engineering proceedings

MDPI

Proceeding Paper

Lessons Learned from an Autonomous Race Car Competition †

Zalán Demeter *, Máté Hell and Gergely Hajgató

HUMDA Lab Nonprofit Ltd., Széchenyi István University, H1113 Budapest, Hungary; mate.hell@humda.hu (M.H.); gergely.hajgato@humda.hu (G.H.)

* Correspondence: zalan.demeter@humda.hu

† Presented at the Sustainable Mobility and Transportation Symposium 2024, Győr, Hungary, 14–16 October 2024.

Abstract: The advancement of AI technologies and the increasing processing power of computers have made high-speed autonomous racing possible. Different leagues, such as the Abu Dhabi Autonomous Racing League (A2RL) and the Indy Autonomous Challenge (IAC), are organizing races in simulation and with real race cars. In this paper we will describe our experience with the inaugural A2RL event and a SIM race organized by IAC. With respect to A2RL, we will give an overview of the physical parameters of the race car, the sensors we worked with, and our software solution including how we created trajectories for different test scenarios.

Keywords: simulation; autonomous driving; race car driving; autonomous racing; autonomous vehicles; software development

Citation: Demeter, Z.; Hell, M.; Hajgató, G. Lessons Learned from an Autonomous Race Car Competition. *Eng. Proc.* **2024**, *79*, 25. https://doi.org/10.3390/engproc2024079025

Academic Editors: András Lajos Nagy, Boglárka Eisinger Balassa, László Lendvai and Szabolcs Kocsis-Szürke

Published: 5 November 2024

1. Introduction

While significant effort is taken to reduce the local emissions of passenger cars as an individual, the sheer number of such vehicles is expected to grow in the forthcoming years [1]. As car manufacturers are coming closer to the physical limits of pollution reduction, new approaches are coming into the foreground of research to make mobility sustainable. Autonomous operation of vehicles is one of the most promising approaches that can mitigate the environmental effects of passenger vehicles both on the individual (e.g., automated eco-driving mode [2]) and on the swarm (e.g., platooning [3]) level.

Autonomous driving systems (ADSs) are still in the focus of research as more complex tasks can be solved with more accessible computational resources and with the advances in machine learning. Novel ideas cannot be tested on public roads, though, leaving real-life experiments to closed areas like proving grounds and racetracks. While the former are suitable to test planned scenarios, autonomous racing involves ad hoc situations that can be challenging for ADSs in unforeseen ways. Hence, autonomous racing is the test environment of the cutting-edge solutions in autonomous driving.

The spectrum of autonomous racing is wide, spanning from racing with toy cars (e.g., F1TENTH [4]) to racing with full-fledged formula cars (e.g., IAC [5] and A2RL [6]). As only a handful of teams are involved in the latter series, information on series-specific challenges is somewhat limited.

The present paper summarizes the experiences gained by the team of H UMDA Lab during the IAC simulation challenge and during the inaugural A2RL race. As a team participating formerly in the F1TENTH series, we believe, that sharing these insights contributes to the field of autonomous racing by highlighting the difficulties—with possible solutions—that arise when starting to experiment with a real-sized autonomous racecar.

2. IAC Simulation Race Event

The IAC Simulation Race Event was a simulation-only competition which included three challenges, a time trial, a race against an AI-driven opponent, and a similar race but with noisy GPS signals.

2.1. Simulation Environment

The simulation environment was developed by Autonoma Labs and is built on the AWSIM environment. It included the physical simulation of the vehicle platform (Dallara AV-21R) used in IAC [7]. As the race progressed, necessary features such as multi-vehicle support and GPS dropouts were added for the different rounds. The simulation of the sensors was limited, not including any perception sensors, and the speed, position and opponent vehicle position data could be considered ground truth. It provided two interfaces to control the car, a keyboard-based human control interface and an autonomous interface accessible via the robot operating system (ROS).

2.2. Software Pipeline

The software pipeline needed to be extensive, as it had to be able to handle different challenges in different rounds. The task was to complete the fastest possible lap on a racetrack and to be able to cope with different multi-vehicle environments. To accomplish this, the pipeline needed to be able to follow, approach, and overtake cars in front of it. To achieve this, such a software architecture was used that is well known in the field of robotics and autonomous driving, and in which the design of the necessary autonomous functions is based on a three-way partitioning of perception–planning–control.

The architecture of the software pipeline and its connection to the simulation environment is depicted in Figure 1. The perception group processes the simulated sensor data and other ground truth measurements. The only module in this group, state estimation, is intended to solve GPS dropout compensation, as near-ground-truth position data could be obtained from the simulation environment. The detection and tracking of opponent vehicles are also provided by the simulation.

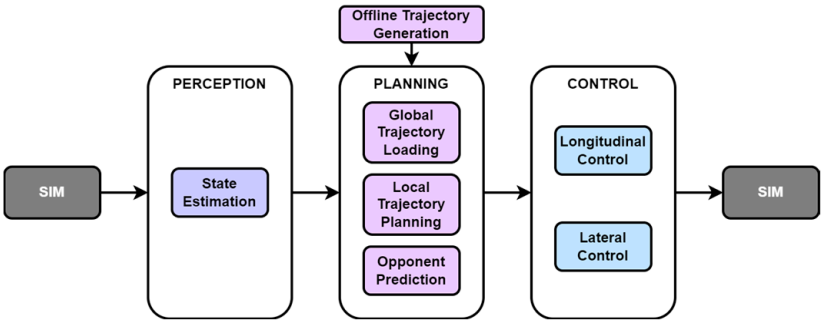

Figure 1. The software solution used by HUMDA Lab for the IAC simulation race follows a pipeline structure based on the classical perception–planning–control partitioning.

The corrected position is the most important input to the planner group. The planner's task is to design a feasible and optimal reference for the control algorithm based on the vehicle's position. Three submodules were distinguished, the first being a global trajectory planning submodule that is required to complete the timed laps. To achieve this, we use offline trajectory generation and loading within the pipeline. The other two submodules are required to meet multi-vehicle challenges. Predicting the movement of opponent vehicles is necessary to plan the right action at any given time. The task of local trajectory planning is to devise a different trajectory from the global one to overtake the opponent when the right opportunity arises. To implement these functions, we use a graph-based dynamic local path planning approach [8].

A simple but robust geometric controller based on the Hoffmann–Stanley algorithm was chosen to implement the lateral control of the race car [9]. Due to the nature of the simulation environment, the vehicle dynamics are simplified significantly compared to the real behavior, so many complex effects are negligible. As a result, the stability of such a geometric controller can also be comparable to more complex predictive or feedback controllers. The controller consists of four factors: an orientation error term, a position

error term, a yaw compensation term, and a steering delay compensation term. The sum of the above two error terms and two compensation terms gives the target steering angle at a given time instant. For the longitudinal control we used a simple PID regulator, which receives the target velocity from the planner and regulates the current speed by the pedal positions. These are complemented by the anti-lock braking system (ABS) and traction control (TC) to improve the stability of the car. The resulting calculated control signals are directly connected to the vehicle input interface provided by the simulation environment.

We propose this solution based on extensive testing of our autonomous system developed for F1Tenth competitions [10]. During the development, we evaluated various software solutions from other teams, including the race-winning architecture following the sense–think–act paradigm presented by ForzaETH and other reactive methods presented for solving obstacle avoidance problems [11,12]. Solving the competition tasks required a modification of the planner module as described above, but the previously used controller proved to be stable in this simulated environment.

3. A2RL Race Event

The Abu Dhabi Autonomous Racing League organized its first race event in April 2024. Teams could access the cars and the corresponding simulator two months prior to the race. There were three trials during the event: a time trial where one car was on the tracks, a head-to-head race with two cars where teams had to showcase their ability to overtake opponents, and finally a multi-car race during the finale.

3.1. Vehicle Platform

The vehicle platform used in the competition is built on a chassis codenamed EAV24, which is based on the Dallara SF23 used in the 2023 Super Formula season. One of the interesting features of the platform is that it allows brake-based torque vectoring, as it has a brake actuator system that can control four brake circuits separately.

The computer system consists of three main units in addition to low-level controllers and network devices. The high-performance computer (HPC) for implementing autonomous functions is a rugged Neousys RGS-8805GC model. It is configured with a powerful processor and graphics card and has a wide range of input and output interface connectors, as well as an easily expandable storage module. In addition, there is a body system unit (BSU) which connects the high- and low-level computer systems. This is an embedded, hard real-time unit collecting data from the chassis sensors and providing them to the HPC and realizes a well-defined interface to the actuators. The third unit is a logger module, whose task is to collect and send vehicle data for race control in real time.

The layout of the sensors enabling the implementation of the autonomous functions is shown in Figure 2. To enable multi-vehicle racing, it is necessary to use many different types of perception sensors. The car has three Seyond (Innovusion) Falcon Kinetic FK1 LiDAR sensors, providing almost 360 degrees of coverage. These are particularly important for accurate distance estimation and for certain mapping algorithms. In addition, there are four ZF ProWave RaDAR sensors, positioned on four sides of the car. These sensors are essential for the detection and tracking of various opponent vehicles, as they also provide speed information for the measurement points. In addition, there are seven Sony IMX728 cameras on the car, which can be used to implement surround-view vision, which also helps in detecting and tracking objects. The Vectornav VN-310 module is a combined GNSS and IMU (inertial measurement unit) sensor that is particularly useful for positioning and estimation tasks. There is also a Kistler OSS (optical speed sensor) on the car, which provides accurate velocity measurement in longitudinal and lateral directions and features an integrated IMU.

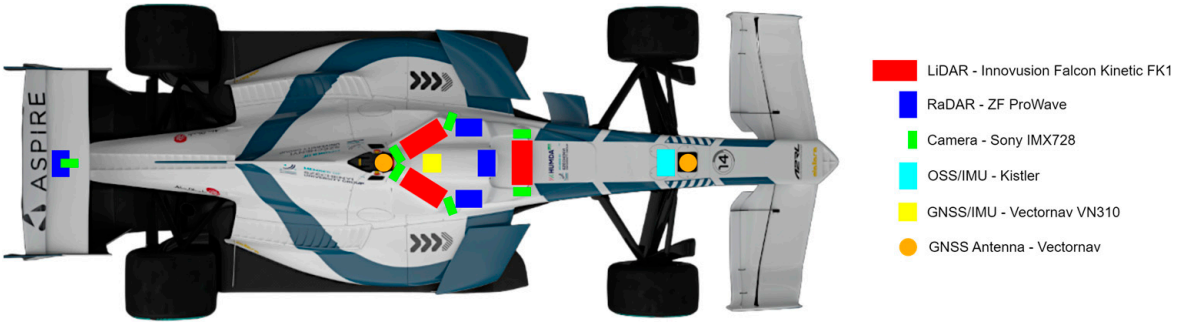

Figure 2. The vehicle platform used by the A2RL competition. The most important sensors needed to implement autonomous functions are shown, excluding those used in classical motor sports.

3.2. Software Pipeline

As with the solution used in the IAC simulation race, the software pipeline used in the A2RL event also uses the perception–planning–control partitioning. It has a more extensive feature set than the former, but from an architectural point of view they share many similarities with each other. The role of the perception group is prominent, as the localization and detection of opponent vehicles is not a trivial problem in a real environment.

As shown in Figure 3, the perception group has two submodules, of which state estimation takes the most emphasis. Among the sensors of the vehicle platform presented above, the localization is implemented by the fusion of GNSS, IMU, and OSS sensors using an EKF (extended Kalman filter)-based algorithm. The motion model is based on a simple kinematic rigid-body model with the yaw rate from the IMU and the velocity in the longitudinal and lateral directions from the OSS as input. To implement the opponent detection, an algorithm based purely on LiDAR measurements is used, which consists of point cloud segmentation, down-sampling, and ROI (region of interest) filtering.

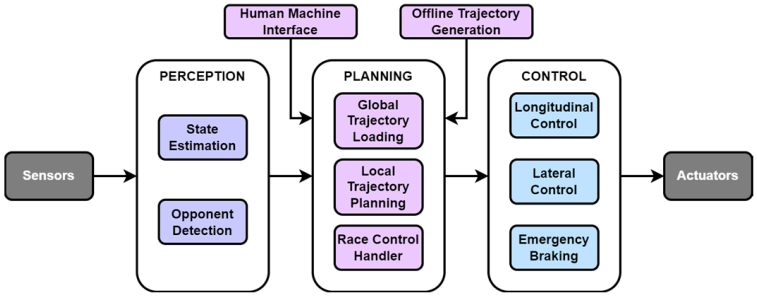

Figure 3. The software solution used by HUMDA Lab for the A2RL competition follows a pipeline structure based on the classical perception–planning–control three-way partitioning.

This is followed by the planning group in the pipeline, where an offline global trajectory loader module initializes the planner. These global trajectories serve as the reference position and speed for the timed laps. The offline trajectory generation process is described in detail in Section 3.3. For solving multi-vehicle challenges, a split-lanes method is implemented, wherein three global trajectories are loaded: an ideal line, a line to the left of the track, and a line to the right of the track. The task of the local trajectory planner is to plan trajectories between these predefined global options. This is achieved by using twice continuously differentiable splines, thus ensuring a continuous curvature of the moving path. In addition, the velocity profile of the transfer path is continuously matched between the start and end points, while maintaining the appropriate physical constraints. In addition to these matters, one of the important responsibilities of the planner is to comply with the rules required by race control. These include stopping the race car safely under certain conditions, respecting speed limits, and changing the operating states of the car. To

specify different behaviors, a human–machine interface with a graphical user interface is also developed to give the system adequate supervision.

For the lateral control a steering controller using both feedback and feedforward mechanisms is used to ensure the stability of the vehicle even at the limits of handling and to minimize the lateral deviation from the reference path [13]. This geometric controller is robust to sudden increases in error terms and parameter-invariant over a reasonable range. The longitudinal control is realized by a PI controller based on a force requirement, handling engine braking and the proper calculation of the acceleration and braking forces.

Publications from teams experienced in IAC competitions played a major role in the selection of software solutions. The supervisory architecture used by TII EuroRacing and the perception–planning–control partitioning architecture developed by TUM Autonomous Motorsport were used as a baseline [14,15]. However, instead of the complex predictive approaches presented by them, we opted for simpler, robust solutions to facilitate our introduction to the competition series.

3.3. Trajectory Generation and Validation

As some of the tasks depend on different trajectories—such as driving out from the pit or stopping on the grid—the car is the closest to its physical limits while driving on the ideal line—or so-called race line—with the highest speed possible. Hence, the race line trajectory has a distinguished role among the others, and the corresponding speed profile is an integral part of it.

Candidates for race lines lie on the Pareto front of trajectories considering their length and the average velocity achievable on them, making the race line generation process a multi-objective optimization. Achievable velocities are approximated with the curvature of the trajectory in the optimization, as it is in direct correlation with the largest feasible speed and can be calculated without modelling the dynamics of the vehicle. As the curvature has a greater effect on the lap times, the length of the race line can be left out from the optimization. Such a generation process is called minimum-curvature trajectory planning, and it has proved to be robust in real-life racing conditions [16]. In contrast, obtaining an optimal target speed profile depends on a calibrated vehicle model, which was available to the teams of A2RL in a closed-source simulator only, making the usability of the model limited for optimization.

To overcome this shortage, both the trajectory and the speed profile generation were left to a human driver, who drove multiple laps in the simulator. The recorded laps were analyzed by a race engineer, and the valuable parts were fitted together to form a full lap. The merging technique was the same as for the split-lanes method: the curves were joined by a cubic spline with C^2 continuity, and the speed profiles were joined with a smooth linear transition. The length of the joining sections was a parameter, and the fine-tuning of the speed profile was also possible. The final trajectories were then re-evaluated in the simulator to reveal those areas where the car cannot follow the trajectory due to the limitations of the controller.

4. Software Scaling

Testing full-scale autonomous race cars can be expensive due to high component costs and frequent software failures. Therefore, we believe that initial real-world testing of software should be conducted on small-scale platforms and scaled up gradually. This brings several challenges, which we have also faced in the transition from the F1TENTH platform. The main challenge lies in the different hardware sets provided by the vehicle platform. In addition, different race cars have different dynamic properties and therefore require distinct software solutions to be competitive. In our architecture, we use a thin hardware layer and easily interchangeable modular components that allow multiple pipelines to run in parallel. This enables us to develop algorithms for different platforms in an easily adaptable uniform system [17].

5. Conclusions

This paper described two distinct autonomous race car competitions, the software pipelines we used during the events, our solution to generate trajectories for different race scenarios, and our experience in tackling the challenges encountered. In the IAC Simulation Race Event participants had to tackle three challenges in a simulator, and in the A2RL eight teams had to compete with real race cars. While a simulator with high-fidelity physics was provided by the IAC to conduct the races, some major environmental effects such as wind, weather-dependent track temperature, etc., were not considered in the simulation, and all the sensor signals were free from noise except the GPS in a confined segment of the track. In contrast, our team faced highly varying track temperature, wind gusts, and noisy sensor signals at the A2RL competition. Despite the differing environments, our team achieved fourth place in the IAC simulator challenge and fifth place at the inaugural A2RL event, proving the robustness of the applied software solution.

Author Contributions: Conceptualization, Z.D. and G.H.; methodology, Z.D. and G.H.; software, Z.D.; validation, M.H., G.H. and Z.D.; formal analysis, M.H.; investigation, M.H. and Z.D.; resources, Z.D. and G.H.; data curation, M.H.; writing—original draft preparation, Z.D., M.H. and G.H.; writing—review and editing, Z.D.; visualization, Z.D.; supervision, Z.D.; project administration, G.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors and the affiliated institution declare that there are no conflicts of interest related to this publication.

References

1. Statista Passenger Cars-Worldwide. Available online: https://www.statista.com/outlook/mmo/passenger-cars/worldwide#unit-sales (accessed on 3 July 2024).
2. Wadud, Z.; MacKenzie, D.; Leiby, P. Help or Hindrance? The Travel, Energy and Carbon Impacts of Highly Automated Vehicles. *Transp. Res. Part A Policy Pract.* **2016**, *86*, 1–18. [CrossRef]
3. Mersky, A.C.; Samaras, C. Fuel Economy Testing of Autonomous Vehicles. *Transp. Res. Part C Emerg. Technol.* **2016**, *65*, 31–48. [CrossRef]
4. O'Kelly, M.; Sukhil, V.; Abbas, H.; Harkins, J.; Kao, C.; Pant, Y.V.; Mangharam, R.; Agarwal, D.; Behl, M.; Burgio, P. F1/10: An Open-Source Autonomous Cyber-Physical Platform. *arXiv* **2019**, arXiv:1901.08567.
5. Indy Autonomous Challenge. Available online: https://www.indyautonomouschallenge.com/ (accessed on 1 February 2024).
6. ASPIRE Abu Dhabi Autonomous Racing League in UAE. Available online: https://a2rl.io/ (accessed on 1 February 2024).
7. Autonoma Inc. AWSIM Racing Simulator. Available online: https://github.com/autonomalabs/AWSIM (accessed on 3 July 2024).
8. Betz, J.; Wischnewski, A.; Heilmeier, A.; Nobis, F.; Hermansdorfer, L.; Stahl, T.; Herrmann, T.; Lienkamp, M. A Software Architecture for the Dynamic Path Planning of an Autonomous Racecar at the Limits of Handling. In Proceedings of the 2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE), Graz, Austria, 4–8 November 2019.
9. Hoffmann, G.M.; Tomlin, C.J.; Montemerlo, M.; Thrun, S. Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing. In Proceedings of the 2007 American Control Conference, New York, NY, USA, 9–13 July 2007; pp. 2296–2301.
10. Fazekas, M.; Demeter, Z.; Tóth, J.; Bogár-Németh, Á.; Bári, G. Evaluation of Local Planner-Based Stanley Control in Autonomous RC Car Racing Series. In Proceedings of the 2024 IEEE Intelligent Vehicles Symposium (IV), Jeju Island, Republic of Korea, 2–5 June 2024. [CrossRef]
11. Baumann, N.; Ghignone, E.; Kühne, J.; Bastuck, N.; Becker, J.; Imholz, N.; Kränzlin, T.; Lim, T.Y.; Lötscher, M.; Schwarzenbach, L.; et al. ForzaETH Race Stack—Scaled Autonomous Head-to-Head Racing on Fully Commercial Off-the-Shelf Hardware. *J. Field Robot.* **2024**. [CrossRef]
12. Sezer, V.; Gokasan, M. A Novel Obstacle Avoidance Algorithm: "Follow the Gap Method". *Robot. Auton. Syst.* **2012**, *60*, 1123–1134. [CrossRef]

13. Kapania, N.R.; Gerdes, J.C. Design of a Feedback-Feedforward Steering Controller for Accurate Path Tracking and Stability at the Limits of Handling. *Vehicle Syst. Dyn.* **2015**, *53*, 1687–1704. [CrossRef]
14. Raji, A.; Caporale, D.; Gatti, F.; Giove, A.; Verucchi, M.; Malatesta, D.; Musiu, N.; Toschi, A.; Popitanu, S.; Bagni, F.; et al. er.autopilot 1.0: The Full Autonomous Stack for Oval Racing at High Speeds. *Field Robot.* **2024**, *4*, 99–137. [CrossRef]
15. Betz, J.; Betz, T.; Fent, F.; Geisslinger, M.; Heilmeier, A.; Hermansdorfer, L.; Herrmann, T.; Huch, S.; Karle, P.; Lienkamp, M.; et al. TUM autonomous motorsport: An autonomous racing software for the Indy Autonomous Challenge. *J. Field Robot.* **2023**, *40*, 783–809. [CrossRef]
16. Heilmeier, A.; Wischnewski, A.; Hermansdorfer, L.; Betz, J.; Lienkamp, M.; Lohmann, B. Minimum Curvature Trajectory Planning and Control for an Autonomous Race Car. *Vehicle Syst. Dyn.* **2020**, *58*, 1497–1527. [CrossRef]
17. Demeter, Z.; Bogdán, P.; Bogár-Németh, Á.; Bári, G. Scalable Supervisory Architecture for Autonomous Race Cars. In Proceedings of the 2024 IEEE Intelligent Vehicles Symposium (IV), Jeju Island, Republic of Korea, 2–5 June 2024; pp. 264–271.