World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:18, No:5, 2024

Hands-off Parking: Deep Learning Gesture-Based System for Individuals with Mobility Needs

Javier Romera, Alberto Justo, Ignacio Fidalgo, Javier Araluce, Joshué Pérez

Abstract—Nowadays, individuals with mobility needs face a significant challenge when docking vehicles. In many cases, after parking, they encounter insufficient space to exit, leading to two undesired outcomes: either avoiding parking in that spot or settling for improperly placed vehicles. To address this issue, this paper presents a parking control system employing gestural teleoperation. The system comprises three main phases: capturing body markers, interpreting gestures, and transmitting orders to the vehicle. The initial phase is centered around the MediaPipe framework, a versatile tool optimized for real-time gesture recognition. MediaPipe excels at detecting and tracing body markers, with a special emphasis on hand gestures. Hands detection is done by generating 21 reference points for each hand. Subsequently, after data capture, the project employs the MultiPerceptron Layer (MPL) for in-depth gesture classification. This tandem of MediaPipe's extraction prowess and MPL's analytical capability ensures that human gestures are translated into actionable commands with high precision. Furthermore, the system has been trained and validated within a built-in dataset. To prove the domain adaptation, a framework based on the Robot Operating System 2 (ROS2), as a communication backbone, alongside CARLA Simulator, is used. Following successful simulations, the system is transitioned to a real-world platform, marking a significant milestone in the project. This real-vehicle implementation verifies the practicality and efficiency of the system beyond theoretical constructs.

Keywords—Gesture detection, MediaPipe, MultiLayer Perceptron Layer, Robot Operating System.

I. Introduction

INDIVIDUALS with mobility challenges encounter significant obstacles when parking their vehicles [1]. It is necessary to address this issue with sensitivity, particularly when considering the needs of people with disabilities and the elderly. The goal is to ensure that everyone, regardless of their physical limitations, can access the freedom and independence that come with personal transportation.

As our population ages, the number of elderly drivers is on the rise [2]. A significant portion of them depend on customized cars to cope with restricted mobility. Despite the ability to operate these adapted vehicles, they face difficulties in locating suitable parking areas. Such locations are not only mandated by law but are also essential for those reliant on them.

Statistics indicate an increase in elderly drivers and a corresponding need for adapted vehicles [3]. However, while accessible parking spaces are legally required, actual availability falls short. As a result, individuals with mobility limitations often face the difficult choice of either avoiding inadequate parking options or tolerating inconsiderate vehicle placement.

Javier Romera and Ignacio Fidalgo are with University of Deusto, Avenida Universidades 24. 48007 Bilbao, Bizkaia, Spain (e-mail: jromera, ignacio.fidalgo{@deusto.es}).

Alberto Justo, Javier Araluce and Joshué Pérez are with TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Bizkaia, Spain. (e-mail: alberto.justo, javier.araluce, joshue.perez{@tecnalia.com}).

In this paper, we present a gestural teleoperation-based parking control system designed to address the challenges of controlling parking. The framework and setup schematic of the system can be found in Fig. 1. Our contributions are as follows:

- Remote-controller free exterior control system.
- Real-time gesture recognition using MediaPipe framework and a MultiPerceptron Layer (MPL) for gesture classification.
- A three-phase distributed system: capturing body markers, interpreting gestures, and transmitting orders to the vehicle.
- Real world integration.

The following sections will delve into the details of our parking control system, its development, and the validation process, highlighting its significant impact on the lives of those who rely on it.

II. Related Works

The development of control systems for parking cars has been a topic of extensive research in recent years, with several significant advancements being made.

One of the most notable advancements is the unified motion planning method for parking an automated vehicle in the presence of irregularly placed obstacles. This method [1], proposed by Li and Shao, addresses the complex problem of navigating a vehicle into a parking space while avoiding obstacles. The unified motion planning method represents a significant step forward in the field, providing a comprehensive solution that takes into account the unique challenges presented by irregularly placed obstacles.

Building on this work, Li et al. introduced an optimization-based trajectory planning method [4]. This approach refines the process of planning the vehicle's path into the parking space. By using optimization techniques, this method provides a more efficient and effective solution to the problem of automated parking.

Furthermore, emerging technologies such as haptic displays are beginning to be incorporated into parking systems. For instance, Zhang et al. have explored the use of magnetic field control for haptic display [5]. This technology could potentially be applied to provide drivers with tactile feedback during the parking process, enhancing user experience and improving safety.

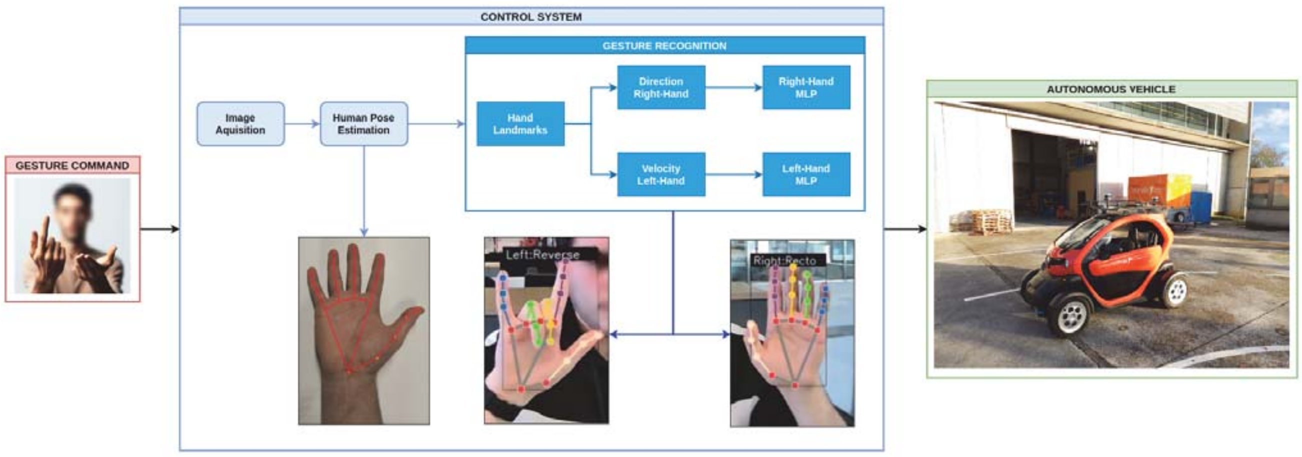


Fig. 1 Hands-off Parking Framework

These developments represent the current state of the art in parking control systems. However, they share a common issue: the operator is required to remain inside the car throughout the entire parking process. To address this challenge, this project considers two main concepts: teleoperation, which involves operating a vehicle from the outside, and the interpretation of orders through artificial vision.

A. *Vehicle Teleoperation*

Automated vehicle control without human presence is an active research area that is developing rapidly, especially in the field of Automated vehicles and remote driving [6]. In this area, there is a distinction between teleoperation from a remote location and teleoperation in the vehicle's proximity.

1) Remote Control: In the realm of controlling vehicles remotely, the most notably advancing solution, particularly in military applications, is the adoption of digital twins and Virtual Reality (VR) [7].

A virtual system and a physical system cooperate to form the control system. A virtual reality system uses a Head-Mounted Display (HMD) to show the user a digital twin of a vehicle and the environment in which it is located. The user operates the vehicle with a set of virtual reality controls. Control data are immediately translated from VR to the actual vehicle in real-time. This system makes ultra-remote control possible and facilitates control of multiple vehicles [8].

VR and digital twins for controlling vehicles remotely are undoubtedly a substantial progression. Nevertheless, our control system improves specific facets over this solution. Firstly, the cost of VR systems can be prohibitive for many users. Although prices are decreasing as the technology becomes more common, it remains a barrier to widespread adoption. Furthermore, accessibility is another issue. Not all users can comfortably use VR controls or head-mounted displays. This can be especially problematic for individuals with specific disabilities or health conditions.

2) On-site Control: In the area of on-site remote control, solutions have been widely available for a long time. This can be seen in the trucking industry, where some cranes are remotely controlled using handsets. However, a new remote-control system that allows the truck itself to be operated via a radio frequency-connected remote control was introduced by Volvo in 2018 [9]. This system was specifically designed to enable simple maneuvering in the absence of external assistance at low speeds (below 10 km/h).

Our plan is to implement the suggested control system (Fig. 1) by replacing the current control method. Specifically, we would use gesture detection instead of a remote control. Using gestures as a control method can provide numerous advantages over traditional remote control, particularly in the remote vehicle operation context:

- Accessibility: Gesture-based control systems increase accessibility for individuals with disabilities or those who have difficulty using traditional remote controls. This inclusivity is crucial for ensuring a broader range of users can operate and interact with the vehicle.
- Reduced Equipment: Conventional remote controls generally need physical devices like handheld controllers, which have a tendency to be lost or damaged easily. In contrast, gesture-based controls require no supplementary equipment, thereby mitigating potential issues stemming from damaged control devices.
- Simplicity: The control process can be simplified through the use of gestures. Users can convey their intentions through simple, natural movements that are often easier than manipulating buttons or joysticks on a remote control.

B. *Gesture Detection with Computer Vision*

Gesture detection via computer vision is a rapidly developing research area with a broad range of applications in human-computer interaction (HCI). Amongst the applications of hand gesture recognition systems, they are notable in user interfaces for sign language communication, automotive systems, and air gesture classifiers in new smartphones [10].

There is plenty of variety of methods for gesture detection, but the most important ones are stated here:

- OpenPose [11], [12] is a real-time library for human pose detection. It was the first to demonstrate the ability to jointly detect human body, foot, hand, and facial keypoints in individual images. OpenPose is capable of detecting a total of 135 key points and has been shown to be robust for varying numbers of people.
- MediaPipe [13] is a library that facilitates the integration of advanced machine learning features into your applications by using minimal code. MediaPipe offers essential machine learning models for standard duties, such as tracking hands, removing the development bottleneck that plagues many machine learning applications. These models and their user-friendly APIs simplify the development process and accelerate project completion for computer vision applications.

A recent study [14] compared various head pose estimation algorithms that were designed to capture facial geometry from videos. The study analyzed the performance of OpenFace 2.0 [15], MediaPipe [13], and 3DDFA_V2 [16]. The results revealed that 3DDFA_V2 had a mean error of less than or equal to 5.6°, depending on the plane of motion, while OpenFace 2.0 and MediaPipe had mean errors of 14.1° and 11.0°, respectively. This study demonstrated the superiority of the 3DDFA_V2 algorithm in head pose estimation, across different directions of motion.

Overall, OpenPose has been proven effective in detecting human poses, although its high computational demand may limit its use. In contrast, MediaPipe provides a more flexible alternative that may be better suited for real-time applications. Ultimately, the choice of method was made based on the requirements of this project, in order to make it low-cost, lightweight and scalable.

III. Theoretical Background

This section discusses the theoretical foundations of Human Pose Estimation, Mediapipe, and MultiLayer-Perceptron, which are essential for comprehending this project.

A. Human Pose Estimation

In order to determine both the configuration and location of the user's body, mainly their hands position, we are employing a technique known as Human Pose Estimation (HPE), which involves identifying and categorizing key points of the human body, such as joints (e.g., arms, head, torso). These key points collectively describe the individual's pose and are connected to form pairs.

The pairing of these key points is significant, and not all points can form a pair. The goal of HPE is to create a skeletal model of the human body, which can be applied to specific tasks. This skeletal model is represented by a choice of three main procedures: the skeleton-based model, the contour-based model, and the volume-based model, as illustrated in Fig. 2.

To determine the user's body configuration and location, specifically the positions of their hands, we utilize HPE. This approach involves identifying and categorizing crucial points on the human body, including joints like arms, head, and torso, which describe a person's pose. By connecting key points, pairs are formed. It is significant to note that not all points can be paired. HPE endeavors to generate a skeletal replica of the human body that can be utilized for specific purposes, presenting three primary methods: the skeleton-oriented model, the contour-oriented model, and the volume-oriented model.

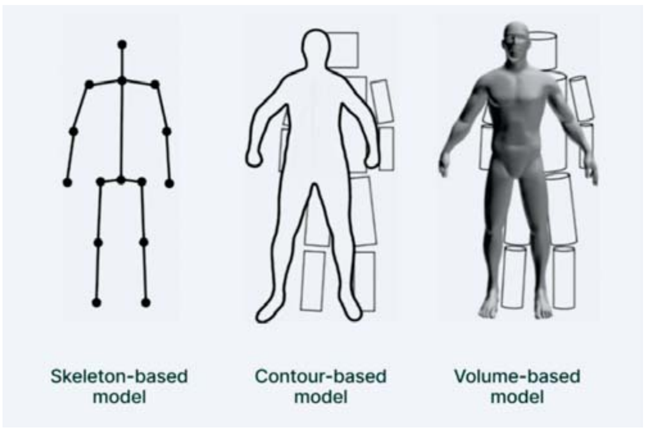


Fig. 2 Human Body Detection models

HPE operates within computer vision, striving to understand the geometric and motion characteristics of the human body. Two methodologies are employed: the classical approach and the deep learning-based one. In this project, we exclusively implement the latter, harnessing the capabilities of deep learning. Deep learning, exemplified by Convolutional Neural Networks (CNNs) [17], excels in computer vision tasks, including HPE. This project leverages deep learning and CNNs to robustly capture and interpret body poses, providing a more precise and versatile solution compared to traditional methods.

CNNs are capable of effectively extracting patterns and features from input images, making them extremely valuable for tasks such as classification [18], object detection [19], and image segmentation [20].

The HPE method uses CNNs to estimate body joint positions, treating it as a regression problem. A cascade of regressors improves our neural network's structure, greatly enhancing accuracy. This advanced architecture can model complex data, capturing both simple and nuanced poses often seen in real situations. This capability, previously hard to achieve, broadens its applications from human-computer interaction to robotics and so on.

B. MediaPipe

MediaPipe [13] is a cross-platform framework that enables the development of multimodal applications utilizing computer vision, machine learning, and media processing. Developers can utilize MediaPipe to construct pipelines of components capable of handling various types of data, including images, video, audio, and sensor data. MediaPipe offers a range of pre-designed models and tools for commonly used tasks, including detecting faces and hands, identifying objects, and segmenting images [21].

The theoretical foundation of MediaPipe [22] can be categorized into three key aspects: graph-based architecture, stream-based processing, and model customization.

- In MediaPipe, the protobuf (.pbtxt) text file establishes a graph structure. The graph's structure consists of nodes and edges, representing different elements in an assembly line. Nodes fall into three categories: calculators, subgraphs, and input/output nodes. Calculators perform operations on input data and produce output data. Subgraphs are reusable components made up of other nodes. Input/output nodes establish connections between the graph and external data sources or destinations. The graph definition outlines each node's name, type, options, and the connections between nodes using stream tags.
- Stream-Based Processing: MediaPipe utilizes streams to transmit data between nodes in a graph. A stream is a sequence of packets that transport data of a specific type, including images, tensors, landmarks, or detections. Each packet is timestamped to indicate when the data were either produced or consumed. MediaPipe provides various stream types, such as immediate streams, throttled streams, back-edge streams, and side-packet streams. Each stream type exhibits unique properties and behaviors that impact the flow of data in the graph.
- Model customization: MediaPipe offers multiple models for everyday tasks, including detecting objects, creating a face mesh, estimating poses, and segmenting images. MediaPipe Model Maker also customizes models with their data to detect objects or segments that are not covered by the provided models [23]. Model Maker is a tool that employs transfer learning to retrain existing models with a smaller dataset, achieving high performance. The customized models are compatible with MediaPipe and can be utilized in the graph definition.

C. *MultiLayer-Perceptron*

By definition, a MultiLayer Perceptron (MLP) is an artificial neural network that comprises numerous layers of interconnected neurons. These neurons are designed to mirror the structure of human brain neurons, allowing them to learn complex information and make meaningful predictions [24].

MLPs are feed-forward networks that exclusively transmit data in one direction. In contrast to recurrent neural networks, which allow data to flow in both directions to form a cycle. Despite this, MLPs still serve as the fundamental algorithm for potent neural networks like CNNs. CNNs are a specialized type of neural network designed for processing grid-like data, such as images. They have proven to be highly effective in tasks like image classification, object detection, and image segmentation [25].

The process of a MLP operates as follows. Each neuron within the MLP is solely capable of solving simple problems. However, when these neurons collaborate, they can solve intricate problems. Input data progress through various layers of interconnected neurons, with each layer solving a particular facet of the issue until the final output is generated, representing the resolution to the intricate problem.

An artificial neural network known as the MLP is structured as a series of interconnected layers of neurons, consisting of three primary components: the input layer, the hidden layers, and the output layer.

The input layer processes the raw input data, while the hidden layers perform computations based on this data. Finally, the output layer generates the desired output.

It is important to emphasize that the number of neurons in the input layer should align with the dimensions of the training instances, and the neuron count in the output layer should match the size of the output labels. The design of the hidden layer, in terms of the number of neurons and layers, can be tailored to meet specific requirements. The addition of more neurons in the hidden layer enhances the network's capacity to tackle progressively intricate problem-solving tasks.

IV. Proposal

During a time when advanced technology and innovation are revolutionizing the field of automated vehicles control systems, we propose a comprehensive project consisting of five critical components: Camera Selection, MediaPipe, Dataset, Multilayer Perceptron, and Integration.

A. *Camera Selection*

We chose a basic off-the-shelf webcam for this project with the primary objective of achieving a low-cost system. A comparison between industrial A9120 Bassler camera and a built-in webcam was done, taking as conclusion the fact that the differences in image quality and frame rate between the industrial A9120 Bassler camera and the built-in webcam were not significant enough to justify the substantial cost difference. The Bassler camera, while offering advanced features and slightly better performance, did not provide a proportionate increase in system efficiency for our specific application. Furthermore, by selecting the off-the-shelf webcam, we ensured broader accessibility and replicability of our project, as potential users or researchers might not have the budget or inclination to invest in high-end industrial cameras. This decision, therefore, not only met our cost-saving objective but also prioritized the wider applicability and adaptability of our system. In light of these findings, the basic webcam was deemed sufficient and more economical for our project's requirements.

B. *MediaPipe*

For this application we have employed the MediaPipe API for a specific purpose, which involves the holistic analysis of video input using the framework's comprehensive approach. This method simplifies tracking a single operator in the video stream. Our focus is on extracting hand landmarks and distinguishing between the left and right hands. This segmentation permits precise and detailed analysis of hand gestures, a pivotal component of our research, that enables us to deepen our comprehension of the subtle interactions and expressions conveyed through hand gestures. From the MediaPipe API, we obtained the 21 landmark points (x, y) for each hand, which we used for the control system.

C. Dataset

During the dataset generation process, the system's potential confusion was always taken into account in recognizing the intended gestures. Thus, separate datasets were created for each hand to minimize confusion. This dataset is composed of a first column signaling the class to which the stored gesture belongs, and then 21 landmarks with their *x* and *y* coordinates, for a total of 42 points. Before storing the hand landmarks, we normalized the coordinates to reduce the impact of operator-to-camera distance variations ensuring the system's robustness and reliability in various real-world conditions.

The steering of the car is controlled by the right hand. This dataset is composed by a total of 5603 hand gestures which classes for it are shown in Table I: Emergency Stop (1996), Move Forward (1522), Turn Right (1094), Turn Left (1021).

For the left hand, which is responsible for the car's velocity, the dataset is composed by a total of 6895 hand gestures which classes are displayed in Table II: Emergency Stop (1966), Emergency Stop (1522), Throttle Level 1 (381), Throttle Level 2 (288), Throttle Level 3 (340), Brake Level 1 (551), Brake Level 2 (352), Brake Level 3 (395), Reverse (1100).

The datasets are divided into training and testing sets for the subsequent use of the network, with 75% of the data assigned for training and 25% for testing. This partitioning strategy is used to evaluate the performance of the models and ensure their capability of making precise predictions when exposed to unfamiliar data.

The custom, minimal datasets utilized in this project were developed to function as evidence of feasibility for our particular application. It should be noted that these datasets were solely created to showcase the system's capabilities and are not intended for commercial use.

TABLE I
RIGHT HAND

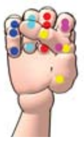
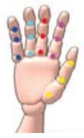
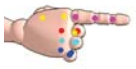
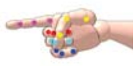
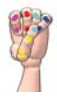
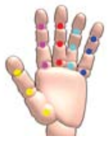
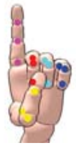
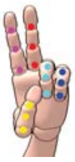
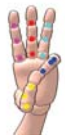
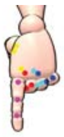
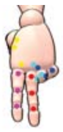
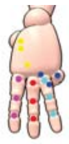
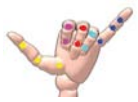
Gesture	Command	Legend
	Emergency Stop	R1
	Move forward	R2
	Turn right	R3
	Turn left	R4

TABLE II
LEFT HAND

Gesture	Command	Legend
	Emergency Stop	L1
	Emergency Stop	L2
	Throttle Level 1	L3
	Throttle Level 2	L4
	Throttle Level 3	L5
	Brake Level 1	L6
	Brake Level 2	L7
	Brake Level 3	L8
	Reverse	L9

D. MLP

The computational core of this project is formed by MLP models, which provides manners to interpret and respond to the hand gestures captured by the selected webcam. MLP models are configured for each individual hand. Figs. 3 and 4 get the 21 (x, y) landmarks associated with the hand gesture of the operator and takes into consideration the unique features and nuances associated with each command. Subsequently, the output of these MLPs represents the operator's interpreted command.

The MLP in our project is trained using labeled datasets. Of the 43 variables in the dataset, the initial column functions as the label signifying the class or category to which each

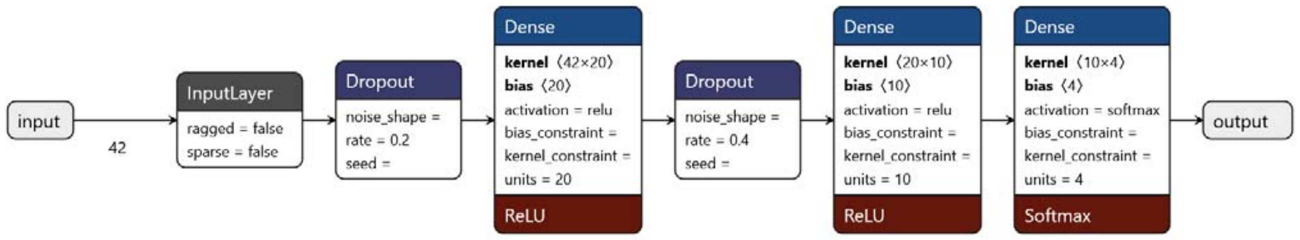


Fig. 3 MLP Steering

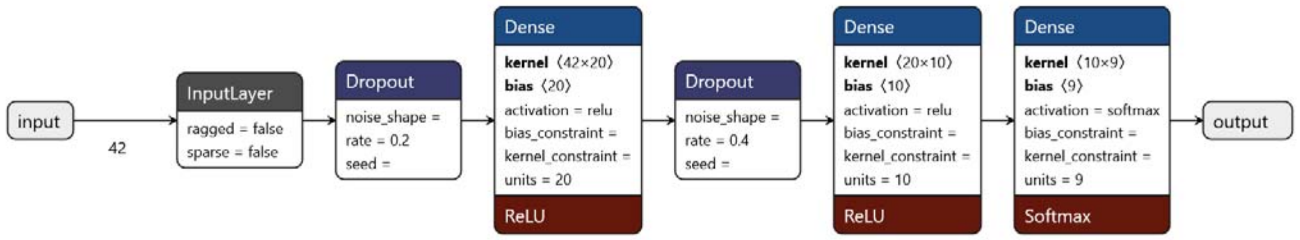


Fig. 4 MLP Velocity

data point pertains. The MLP learns to predict based on the remaining 42 parameters by adjusting its internal weights and biases. Throughout the training process, the model optimizes its parameters to minimize the variance between its predicted outputs and the actual labels provided in the dataset. This supervised learning approach allows the MLP to learn intricate connections among the input features and their corresponding labels, empowering it to deliver precise predictions on novel, unlabeled data points.

E. Integration

In the project's final phase, a critical objective is to seamlessly integrate the gesture recognition system into the distributed control architecture, in the same OBU (On-board Processing Unit) using ROS2 (Robot Operating System 2) [26]. To ensure a smooth integration, a containerization approach has been employed for each component. Firstly, the encapsulation of different modules within this project guarantees that software requirements for gesture recognition system are isolated from the host environment, preventing potential conflicts. Secondly, it simplifies the deployment and management of components across different nodes within the ROS2 ecosystem. This containerized procedure not only unlocks portability but also enables efficient scaling and easy distribution of the system, eventually providing a smooth integration into the ROS2 architecture.

Continuing with the integration process, we establish a dedicated ROS2 *node* for the control system to seamlessly communicate with the car's *node*, which is responsible for overseeing the actuator controllers. The gesture control system publishes the corresponding commands on specific ROS2 *topics*, gathering a range of essential functions, including steering, throttle, brake, or gear shifts, amongst many others.

Expanding on the previously established communication infrastructure, our integrated system enables not only the real-time and context-aware control of the vehicle via recognized gestures, but also advances its abilities by subscribing to the '*speed*' topic. This speed information is therefore utilized in the steering control algorithm, as shown by (1). Here, the variable 'v' represents the current velocity of the automobile. By integrating this dynamic variable into the control system, our system guarantees precise adjustment of the steering angle according to real-time speed data, as well as an accurate and safe process.

$$f(x) = (1,6/(e^{\frac{x}{v}} + 1)) + 0.1 \tag{1}$$

The function offers a steering angle ranging between 50% and 100% of the vehicle's maximum, up to 3 kilometers per hour. Thereafter, the angle progressively decreases to 10 km/h, at which it is practically minimum, set at 10% of the steering angle. Our distributed system facilitates interaction with the car's control system, converting identified gestures into tangible and responsive actions, thereby closing the gap between human input and vehicular control.

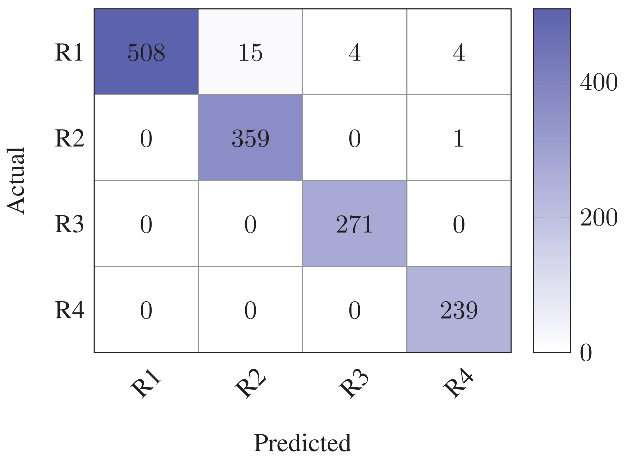
V. Results

For every MLP model, we implement a typical compilation setup that uses the '*Adam*' optimization algorithm [27], '*Sparse Categorical Crossentropy*' loss function [28], and '*accuracy*' as the performance metric.

The obtained values for each MLP are the following:

- Steering MLP - loss: 0.0747 - accuracy: 0.9829
- Velocity MLP - loss: 0.2979 - accuracy: 0.9345

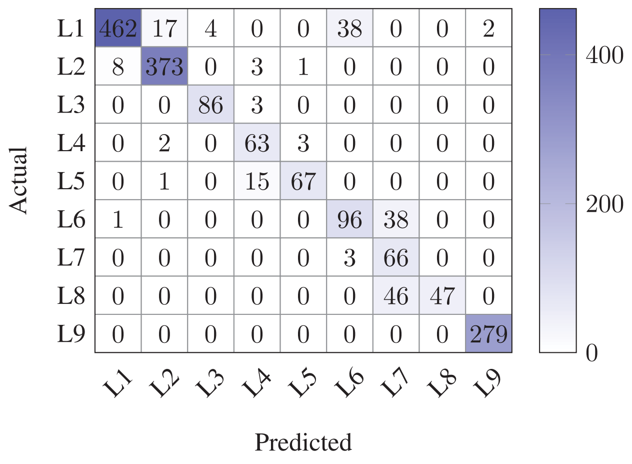
It is worth emphasizing that the outcomes of this study reflect a proof of concept rather than a refined final product. The main aim of this research was to demonstrate the possibility of utilizing hand signals to control a vehicle within a controlled environment. Despite displaying promising proficiency in detecting and responding to gestures, it is crucial to acknowledge that these findings serve as the



Actual \ Predicted	R1	R2	R3	R4
R1	508	15	4	4
R2	0	359	0	1
R3	0	0	271	0
R4	0	0	0	239

Actual
Predicted
400
200
0

Fig. 5 Steering Confusion Matrix



Actual \ Predicted	L1	L2	L3	L4	L5	L6	L7	L8	L9
L1	462	17	4	0	0	38	0	0	2
L2	8	373	0	3	1	0	0	0	0
L3	0	0	86	3	0	0	0	0	0
L4	0	2	0	63	3	0	0	0	0
L5	0	1	0	15	67	0	0	0	0
L6	1	0	0	0	0	96	38	0	0
L7	0	0	0	0	0	3	66	0	0
L8	0	0	0	0	0	0	46	47	0
L9	0	0	0	0	0	0	0	0	279

Actual
Predicted
400
200
0

Fig. 6 Velocity Confusion Matrix

preliminary foundation for further exploration of this idea. Further refinement, rigorous testing, and optimization are necessary to develop this proof of concept into a robust and production-ready solution. Our system's evaluation comprises two crucial phases: validation and testing.

A. Validation

During the validation phase, our analysis primarily focused on assessing the performance of the gesture detectors for each hand, aiming to gain a comprehensive understanding of their limitations and strengths. In this evaluation, we not only considered the extraction of confusion matrices, which provided valuable insights into the detectors' performance regarding different gestures, as depicted in Figs. 5 and 6, but there was a close examination of the network's accuracy and loss metrics. These values were instrumental in quantifying the system's ability to correctly classify and differentiate various hand gestures, highlighting where improvements or adjustments might be necessary.

Furthermore, to emulate real-world scenarios, we conducted a practical test involving the simulation of an environment with multiple individuals within the camera's frame. The primary objective was to assess the system's capability to maintain stability and avoid erratic switching between individuals while interpreting gestures. This real-world scenario testing not only provided insights into the system's adaptability but also allowed us to identify areas that may require subsequent refinement. The combination of performance metrics, confusion matrices, and real-world testing results offers a comprehensive overview of the system's functionality, paving the way for further testing and refinement stages.

B. Implementation

During the evaluation stage, our project went through two crucial assessments to gauge its practicality in the real world. Our initial evaluation consisted of a comprehensive simulation using the CARLA simulator [29]. The simulation encompassed a variety of parking situations that were resolved by different operators, enabling us to gather assorted feedback and insights into the system's performance in diverse scenarios. The simulated environment provided a controlled yet dynamic setting to evaluate the control system's responsiveness and adaptability.

After completing the simulation, we advanced to real-world testing by deploying our system on a modified Renault Twizy outfitted with automated capabilities. This progression to genuine on-road testing allowed us to verify the system's functionality amidst the practical challenges and complexities it could face. These testing phases yielded valuable information regarding the system's dependability, its capacity to manage various parking scenarios, and its potential for deployment in real-world applications.

VI. Conclusions

To sum up, while the development and testing of this project were noteworthy, there are aspects that merit critical attention. The objective to investigate the use of hand gestures for vehicle control was realized, yet there might be instances where hand gestures could be misinterpreted or not detected, leading to possible control issues. While the system does convert hand gestures to vehicle control actions, the real-world applicability and safety implications need rigorous and further scrutiny. Our tests, both in simulated and real-world environments, indicate an undeniable potential, but more extensive testing is essential. This work provides a robust foundation, with a perspective of continuous evaluation and refinement before considering wider applications in the foreseeable future.

VII. Acknowledgments

This work was partially funded by the Horizon Europe programme of the European Union, under grant agreement 101076868 (project AWARE2ALL). Views and opinions expressed here are however those of the author(s) only and do not necessarily reflect those of the European Union or CINEA. Neither the European Union nor the granting authority can be held responsible for them.

References

[1] B. Li and Z. Shao, "A unified motion planning method for parking an autonomous vehicle in the presence of irregularly placed obstacles," *Knowledge-Based Systems*, vol. 86, pp. 11–20, 2015.

[2] L. Hakamies-Blomqvist and B. Peters, "Recent european research on older drivers," *Accident Analysis Prevention*, vol. 32, no. 4, pp. 601–607, 2000.

[3] L. Hakamies-Blomqvist, A. Sirén, and R. Davidse, "Older drivers-a review," 2004.

[4] B. Li, T. Acarman, Y. Zhang, Y. Ouyang, C. Yaman, Q. Kong, X. Zhong, and X. Peng, "Optimization-based trajectory planning for autonomous parking with irregularly placed obstacles: A lightweight iterative framework," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11970–11981, 2021.

[5] Q. Zhang, H. Dong, and A. El Saddik, "Magnetic field control for haptic display: System design and simulation," *Ieee Access*, vol. 4, pp. 299–311, 2016.

[6] R. Lattarulo, J. Pérez, and J. Murgoitio, "Rrt trajectory planning approach for automated semi-trailer truck parking," in *2022 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, pp. 1–7, 2022.

[7] A. Rassõlkin, T. Vaimann, A. Kallaste, and V. Kuts, "Digital twin for propulsion drive of autonomous electric vehicle," in *2019 IEEE 60th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON)*, pp. 1–4, IEEE, 2019.

[8] H. Chen, F. Liu, Y. Yang, and W. Meng, "Multivr: Digital twin and virtual reality based system for multi-people remote control unmanned aerial vehicles," in *2022 17th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pp. 647–652, IEEE, 2022.

[9] V. T. Corporation, "External steering interface exster," 2018. https://stpi.it.volvo.com/STPIFiles/Volvo/FactSheet/EXSTER_Eng_01_309626349.pdf [Accessed: October 23rd, 2023].

[10] P. Shah, R. Shah, M. Shah, and K. Bhowmick, "Comparative analysis of hand gesture recognition techniques: A review," in *Advanced Computing Technologies and Applications* (H. Vasudevan, A. Michalas, N. Shekokar, and M. Narvekar, eds.), (Singapore), pp. 471–478, Springer Singapore, 2020.

[11] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," 2017.

[12] D. Osokin, "Real-time 2d multi-person pose estimation on cpu: Lightweight openpose," 2018.

[13] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "Mediapipe hands: On-device real-time hand tracking," *arXiv preprint arXiv:2006.10214*, 2020.

[14] Y. Hammadi, F. Grondin, F. Ferland, and K. Lebel, "Evaluation of various state of the art head pose estimation algorithms for clinical scenarios," *Sensors*, vol. 22, no. 18, 2022.

[15] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L.-P. Morency, "Openface 2.0: Facial behavior analysis toolkit," in *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*, pp. 59–66, IEEE, 2018.

[16] J. Guo, X. Zhu, Y. Yang, F. Yang, Z. Lei, and S. Z. Li, "Towards fast, accurate and stable 3d dense face alignment," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[17] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[19] J. Terven and D. Cordova-Esparza, "A comprehensive review of yolo: From yolov1 and beyond," 2023.

[20] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 263–272, 2017.

[21] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C. Chang, M. G. Yong, J. Lee, W. Chang, W. Hua, M. Georg, and M. Grundmann, "Mediapipe: A framework for building perception pipelines," *CoRR*, vol. abs/1906.08172, 2019.

[22] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann, "Mediapipe: A framework for building perception pipelines," 2019.

[23] S. Hussain, R. Saxena, X. Han, J. A. Khan, and H. Shin, "Hand gesture recognition using deep learning," in *2017 International SoC design conference (ISOCC)*, pp. 48–49, IEEE, 2017.

[24] H. Taud and J. Mas, "Multilayer perceptron (mlp)," *Geomatic approaches for modeling land change scenarios*, pp. 451–455, 2018.

[25] H. Taud and J. Mas, *Multilayer Perceptron (MLP)*, pp. 451–455. Cham: Springer International Publishing, 2018.

[26] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, may 2022.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[28] A. S. Foundation, "Multi hot sparse categorical cross entropy."

[29] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*, pp. 1–16, PMLR, 2017.