but provided brief one-second intervals during which they could quickly check the road. The game's difficulty increased progressively, eventually reaching a critical point where maximum concentration was required. This critical point was reached at the moment in which the obstacle needed to be avoided by the vehicle.

2) Apparatus: The participants in the experiment additionally used Tobii Pro Glasses 2, a state-of-the-art eye tracking device composed of a pair of glasses and a Central Processing Unit (CPU). The device features two infrared sensors (IR) and cameras for each eye, enabling the acquisition of gaze and pupil data.

V. Results

The results from the analysis of the collected data during the experiment are presented by showing the trajectory followed by the vehicle during the test and its corresponding control signals, and the confidence of the participants during the experiment.

With regards to the control of the vehicle, Fig. 5 provides a visual representation of its performance during the test. It can be observed that the vehicle initiates a progressive acceleration to reach its target speed of 15 km/h, and subsequently employs a smooth braking mechanism at the end of the route. The speed is depicted on a color scale for clarity. At the moment of widening of the road, the vehicle successfully navigates around the obstacle placed in the lane, returning to the lane before the widening ends.

Fig. 6 further demonstrates the vehicle's ability to stay on the road without deviating from the lane. It can be seen that the vehicle makes only the minimum required corrections to maintain its trajectory, with a sharper turn executed during the avoiding maneouver to move away from the obstacle heading to the left.

The results obtained from the participants were also analyzed as part of the experiment. Eye-tracking systems were utilized to study the participants' visual behavior and draw conclusions about their state of visual attention [9], [10]. Attentional variables such as pupil diameter, gaze position in space, saccades, and fixations [11], which have been extensively studied by the scientific community, were employed in this experiment.

The experiment was divided into two parts: before and after the obstacle avoidance event. The exact moment of division was defined as the moment when the obstacle became obvious to the participant.

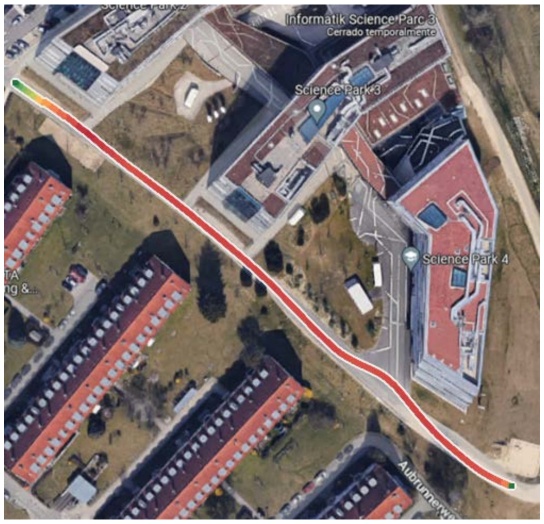

Fig. 5. Speed profile over the test track

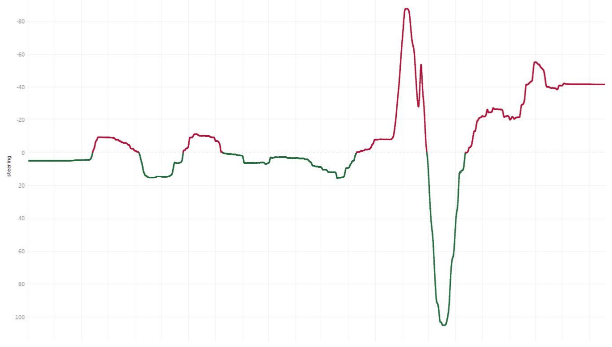

Fig. 6. Steering wheel signal

The pupil diameter and velocity of fixations [12] [13] before the event were used as the base sample, while the data collected after the event was analyzed to study the participants' reaction. Table II presents the Person's correlation coefficient established between the measurements obtained during the test, with statistically significant correlations identified between them.

Fig. 7 highlights the relationship between fixation duration and pupil diameter before and after the event. A value below 1 indicates that the fixation was faster after the event and the pupil diameter larger. In this case, it can be seen that the pupil diameter (represented in blue) did not exhibit a significant result, but the fixation speed did. The average of these data shows that the fixation speed increased by 7% on average, indicating that the participants were aware of the event but were not specially affected by it, emotionally speaking [14]. It is worth noting that none of the participants, even those with a higher acceleration in the fixations, intended to take control of the steering wheel.

TABLE II
CORRELATION BETWEEN MEASURES

	Diameter before event	Fixation duration before event
Diameter after event	p=0.973, p<0.001	-
Fixation duration after event	-	p=0.763, p<0.001

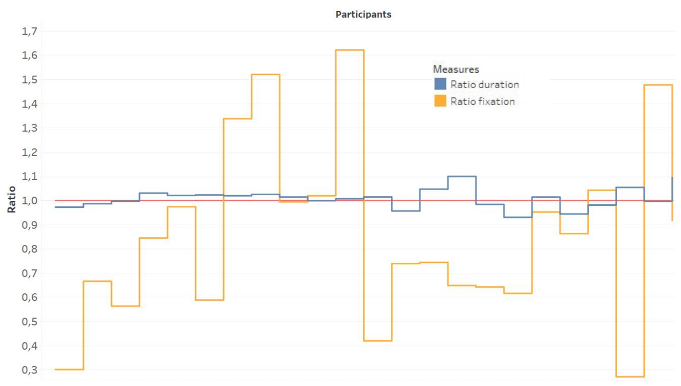

Fig. 7. Ratio Gaze event duration and Pupil diameter

VI. CONCLUSIONS

In conclusion, the software architecture has been implemented in ROS2, incorporating good integration with Openpilot. The end result is a vehicle with factory-based mechanical automation, but with the full range of capabilities of an autonomous vehicle. The efficacy of this implementation was confirmed through testing with 24 participants, serving as passengers in the autonomous vehicle. The results of these evaluations indicate a high level of confidence in the system among the participants, although there were instances of slight apprehensiveness during instances of sudden maneuvers.

Moreover, the proposed architecture presents a cost-effective alternative for researchers who require autonomous platforms for experimental purposes. It facilitates vehicle automation without the requirement of procuring high-priced brand-new vehicles.

This implementation highlights the versatility of the proposed architecture and its ability to seamlessly integrate with existing systems to create innovative solutions. It also underscores the need for continued development and refinement of autonomous vehicle technology to ensure optimal performance and safety. The results of this study contribute valuable insights and information to the ongoing efforts in the field of autonomous vehicles, and provide a foundation for future research and development.

ACKNOWLEDGMENT

The authors would like to extend their sincerest thanks to the Johannes Kepler University (JKU), especially the ITS team, for providing a research environment that allowed for the completion of this work. Our stay at JKU was made possible by the generous support of the Programa Propio of Universidad Politécnica de Madrid. We are grateful for their investment in our research and the opportunities they provided for professional and personal growth. This work would not have been possible without the support and guidance of the ITS team at JKU. We are truly grateful for their contributions and for providing us with the resources and support necessary to achieve our goals.This work was partially supported by the Austrian Ministry for Climate Action, Environment,Energy, Mobility, Innovation, and Technology (BMK) Endowed Professorship for Sustainable Transport Logistics 4.0., IAV France S.A.S.U., IAV GmbH, Austrian Post AG and the UAS Technikum Wien.

REFERENCES

[1] Talavera E, Díaz-Álvarez A, Naranjo JE, Olaverri-Monreal C. Autonomous Vehicles Technological Trends. Electronics. 2021; 10(10):1207. https://doi.org/10.3390/electronics10101207
[2] Certad, N., Morales-Alvarez, W., Novotny, G., Olaverri-Monreal, C. (2022) "JKU-ITS Automobile for research on autonomous vehicles" , Artificial Intelligence and Data Mining for Intelligent Transportation Systems and Smart Mobility, EUROCAST 2022, Las Palmas de Gran Canaria, Canary Islands, Spain.
[3] Novotny, G., Morales-Alvarez, W., Smirnov, N., Olaverri-Monreal, C. (2022) "Development of a ROS-based Architecture for Intelligent Autonomous on Demand Last Mile Delivery", Artificial Intelligence and Data Mining for Intelligent Transportation Systems and Smart Mobility, EUROCAST 2022, Las Palmas de Gran Canaria, Canary Islands, Spain.
[4] Open Robotics. (2023). ROS 2. https://index.ros.org/doc/ros2/
[5] Valle Barrio A., Naranjo Hernández J. E. Madrid, España, (2021). Desarrollo de ecosistema software de control para vehículos autónomos en entornos no estructurados en ROS2. https://oa.upm.es/id/eprint/67451
[6] Felipe Jiménez y col.Equipo para controlar automáticamente la dirección de unvehículo. Patente ES2516568. 2013
[7] Valle Barrio A., Jimenez Alonso F.Application of automation of public road works vehicles in unstructured environments. YRS 2021. Portoroz, Slovenia.
[8] Openpilot supported vehicles. (2023), GitHub, https://github.com/commaai/openpilot/blob/master/docs/CARS.md
[9] Werneke, J. & Vollrath, M. (2012). What does the driver look at? The influence of intersection characteristics on attention allocation and driving behavior. Accident Analysis and Prevention, 45, 610–619. https://doi.org/10.1016/j.aap.2011.09.048
[10] Olaverri-Monreal, C., Hasan, A., Bulut, J., Körber, M., Bengler, K. (2014) "Impact of In-Vehicle Displays Location Preferences on Drivers' Performance and Gaze.", In IEEE Transactions on Intelligent Transportations Systems, vol. 15, no. 4, pp. 1770-1780, Aug. 2014, doi: 10.1109/TITS.2014.2319591
[11] M. A. Recarte, and L. M Nunes, "Effects of verbal and spatial-imagery tasks on eye fixations while driving," Journal of experimental psychology: Applied, 6(1), 31, 2000.
[12] Lemonnier, S., Brémond, R. y Baccino, T. (2014). Discriminating Cognitive Processes with Eye Movements in a Decision-Making Driving Task. Journal of Eye Movement Research, 7 (4), 1–14. https://doi.org/10.16910/jemr.7.4.3
[13] Morales-Alvarez, W., Marouf, M., Tadjine, H. H., Olaverri-Monreal, C. (2021)."Real-World Evaluation of the Impact of Automated Driving System Technology on Driver Gaze Behavior, Reaction Time and Trust", Proceeding IEEE Intelligent Vehicle Symposium 2021, Nagoya, Japan
[14] Vetturi, D., Tiboni, M., Maternini, G. y Bonera, M. (2020). Use of eye tracking device to evaluate the driver's behaviour and the infrastructures quality in relation to road safety. Transportation Research Procedia, 45, 587–595. https://doi.org/10.1016/j.trpro.2020.03.053