AV4EV - Open-source Autonomous Vehicle software for Open-standard Electric Vehicle platforms

Rahul Mangharam

University of Pennsylvania

FINAL REPORT

October 29, 2024

DISCLAIMER

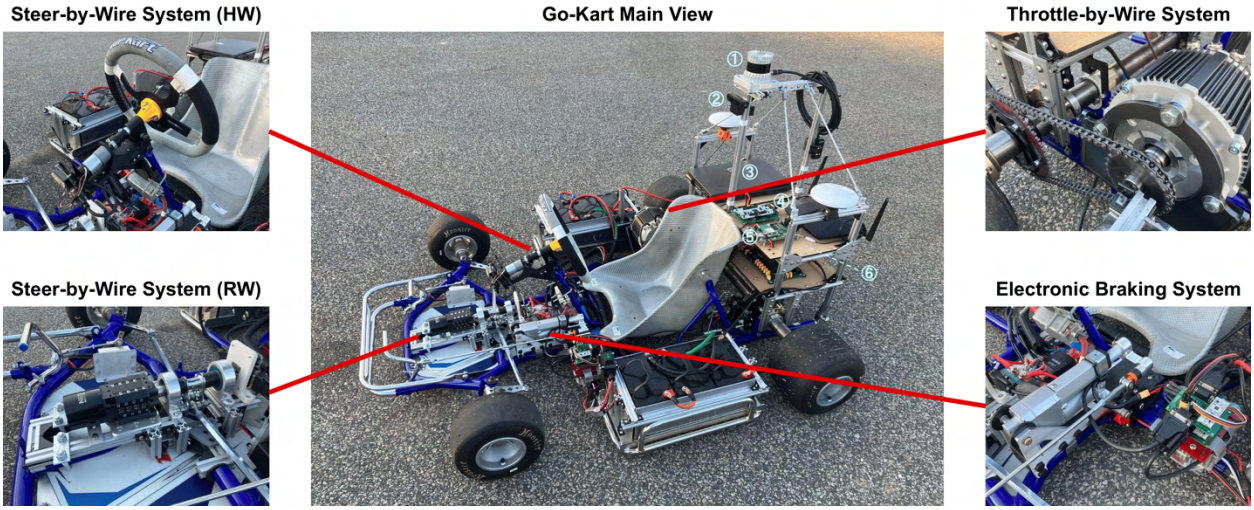
The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, under [grant number 69A3552344811] from the U.S. Department of Transportation's University Transportation Centers Program. The U.S. Government

assumes no liability for the contents or use thereof.

AV4EV - Open-Source Modular Autonomous Electric Vehicle Platform

1. Introduction

The AV4EV project aims to create an accessible, open-source, one-third scale electric vehicle (go-kart) platform designed for autonomous driving research and development. This platform merges the capabilities of full-scale vehicles with the flexibility and lower cost of smaller platforms. The objective is to address the challenges of developing autonomous vehicle (AV) systems and to make AV research more accessible to universities and research institutions. This project offers a modular, open-source design and supports multiple driving modes, including manual, autonomous, and teleoperated.



2. Problem Statement

Research on autonomous vehicles faces significant barriers due to the cost and complexity of full-sized vehicle platforms. Many existing AV platforms are expensive and require large, skilled teams to develop and test in specialized facilities. On the other end, scaled-down RC cars, while affordable, lack the capabilities needed for advanced research. AV4EV addresses this gap by offering an affordable, modular, open-source platform for AV research, enabling universities to develop and test algorithms without the limitations of reduced-size platforms.

3. Approach

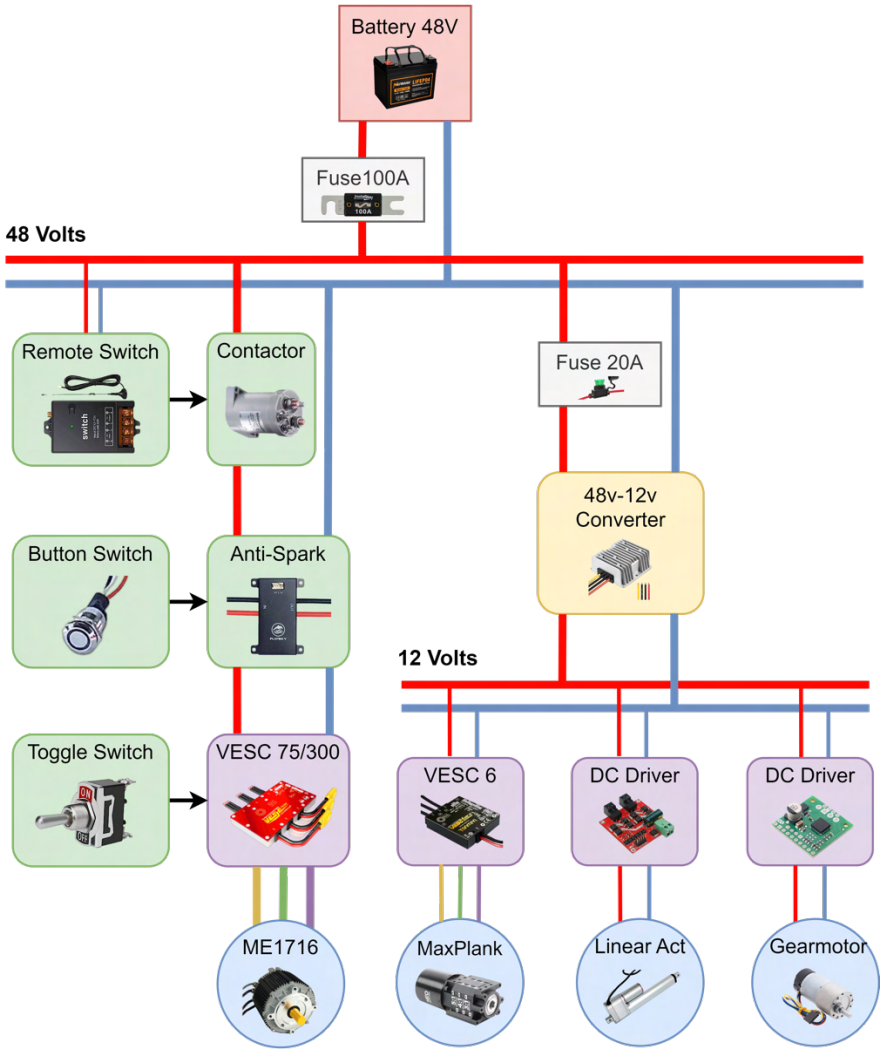
The AV4EV platform combines mechatronics, sensors, and autonomous driving software in a modular design. It uses a flexible sensing suite and open-source autonomous driving software that includes perception, localization, planning, and control algorithms. This allows researchers to develop and test new algorithms while avoiding the prohibitive costs of full-scale vehicle platforms.

The go-kart platform has been validated in competitive settings, including the 2023 Autonomous Karting Series Purdue Grand Prix, where it demonstrated its autonomous driving capabilities.

4. Methodology

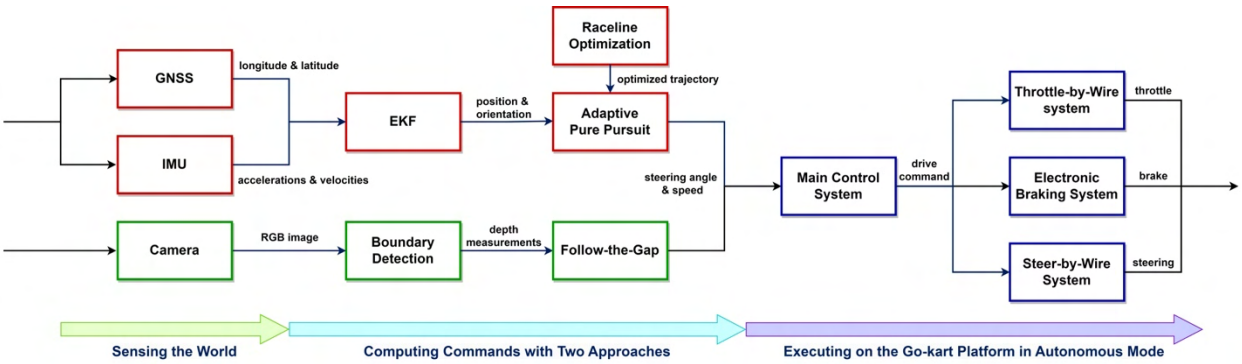
The AV4EV platform consists of the following components:

1. **Mechatronics:** The platform's mechatronic system includes a power distribution system, a main control system, and subsystems for steering, braking, and throttle control, all of which communicate via a controller area network (CAN).
2. **Sensing:** The platform uses a variety of sensors, including a LiDAR, an OAK-D camera, a Global Navigation Satellite System (GNSS), and an Inertial Measurement Unit (IMU). These provide real-time data for perception and localization.
3. **Software:** The autonomous driving software is built on Robot Operating System (ROS2) and includes algorithms for raceline optimization and adaptive pure pursuit control.



5. Findings

1. **System Performance:** The AV4EV platform successfully demonstrated its capabilities in autonomous driving scenarios, including perception, localization, and control. The system was tested in both indoor and outdoor environments, with results showing high performance in terms of speed, accuracy, and reliability.
2. **Accessibility:** By lowering the cost of AV research platforms, AV4EV has made it easier for universities and research groups to develop and test new algorithms in a real-world environment. The system's modularity also allows for customization and reusability across different applications.
3. **Education Impact:** The platform's design enables educational institutions to provide hands-on experience to students, helping them understand the challenges of developing autonomous systems while working with real hardware.



6. Conclusions

The AV4EV project successfully addresses the need for a flexible and affordable autonomous driving platform that can bridge the gap between small-scale and full-scale vehicles. The system's open-source nature and modular design make it ideal for research and educational purposes, providing a scalable solution for AV development.

7. Recommendations

1. **Expand Deployment:** Develop more use cases for the platform, including logistics, warehouse management, and urban transport, to further demonstrate its versatility.
2. **Increase Community Engagement:** Encourage more universities and research institutions to adopt the AV4EV platform and contribute to its development through the open-source community.
3. **Enhance Modularity:** Continue to develop additional modules for specialized use cases, such as autonomous delivery robots or urban mobility solutions.

8. Project Outputs and Documentation

1. **PDFs for any resulting publications:** https://ieeexplore.ieee.org/document/10588611
2. **Dataset URL(s) and Descriptive Metadata:** http://av4ev.org
3. **ORCIDs for Project Investigators:**
    1. Zhijie Qiao: https://orcid.org/0000-0001-9197-2849
    2. Rahul Mangharam: https://orcid.org/0000-0003-XXXX-XXXX

1. Report No. 443	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle AV4EV - Open-source Autonomous Vehicle software for Open-standard Electric Vehicle platforms		**5. Report Date** October 29,2024 **6. Performing Organization Code**
7. Author(s) Rahul Mangharam: https://orcid.org/0000-0003-2539-896X		**8. Performing Organization Report No.** **443**
9. Performing Organization Name and Address University of Pennsylvania 200 S 33rd St, Philadelphia PA 19104		**10. Work Unit No.** **11. Contract or Grant No.** Federal Grant No. 69A3552344811
12. Sponsoring Agency Name and Address Safety21 University Transportation Center Carnegie Mellon University 5000 Forbes Avenue Pittsburgh, PA 15213		**13. Type of Report and Period Covered** Final Report (July 1, 2023-June 30, 2024) **14. Sponsoring Agency Code** USDOT
15. Supplementary Notes Conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.		

16. Abstract

Over the past decade, self-driving capability for all variants of on-street vehicles have promised safer and more efficient transportation. This remains “work in progress” with large unfilled gaps in addressing user-acceptance, safety, ethics, regulation, technology and the business model. Our goal is to develop the Open-source Autonomous Vehicle (AV) software for Open-standard Electric Vehicle (EV) platforms, ie. AV4EV paradigm, to help realize safe, reliable, and efficient autonomy for off-street use cases. We focus on developing the AV4EV Autonomy Essentials Kit (AV4EV-Kit) for known controlled application domains: logistics (in-warehouse mobile robots), material handling (autonomous forklifts) and airside cargo (autonomous ground support equipment). The AV4EV business model addresses these many smaller domains through simplification and modularity. The EV ‘skateboard’ chassis is orders of magnitude simpler than on-street vehicles (~20 moving parts compared to nearly 2,000 in contemporary vehicle architectures) - supporting standardization of interfaces for autonomous driving. Modularity allows AV4EV to address autonomous vehicle market sizes of 50K-250K vehicles/year for each use case by enabling component re-use and efficient customizability to meet specific segment needs. If successful, the AV4EV Kit will create a new business category for Autonomy-as-a-Service with plug-n-play hardware and software for rapid prototyping and deployment. Autonomous machines have a serviceable market of $2.9B with a 15.5% growth rate.

The AV4EV Autonomy Essentials Kit enables logistics customers to kickstart their journey of autonomous machines for safe and efficient movement of people and goods, even if their companies have little prior autonomous system development experience. Using the AV4EV-Kit, customers can rapidly prototype EV platforms into autonomous machines in 10 days for brownfield deployments.

The AV4EV Autonomy Essentials Kit is dedicated to lowering the entry barrier of autonomous driving development and deployment. AV4EV-Kit consists of (1) a plug-in-play hardware platform with sensors and compute, (2) an autonomy software stack to achieve essential autonomous driving functions of perception, sensor fusion, mapping, localization, path planning, obstacle avoidance, traffic light recognition and safe control; and (3) a new Software Defined Vehicle approach for autonomous machine software development and testing in the cloud to lower cost of mixed-criticality software and over-the-air upgrades to enhance safety across the vehicle lifecycle and customize for different deployment scenarios. The AV4EV-Kit conforms to the open-source Autoware autonomous vehicle software standard to interface with the EV’s drive-by-wire system for users to easily integrate navigation functions with vehicle control. The AV4EV-Kit incorporates energy-efficient machine learning-based perception, planning and control algorithms developed by the PI’s and Co-PI’s labs and will be tested by commercialization partners on a variety of EV platforms.

17. Key Words	**18. Distribution Statement**
Autonomous vehicles, robotics, computational thinking, machine learning, control, simulation	No restrictions. This document is available through the National Technical Information Service, Springfield, VA 22161. Enter any other agency mandated distribution statements. Remove NTIS statement if it does not apply.

19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 24	22. Price Refers to the price of the report. Leave blank unless applicable.

AV4EV: Open-Source Modular Autonomous Electric Vehicle Platform for Making Mobility Research Accessible

Zhijie Qiao[1,2*], Mingyan Zhou[1*], Zhijun Zhuang[1], Tejas Agarwal[1,2], Felix Jahncke[1,3], Po-Jen Wang[2], Jason Friedman[1,2], Hongyi Lai[1,2], Divyanshu Sahu[1], Tomás Nagy[1,4], Martin Endler[1,4], Jason Schlessman[2,5], Rahul Mangharam[1,2]

[1]School of Engineering and Applied Science, University of Pennsylvania, Email: rahulm@seas.upenn.edu

[2]Autoware Foundation Center of Excellence for Autonomous Driving

[3]Professorship of Autonomous Vehicle Systems, Technical University of Munich

[4]Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague

[5]Red Hat Research

***Abstract*—When academic researchers develop and validate autonomous driving algorithms, there is a challenge in balancing high-performance capabilities with the cost and complexity of the vehicle platform. Much of today's research on autonomous vehicles (AV) is limited to experimentation on expensive commercial vehicles that require large skilled teams to retrofit the vehicles and test them in dedicated facilities. On the other hand, 1/10th-1/16th scaled-down vehicle platforms are more affordable but have limited similitude in performance and drivability. To address this issue, we present the design of a one-third-scale autonomous electric go-kart platform with open-source mechatronics design along with fully functional autonomous driving software. The platform's multi-modal driving system is capable of manual, autonomous, and teleoperation driving modes. It also features a flexible sensing suite for the algorithm deployment across perception, localization, planning, and control. This development serves as a bridge between full-scale vehicles and reduced-scale cars while accelerating cost-effective algorithmic advancements. Our experimental results demonstrate the AV4EV platform's capabilities and ease of use for developing new AV algorithms. All materials are available at AV4EV.org to stimulate collaborative efforts within the AV and electric vehicle (EV) communities.**

***Index Terms*—Autonomous vehicle, electrical vehicle, open-source design.**

I. INTRODUCTION

The increasing interest in self-driving cars has ushered in a new area of study in recent years: autonomous racing. This involves the development of software and hardware for high-performance racing vehicles intended to function autonomously at unprecedented levels, including high speeds, substantial accelerations, minimal response times, and within unpredictable, dynamic, and competitive settings [1]. However, a significant hurdle remains the unavailability of full-sized vehicles and the accessibility of smaller-scaled RC cars. For full-sized platforms that encompass independent driving capacities such as the Dallara AV21 from Indy Autonomous Challenge [2], testing the limits of safety and performance is costly and hazardous, and also outside the reach of most academic departments and research groups. For smaller-scaled RC cars such as F1TENTH [3], the limited capability of sensing and computing constrains the complexity of the algorithms and the level of research conducted.

To address this issue, we created AV4EV, an accessible, open-source reference model for a one-third-scale autonomous electric racing platform. This platform merges the capabilities of full-sized vehicles with the compactness and adaptability of its smaller size. AV4EV offers open-source designs for mechatronics, sensing, and autonomous driving software, aiming to provide a standardized solution for modular autonomous and electric vehicles.

Our go-kart won the championship at the 2023 Autonomous Karting Series Purdue Grand Prix, where it competed against several other US national teams [4]. This autonomous go-kart solution can easily be adopted by universities and research institutes to promote the safe and effective development and verification of AV.

This work makes the following contributions:

1) We introduced an accessible modular electric vehicle platform with multi-driving modes (manual, autonomous, and teleoperated), bridging the gap between full-scale vehicles and RC cars. The estimated cost of constructing one go-kart, including all mechatronic systems, stands at approximately 12,500 USD. It is expected that with scaled production, the cost will decrease substantially.
2) We developed a flexible sensing suite and demonstrative software solutions to handle autonomous driving capabilities validated through experiments. The estimated cost is around 11,000 USD, while the figure can vary depending on user-specific requirements and customization.
3) We provided comprehensive open-source resources to guide building and testing the one-third-scale electric go-kart with detailed tutorials, GitHub repositories for hardware design and software stacks, demonstration videos, a bill of materials [5]–[7].

*These authors contributed equally to this work.

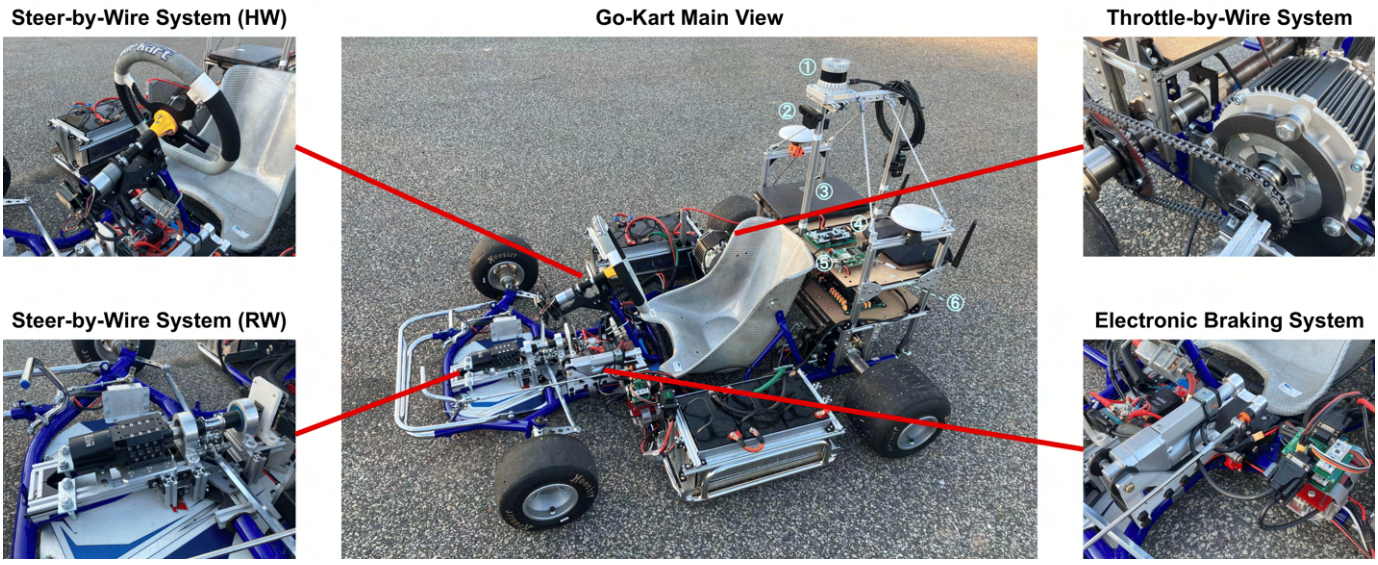


Fig. 1: Go-kart platform overview with Steer-by-Wire System (SBWS) including its hand wheel (HW) and road wheel (RW) components, Throttle-by-Wire System (TBWS), and Electronic Braking System (EBS). The sensors and computing units mounted on the double-deck rear shelf are enumerated from top to bottom as follows: (1) Ouster LiDAR, (2) OAK-D camera, (3) Onboard laptop, (4) Main Control System (MCS), (5) Sepentrio GNSS, and (6) IMU, concealed from the main view perspective, is positioned on the lower deck.

II. Mechatronics

The go-kart mechatronic system is designed as a modular system, consisting of several subsystems that are responsible for different vehicle execution tasks. There are five subsystems which integrated with the base go-kart chassis in a non-intrusive way: Power Distribution System (PDS), Main Control System (MCS), Throttle-by-Wire System (TBWS), Steer-by-Wire-System (SBWS), and Electronic Braking System (EBS) (Fig. 1). All subsystems except the PDS utilize an STM32 Nucleo development board on a standalone PCB as the electronic control unit (ECU). Communication among these modular systems is achieved through the controller area network (CAN), aligning with modern vehicle design standards for efficient information exchange.

A. Power Distribution System (PDS)

The autonomous go-kart is powered by six Nermak Lithium LiFePO4 deep cycle batteries, each possessing a voltage of 12V and a capacity of 50Ah. These batteries are installed on both sides of the go-kart and interconnected via wiring across the chassis. Four of them are linked in a series, yielding a net voltage of 48V, which powers the TBWS motor. A step-down converter is utilized to convert the voltage from 48V to 12V, which in turn provides power to the SBWS and EBS motor. The remaining two batteries, also interconnected in series, produce a net voltage of 24V. This voltage is then fed through several converters to obtain different desired voltages to power up the sensing (Fig. 2a) and control (Fig. 2b) systems.

B. Main Control System (MCS)

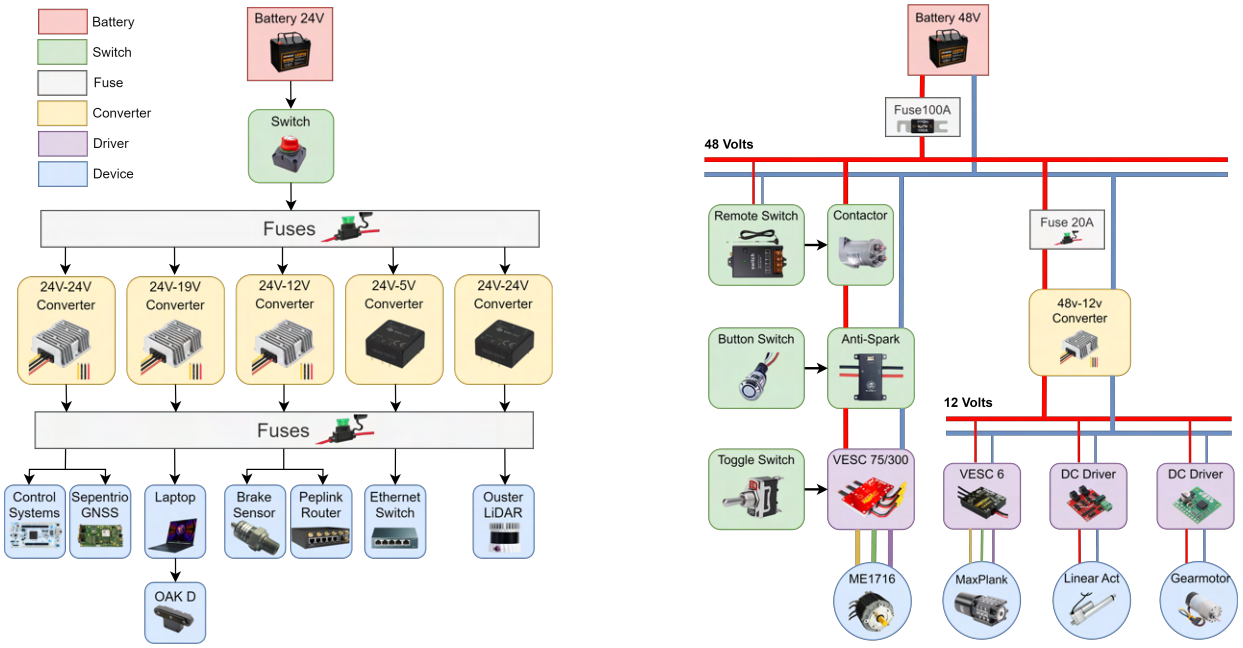
The MCS handles all driving requests from the top-level supervisory controller and dispatches commands (throttle, steering, brake) on the CAN bus [8]. It serves as an interface between the go-kart mechatronic system and the end user. Three different operation modes are supported: manual, remote, and autonomous. In manual mode, input is read from the steering wheel, throttle, and brake pedals of a driver, just like in a conventional vehicle. In remote mode, the operator uses a Spektrum DX6 2.4GHz radio to send driving commands to the MCS. In autonomous mode, the command is transmitted from a high-level computing unit, such as a laptop or an onboard computer, through USB-to-TTL communication. After receiving the desired driving commands, the MCS sends these on the CAN bus to be received by the subsystems. Meanwhile, each subsystem measures its state with sensors and sends feedback on the CAN bus. This feedback is gathered by the MCS and shared with the operator.

C. Throttle-by-Wire System (TBWS)

The TBWS includes the electronic controller unit (ECU) and VESC 75/300 motor driver to control the go-kart's main drive motor. The brushless DC motor (ME1717 from Motenergy) transmits the motion to the go-kart rear axle of through a chain and drives the wheels in the longitudinal direction. The ECU receives the desired speed from the MCS via the CAN bus, measures the current speed through an encoder, and outputs the desired throttle signal to the VESC controller, which then powers up the motor. Additionally, a remote kill switch is added independent of the ECU that allows the user to kill power, thus ensuring safety in the worst-case scenario.

D. Steer-by-Wire System (SBWS)

The SBWS eliminates the mechanical steering shaft between the hand wheel (HW) and road wheel (RW), allowing



(a) Sensing (b) Motor

Fig. 2: Sensing (left) and motor (right) power system with connections and devices.

each part to be governed by its motor, sensor, and ECU [9]. This design reduces weight, space, and cost with the modular structure, while improving the flexibility and availability of autonomous driving functions [10]. Our HW component utilizes a brushed DC motor to coaxially drive the HW. The RW component employs a NEO1650 Brushless DC motor to propel the two front wheels via steering tie rods as linkages.

E. Electronic Braking System

The original go-kart design translates movement from the driver pressing the brake pedal to the master cylinder and reservoir via the push rod, generating hydraulic braking pressure without the need for additional servo motors. To achieve autonomous braking without human input, a linear actuator is mounted at the end of the push rod to create a linear movement simulating the pedal-pressing action. This non-intrusive design allows the safety operator (if present) to press the brake pedal regardless of the linear actuator state. Finally, a pressure sensor is installed onto the braking hydraulic system to collect data for effective feedback control.

III. Sensing

The sensing system is a fundamental module for research and development for perception and localization. Our design employs a flexible sensor setup that can be customized and reconfigured to suit different objectives and priorities.

To start up, an Ouster OS1 LiDAR is positioned at the highest point on the rear end of the go-kart to leverage its max 200-meter range and 360-degree field of view. The OAK-D camera, placed below the LiDAR, has the capabilities of high-resolution image capturing, depth measuring, and long-range object tracking. These features work seamlessly with the LiDAR point cloud for object fusion and post-processing.

Moreover, the go-kart is equipped with a Global Navigation Satellite System (GNSS) and an Inertial Measurement Unit (IMU). For GNSS, we utilized the Sepentrio Mosaic-H carrier board with two Multiband antennas (IP66) from ArduSimple mounted on both rear sides of the go-kart. We also subscribed to Swift Navigation's real-time kinematic positioning (RTK) service, enabling our GNSS to achieve centimeter-level position accuracy. In situations where GNSS signals are disrupted due to severe weather or signal obstructions, an IMU is needed for localization filtering. Thus, we placed a BNO055 9-DOF IMU on the go-kart's centerline of mass to provide accurate accelerometer, gyroscope, and magnetometer information.

All sensors transmit data to an onboard laptop, which then executes algorithms and transmits drive commands to the MCS. We used the MSI Pulse GL66 15.6" Gaming Laptop, integrated with an Intel Core i7-12700H, an RTX3070 GPU, 16GB of internal RAM, and a storage capacity of 512GB. The laptop also contains three USB 3.0 ports and one Ethernet port to support high-speed data transmission with the sensors.

IV. Software

We designed an autonomous racing framework using the Robot Operating System (ROS2) for our go-kart platform with the free tooling of Python and C++. This framework incorporates two primary algorithms: a GNSS-based pure pursuit method for pre-mapped racing and a camera-based follow-the-gap (FTG) algorithm [11] for reactive racing. In the framework outlined in this paper, LiDAR was not used due to its complexity. Nevertheless, the LiDAR data is readily available and will be integrated into future research. The software pipeline within the holistic autonomous driving workflow is illustrated in Fig. 3.

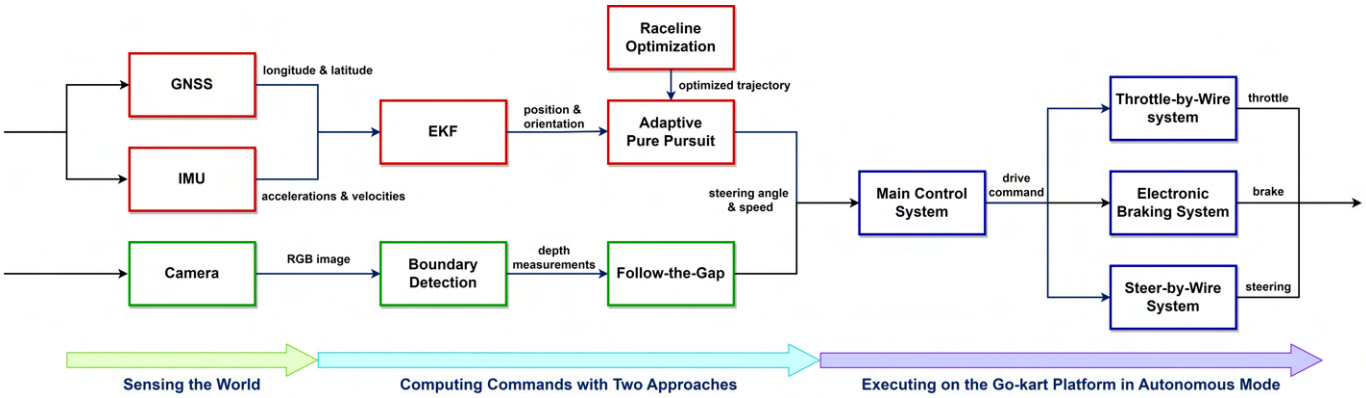


Fig. 3: Software pipeline for go-kart autonomous driving capabilities: GNSS-based adaptive pure pursuit (red), camera-based follow-the-gap (green), go-kart mechatronics execution (blue).

A. Localization

The position measurements from the GNSS are presented in latitude and longitude. To convert these geographical coordinates into a more interpretable format within a local frame, we utilized the equirectangular projection method as in equations (1):

$$x = r \cdot \cos(lat) \cdot lon, \;\; y = r \cdot lat, \tag{1}$$

where r symbolizes the mean radius of the Earth, which is 6371 kilometers, lat stands for latitude (radians), and lon denotes longitude (radians). A reference point is first established, and all subsequent coordinates are defined with respect to this reference point, treating it as the origin [12]. While this approach has the potential to introduce distortion, in our case, the impact is negligible due to the small size of the testing field.

As previously mentioned, there are instances where the GNSS signal may experience interruptions. To guarantee timely and accurate localization information, we implemented an Extended Kalman Filter (EKF) that integrates IMU data. Evolving dynamically over time t, the velocity motion model X_t adopted for the go-kart is consisting of the position x_t, y_t and orientation ψ_t; the input to the system is linear velocity v_t and angular velocity ω_t; and the identity covariance matrix P_t that is initialized at timestamp zero:

$$X_t = [x_t, y_t, \psi_t]^T, \tag{2}$$

$$u_t = [v_t, \omega_t], \tag{3}$$

$$P_t = \begin{matrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\psi} \\ \sigma_{yx} & \sigma_y^2 & \sigma_{y\psi} \\ \sigma_{\psi x} & \sigma_{\psi y} & \sigma_\psi^2 \end{matrix}. \tag{4}$$

At timestamp t, the system is linearized around the current state, and the prediction step is executed as follows:

$$X_{t+1|t} = \begin{matrix} x_t + v_t \Delta t \cdot cos(\psi_t) \\ y_t + v_t \Delta t \cdot sin(\psi_t) \\ \psi_t + \Delta t \omega_t \end{matrix}. \tag{5}$$

For each state in the system, we calculated the partial derivatives with respect to the other states to obtain the Jacobian matrix:

$$J = \begin{matrix} 1 & 0 & -\Delta t \cdot v \cdot \sin(\psi) \\ 0 & 1 & \Delta t \cdot v \cdot \cos(\psi) \\ 0 & 0 & 1 \end{matrix}. \tag{6}$$

The prediction update of the covariance matrix is as follows:

$$P_{t+1|t} = JPJ^T + R, \tag{7}$$

where the dynamic noise R is approximated as a constant diagonal matrix of 0.1, with units in meters and radians.

For the observation step, we extracted position data x and y from the GNSS and orientation data ψ from the IMU, and denote them with subscripts:

$$X_{obs} = [x_{obs}, y_{obs}, \psi_{obs}]^T. \tag{8}$$

Given that the observation directly corresponds to the state, the Jacobian is equivalent to the identity matrix. By combining the variance readings from the sensors that are organized as a diagonal matrix M, we could calculate the Kalman gain K:

$$M = \begin{matrix} \sigma_{xobs}^2 & 0 & 0 \\ 0 & \sigma_{yobs}^2 & 0 \\ 0 & 0 & \sigma_{\psi obs}^2 \end{matrix}, \tag{9}$$

$$K = P_{t+1|t} I^T (I P_{t+1|t} I^T + M)^{-1}, \tag{10}$$

Finalize the update step to complete localization:

$$X_{t+1|t+1} = X_{t+1|t} + K(X_{t+1|t} - X_{obs}), \tag{11}$$

$$P_{t+1|t+1} = (I - KI) P_{t+1|t}. \tag{12}$$

B. Raceline Optimization

In pre-mapped racing scenarios, a reference racing line is generally acquired in advance and subsequently tracked by the controller. The raceline is represented by a sequence of waypoints consisting of the target position x, y, velocity v, etc. While manually piloting the go-kart, waypoints are gathered at consistent temporal or spatial intervals. It may not account for vehicle dynamics, resulting in a non-smooth trajectory.

Therefore, we integrated a min-curvature raceline optimization algorithm as proposed in [13]. First, we calibrated the physical properties of the go-kart such as mass, width, maxi-

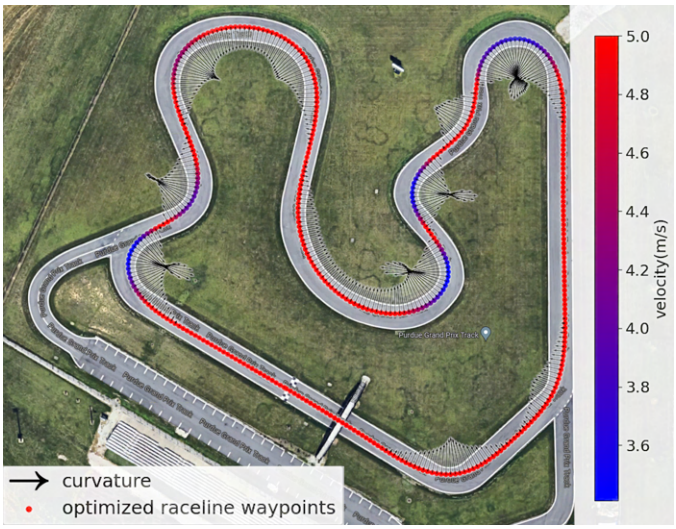


Fig. 4: Waypoints collection and raceline optimization at Purdue Grand Prix racing track, which spans a distance of 434 meters.

mum turning radius, maximum acceleration, etc., assuming a track of uniform width, and then utilized manually collected waypoints to depict the centerline of the track. The raceline points can be parameterized as:

$$\vec{r}_i = \vec{p}_i + \alpha_i \vec{n}_i, \tag{13}$$

where $\vec{p}_i = [x_i, y_i]^T$ is the center line point, $\vec{n}_i$ is the unit length normal vector, and a_i encodes the track boundaries. The raceline is then defined through third-order spline interpolations of the points r_i in x and y coordinates. Followed the formulation in [13], we minimized the discrete squared curvature γ_i of the splines along the raceline:

$$\underset{[\alpha_1 \cdots \alpha_N]}{\text{minimize}} \quad \sum_{i=1}^{N} \gamma_i^2(t) \tag{14}$$

$$\text{subject to} \quad \alpha_i \in [\alpha_{i,\min}, \alpha_{i,\max}] \quad \forall 1 \leq i \leq N. \tag{15}$$

Subsequently, we generated the velocity profile considering the longitudinal and lateral acceleration limits of the car at various velocities. As depicted in Fig. 4, the optimized raceline shows reduced curvature, thereby enhancing smoothness and eliminating overlapping waypoints.

C. Adaptive Pure Pursuit Controller

To track the generated raceline, we implemented an adaptive pure-pursuit controller based on the geometric bicycle model [14]. Initially, a lookahead point is chosen on the raceline, situated at a fixed lookahead distance L from the vehicle. L is adaptively interpolated between a minimum $L_{min} = 2$m and a maximum $L_{max} = 5$m, proportionally scaled to the vehicle's current velocity v and regulated by the maximum velocity $v_{max} = 5$m/s:

$$L = L_{min} + \frac{v}{v_{max}}(L_{max} - L_{min}). \tag{16}$$

The lookahead point comprises both the desired velocity and position. Intuitively, for the vehicle to trace the arc from its current position to the lookahead point, the steering angle should be proportional to the arc curvature γ. Utilizing geometric relationships, we deduced the radius r of the arc and subsequently determine γ:

$$\gamma = \frac{1}{r} = \frac{2|y|}{L^2}, \tag{17}$$

where $|y|$ is the lateral distance from the vehicle to the lookahead point. To actuate the steering angle and enhance stability, we utilized a Proportional-Derivative (PD) controller that modulates the steering angle δ_t according to γ:

$$\delta_t = K_p \gamma_t + K_d \frac{d\gamma_t}{dt}, \tag{18}$$

where γ is treated as the cross-track error term [15], reflecting the lateral deviation. In practice, $K_p = 2.0$, $K_d = 1.0$.

D. Boundary Detection

We devised a vision-based algorithm for detecting race track boundaries for the reactive component of the AKS competition, where pre-mapping was not permitted. The algorithm relies on grass detection surrounding the race track, employing classical computer vision techniques with OpenCV.

To identify grass regions, the input RGB camera image is blurred using a Gaussian filter to eliminate unwanted noise. Its blue and green channels are then extracted and normalized in grayscale, which grants higher intensities to green pixels than to pixels of other colors. Green pixels $G(x, y)$ are identified by the green g and the blue b channel with a threshold τ, where τ can be affected by many factors such as the environment and the lighting condition:

$$G(x,y) = \begin{array}{ll} 1, & \text{if } 0.6 \cdot g - b \geq \tau \\ 0, & \text{if } 0.6 \cdot g - b < \tau \end{array}. \tag{19}$$

The resultant binary image G represents a mask for grass regions. This mask is then processed with open and then close morphology operations to remove small noise.

Next, we conducted a bird's-eye view (BEV) projection that converts an image from a front view to a top-down view. A transformation matrix is determined offline by mapping four points in the image to their respective BEV coordinates using OpenCV's getPerspectiveTransform function (Fig 5).

The final step is to convert the grass BEV into a 2D depth format. The depth data is denoted by a vector $s \in \mathbb{R}^d$, where each s_i is a distance measurement from the go-kart to an object. Correspondingly, a vector $a \in \mathbb{R}^d$ captures the angles associated with s_i. The range of detection is $[-\pi/2, \pi/2]$, indicating a 180-degree field of view ahead of the vehicle sampled at 0.5° resolution. The zero angle is aligned with the vehicle's heading while angles are measured counter-clockwise.

E. Follow-the-Gap

After acquiring depth data from boundary detection, we employed the FTG method to identify the largest gap that

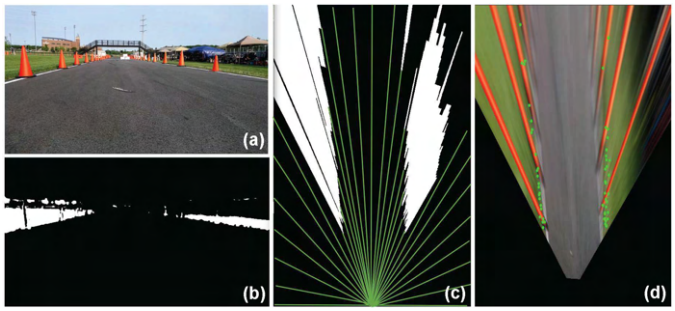


Fig. 5: Grass boundary detection. (a) Raw camera input. (b) Filtered grass mask. (c) The BEV of the grass mask. Green lines indicate the angles for searching grass distances. (d) The converted depth data is plotted as green dots and overlaid onto the BEV image of the camera input.

meets the required safety distance from the vehicle and navigate toward it. First, we defined a gap g as a continuous subsequence $[s_i, s_j]$ where i and j are the starting and ending indices respectively, such that:

$$\forall k \in [i, j],\ s_k \geq \epsilon, \tag{20}$$

where $\epsilon = 2.5$m is a safety distance threshold that determines the minimum allowable distance for a gap. We chose the largest gap as the optimal one, which starts at index i_{opt} and ends at index j_{opt}. Then, we chose the midpoint of the optimal gap as the goal at index k_{mid} to reduce unnecessary oscillation:

$$k_{mid} = \frac{i_{opt} + j_{opt}}{2}. \tag{21}$$

Since the zero angle is parallel to the vehicle's heading, we calculated the steering angle δ from the angle vector a:

$$\delta = a_{k_{mid}}. \tag{22}$$

Thereafter, the desired velocity v is interpolated between a minimum $v_{min} = 2$m/s and a maximum $v_{max} = 5$m/s, proportionally scaled to the vehicle's current steering angle δ, and regulated by the maximum allowable steering angle $\delta_{max} = 1.0$rad:

$$v = v_{min} + \frac{\delta}{\delta_{max}}(v_{max} - v_{min}). \tag{23}$$

V. Conclusion

In this paper, we introduced an open-source design for an electric go-kart platform enabling advanced research and development in autonomous driving systems. The design's modular mechatronic systems seamlessly support different driving modes. Additionally, we have implemented an adaptable sensor stack to execute tasks such as perception, localization, planning, and control. Our experimentation has showcased the go-kart's versatility, demonstrating its proficiency in the autonomous mode while running the pure pursuit and follow-the-gap algorithms. This innovative design effectively bridges the gap between reduced-scale cars and full-scale vehicles, enabling both widespread accessibility with high performance. It consequently provides immense value to universities and research institutions, fostering collaboration towards the open development and validation of autonomous vehicles.

Future work will focus on the continuous improvement of the mechatronic, sensing, and software systems. We plan to leverage the platform's different driving modes and explore human-machine interactions, such as the imitation learning algorithm [16], which involves dynamic cooperative control between the driver and the vehicle.

Acknowledgements

The authors wish to express their gratitude to the Autoware Foundation for providing financial support for the project. Our thanks are also extended to Dr. Jack Silberman and his team from the University of California, San Diego for their contributions to the autonomous go-kart community. Lastly, we would like to thank Andrew Goeden for his efforts in organizing the 2023 AKS Purdue Grand Prix, providing an outstanding and enriching experience.

References

[1] J. Betz, A. Wischnewski, A. Heilmeier, F. Nobis, T. Stahl, L. Hermansdorfer, B. Lohmann, and M. Lienkamp, "What can we learn from autonomous level-5 motorsport?" in *9th International Munich Chassis Symposium 2018*, P. Pfeffer, Ed. Wiesbaden: Springer Fachmedien Wiesbaden, 2019, pp. 123–146.

[2] "Indy autonomous challenge," accessed: 2023-01-10. [Online]. Available: https://www.indyautonomouschallenge.com

[3] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam, "F1tenth: An open-source evaluation environment for continuous control and reinforcement learning," *Proceedings of Machine Learning Research*, vol. 123, 2020.

[4] Autonomous karting series official website. [Online]. Available: https://autonomouskartingseries.com/

[5] University of pennsylvania go-kart documentation. [Online]. Available: https://go-kart-upenn.readthedocs.io/en/latest/index.html

[6] University of pennsylvania go-kart mechatronics github repository. [Online]. Available: https://github.com/mlab-upenn/gokart-mechatronics

[7] University of pennsylvania go-kart sensor github repository. [Online]. Available: https://github.com/mlab-upenn/gokart-sensor

[8] L. Ran, W. Junfeng, W. Haiying, and L. Gechen, "Design method of can bus network communication structure for electric vehicle," in *International Forum on Strategic Technology 2010*, 2010, pp. 326–329.

[9] T. Kaufmann, S. Millsap, B. Murray, and J. Petrowski, "Development experience with steer-by-wire," *SAE transactions*, pp. 583–590, 2001.

[10] S. A. Mortazavizadeh, A. Ghaderi, M. Ebrahimi, and M. Hajian, "Recent developments in the vehicle steer-by-wire system," *IEEE Transactions on Transportation Electrification*, vol. 6, no. 3, pp. 1226–1235, 2020.

[11] V. Sezer and M. Gokasan, "A novel obstacle avoidance algorithm: "follow the gap method"," *Robotics and Autonomous Systems*, vol. 60, no. 9, pp. 1123–1134, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889012000838

[12] W. contributors, "Equirectangular projection," https://en.wikipedia.org/wiki/Equirectangular_projection, 2021, accessed: 2023-03-01.

[13] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, and B. Lohmann, "Minimum curvature trajectory planning and control for an autonomous race car," *Vehicle System Dynamics*, vol. 58, no. 10, pp. 1497–1527, 2020. [Online]. Available: https://doi.org/10.1080/00423114.2019.1631455

[14] V. Sukhil and M. Behl, "Adaptive lookahead pure-pursuit for autonomous racing," 2021.

[15] J. M. Snider *et al.*, "Automatic steering methods for autonomous automobile path tracking," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08*, 2009.

[16] X. Sun, M. Zhou, Z. Zhuang, S. Yang, J. Betz, and R. Mangharam, "A benchmark comparison of imitation learning-based control policies for autonomous racing," 2022.