Controller Design For Autonomous Racing Vehicle

Kaijie Zhu
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN, USA
zhu1089@purdue.edu

Zhishan Wang
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN, USA
wang6158@purdue.edu

Kieran Venkat Desireddi
First-Year Engineering
Purdue University
West Lafayette, IN, USA
kdesired@purdue.edu

Chi-Che Chen
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN, USA
chen5966@purdue.edu

Jacob Forrest
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN, USA
forrest3@purdue.edu

Brian Sam Lee
First-Year Engineering
Purdue University
West Lafayette, IN, USA
lee5372@purdue.edu

Abstract— **This study presents a comprehensive system design and evaluation pipeline for autonomous vehicle control, which integrates a physics-based vehicle model, controller implementation, and closed-loop testing within the MATLAB environment. The physical model was developed incrementally, beginning with a minimal runnable dynamics representation and subsequently refined to incorporate more realistic tire behavior, longitudinal resistance effects, and actuator feasibility constraints. These enhancements were aimed at improving stability, repeatability, and practical executability.**

Utilizing a multi-objective evaluation framework that balances speed and safety, the proposed approach was assessed against a baseline controller through Pareto-based comparisons, constraint violation metrics, and indicators of control smoothness. Experimental results demonstrate a significant improvement in overall performance: lap times are reduced to approximately ninety-three percent of the baseline while the rate of violations decreases substantially; additionally, steering actions exhibit greater smoothness and consistency with realistic execution limits.

These findings indicate that the proposed system achieves an improved trade-off between speed and safety while providing a reliable foundation for further development as well as future real-world validation.

I. INTRODUCTION

Autonomous racing has emerged as a demanding benchmark for high-performance vehicle control, pushing the limits of perception, planning, and control under extreme dynamic conditions. Unlike conventional autonomous driving, racing scenarios require vehicles to operate near handling limits while maintaining safety, stability, and responsiveness. Small modeling errors, external disturbances, or suboptimal controller tuning can lead to significant performance degradation or unsafe behavior. As a result, the design of robust, fast, and smooth control strategies is a critical challenge for high-speed autonomous racing systems.

Traditional control approaches such as Proportional–Integral–Derivative (PID) controllers remain widely used due to their simplicity, interpretability, and ease of implementation. When properly tuned, PID controllers can achieve acceptable tracking performance in structured environments. However, autonomous racing introduces rapidly changing dynamics, nonlinear tire forces, and external disturbances that can limit the effectiveness of classical PID control. More advanced control strategies, including Model Predictive Control (MPC), Sliding Mode Control (SMC), and Active Disturbance Rejection Control (ADRC), have been proposed to improve robustness and performance in such settings. These methods aim to explicitly or implicitly handle modeling uncertainty and disturbances while maintaining real-time feasibility.

In this work, we investigate and compare multiple feedback control strategies for an autonomous racing vehicle within the RAPID-ARC platform. Our objective is to develop a closed-loop control system that achieves a balance between lap-time efficiency, safety constraint satisfaction, and smooth control action. To quantitatively evaluate controller performance, we define a composite objective function that incorporates normalized lap time, safety violation metrics, and control smoothness. This formulation enables systematic comparison of controllers under identical track conditions and highlights trade-offs between aggressive driving and stable behavior.

We begin by constructing a simplified dynamic model of the racing vehicle and implementing a baseline PID controller to establish reference performance. Parameter tuning is performed to analyze the effects of proportional, integral, and derivative gains on stability, responsiveness, and overshoot. Building on this baseline, we explore Active Disturbance Rejection Control, which leverages an extended state observer to estimate and compensate for internal and external disturbances in real time. In particular, a nonlinear ADRC (NLADRC) framework is employed to improve transient response

All authors are with Purdue University, West Lafayette, IN, USA.

and disturbance rejection compared to conventional PID control, while maintaining computational efficiency suitable for real-time implementation.

Experimental evaluations are conducted in simulation across multiple test scenarios to assess speed, safety, and smoothness. Results demonstrate that both PID and ADRC controllers can meet basic performance requirements, while ADRC exhibits improved robustness to disturbances and smoother control behavior under aggressive driving conditions. These findings suggest that disturbance-aware control strategies offer promising advantages for autonomous racing applications, particularly in environments with uncertain or rapidly varying dynamics. The remainder of this paper describes the system design, controller formulations, experimental setup, results, and future directions for improving autonomous racing performance.

II. System design

A. Starting from the goal, first clarify what problem the model aims to solve

Before starting to write the model, it is necessary to clarify the purpose of this physical model: it is not intended to reproduce all the real details, but to enable the controller to be stably and repeatably tested in the simulation. That is to say, the model must be able to reflect the main motion laws of the vehicle during acceleration, braking and steering, while at the same time being lightweight enough to run quickly during repeated iterations. Therefore, we first defined the scope of application of the model, such as only considering the vehicle's movement on a plane, maintaining consistent road conditions within a test scenario, and allowing some details that are difficult to observe directly to be replaced by adjustable parameters.

B. First, unify the "input and output interfaces" to ensure that no matter how you upgrade later, it won't be chaotic

The first step in building a model is not to write the dynamics, but to clearly define the input and output of the model. We have clarified which control instructions the model needs to receive, such as acceleration or braking instructions, as well as steering instructions. At the same time, it is necessary to clearly define which quantities the model should output for control and evaluation, such as the vehicle's motion state, position and attitude, as well as error-related indicators that can be used to measure the tracking effect. The significance of this step is that no matter which part of the model details we replace in the future, the controller and test script do not need to be modified. As long as the model still follows the same input and output format, the effects of the new and old versions can be continuously compared.

C. First, create the "smallest model that can run" to complete the closed-loop simulation

Next, we will build the most basic version, enabling the vehicle to move, turn and complete a simple scenario in the simulation. The key points of this minimum model are stability and operability: it does not pursue the precision of extreme working conditions, but can ensure that the controller runs smoothly, the data flow is complete, and the results can be drawn. In MATLAB, this step typically corresponds to a simplest simulation main script and a core function for updating the vehicle status. Through continuous iteration at fixed time steps, it generates trajectory, speed curve, and control input curve.

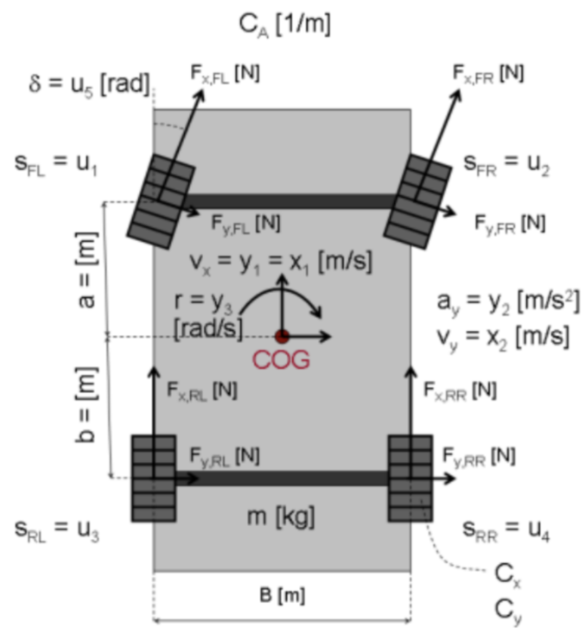

Fig. 1: Car Dynamic Model

D. Prioritize upgrading the parts that have the greatest impact and the greatest error: tires and steering response

After the minimum model runs smoothly, we will observe which behaviors are the least intuitive or have the greatest impact on the control effect. Usually, the most crucial aspect is the steering response, that is, the lateral force and attitude change generated by the vehicle when it turns. Therefore, we gradually upgraded the description methods related to tires on the basis of the basic model. The upgrade strategy is not achieved in one step. Instead, it starts with a simple and controllable version, then introduces more realistic nonlinear characteristics, and retains the switching mechanism. This way, different model versions can be compared under the same test conditions to confirm the actual benefits brought by the upgrade. At the same time, to be closer to real vehicles, we will also ensure that the tire response is not completed instantaneously but is gradually established, thereby more realistically presenting the dynamic delay during rapid handling.

E. Further upgrade the vertical section: Make the resistance more reasonable to make acceleration and deceleration closer to reality

After the horizontal behavior becomes more reliable, the next step is usually to deal with the error of the vertical speed variation. We will break down the originally overly rough description of resistance, allowing resistance from different sources to present different effects in different speed ranges. This can not only explain the ”dragging sensation” at low speeds but also the significantly enhanced resistance effect at high speeds. This makes the speed curves of the model in scenarios such as straight-line acceleration, braking, and cornering acceleration more in line with expectations, thereby enhancing the credibility of the controller’s tests in speed planning and traction control.

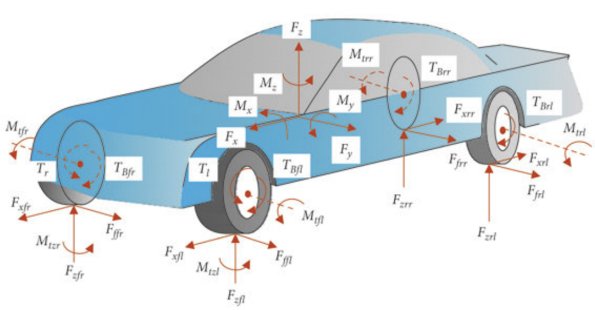

Fig. 2: Car Dynamics Model

F. Add actuator restrictions to make the model ”unable to perform certain commands just like a real car”

When the model is sensitive enough, a common problem will arise: the controller may output very aggressive instructions in the simulation, but in reality, the vehicle is limited by factors such as steering speed, maximum driving force, and maximum braking force, and thus cannot achieve this. So we incorporate the constraints and response characteristics of the actuator layer into the model: allowing instructions to be limited, with restricted change speeds, and even slight lags. This layer is extremely crucial because it can distinguish between ”what the controller theoretically intends to do” and ”what the vehicle can actually do”, thus avoiding the outcome that seems perfect in simulation but is unattainable in reality.

G. Handle numerical stability and reproducibility to ensure that the model can operate stably for a long time

When the model becomes more realistic and complex, numerical stability will become a problem that must be solved. We unified the time step of the simulation to ensure that the update rhythm of all modules is consistent. At the same time, protection mechanisms should be added at links where extreme values may occur to prevent unreasonable jumps when the vehicle speed is very low or the steering changes rapidly. The objective of this step is to ensure that the model does not diverge or explode during long-term simulation, and that consistent results are obtained each time it is run, facilitating comparative experiments and parameter regression.

H. Practical Application in MATLAB: Hierarchical Verification from Unit Testing to Track Scenario Testing

To prove that the model is not ”seemingly reasonable” but truly ”usable and reliable”, we adopted a hierarchical testing process in MATLAB:

Unit test: Test respectively whether the tire response, resistance change, and actuator limit meet expectations.

Subsystem testing: Only provide steering input and observe the vehicle’s turning behavior; Only input acceleration and deceleration and observe the speed changes.

Scene testing: Run closed-loop control under trajectory or track conditions that are closer to real tasks, and record changes in errors, control inputs, and vehicle status.

Regression testing: After each model upgrade, the same set of fixed working conditions is automatically run to ensure that the new version does not cause the previously passed tests to suddenly fail.

Through this testing system, we can clearly answer: what improvements have been brought about by the model upgrade, whether new unstable factors have been introduced, and whether the change in controller performance is caused by the controller problem or the model change.

I. Dynamic System Model (Planar Bicycle Model)

a) State and input.:

$$x \triangleq \begin{bmatrix} X & Y & \psi & v_x & v_y & r \end{bmatrix}^\top, \qquad u \triangleq \begin{bmatrix} F_x & \delta \end{bmatrix}^\top. \tag{1}$$

b) Kinematics (global frame).:

$$\dot{X} = v_x \cos\psi - v_y \sin\psi, \tag{2}$$

$$\dot{Y} = v_x \sin\psi + v_y \cos\psi, \tag{3}$$

$$\dot{\psi} = r. \tag{4}$$

c) Dynamics (body frame).:

$$m\dot{v}_x = F_x - F_{yf}\sin\delta + mv_y r - F_{\text{res}}(v_x), \tag{5}$$
$$m\dot{v}_y = F_{yf}\cos\delta + F_{yr} - mv_x r, \tag{6}$$
$$I_z\dot{r} = \ell_f F_{yf}\cos\delta - \ell_r F_{yr}. \tag{7}$$

d) Slip angles.:

$$\alpha_f = \arctan\left(\frac{v_y + \ell_f r}{v_x}\right) - \delta, \tag{8}$$
$$\alpha_r = \arctan\left(\frac{v_y - \ell_r r}{v_x}\right). \tag{9}$$

e) Linear tire forces.:

$$F_{yf} = -C_f\alpha_f, \tag{10}$$
$$F_{yr} = -C_r\alpha_r. \tag{11}$$

f) Resistive forces.:

$$F_{\text{res}}(v_x) = F_{\text{drag}}(v_x) + F_{\text{roll}}, \tag{12}$$
$$F_{\text{drag}}(v_x) = \frac{1}{2}\rho C_d A v_x^2, \tag{13}$$
$$F_{\text{roll}} = C_{rr} mg. \tag{14}$$

g) Compact state-space form and discretization.:

$$\dot{x} = f(x, u), \tag{15}$$
$$x_{k+1} = x_k + \Delta t\, f(x_k, u_k), \tag{16}$$
$$x_{k+1} = \Phi(x_k, u_k, \Delta t). \tag{17}$$

III. PID AND PARAMETER TUNING

In order to obtain an accurate metric for how successful our ADRC control scheme is, we need to compare it to a more generalized system. We chose to use a PID control system for this, as it is generally considered to be the golden standard. In order to create a PID control system, we need to tune it for our system to be as good as possible, so that it can be an accurate comparison. We chose to use a manual tuning method for this process.

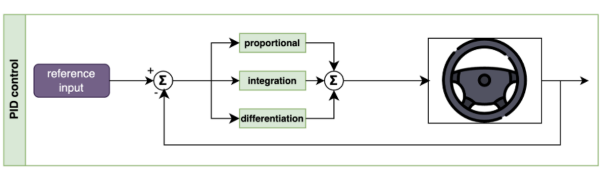

Fig. 3: PID controller design

To start, we set all of our PID parameters (Kp, Ki, and Kd) to zero, and then gradually increased Kp until we saw oscillation with little to no decay over time. We then set Kp to half of this value, and increased Ki until all of our steady state error was gone. As a result of our system being entirely mathematical at this point and not having introduced any error, we did not experience much steady state error and thus did not need to increase Ki significantly. After increasing Ki, we then increase Kd until any overshoot or oscillation is gone. This is a very generalized process that we found to be useful for optimizing our system, and was used to create the system that we will compare our ADRC control scheme to.

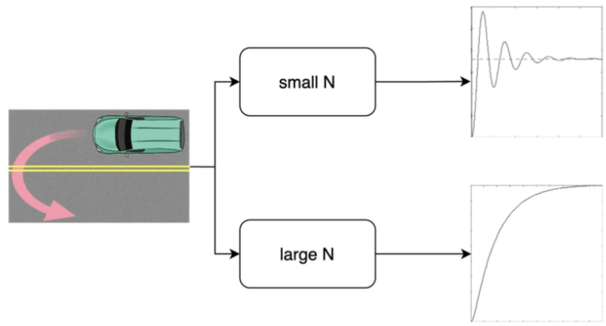

Fig. 4: parameter tuning process

Here are our parameter tuning results:
For PID controller(Lateral):

$$K_p = 0.22,$$
$$K_i = 0.02,$$
$$K_d = 0.045,$$
$$N = 8,$$
$$K_{\text{aw}} = 0.55.$$

For PI controller(Longitudinal):

$$K_{pv} = 0.8,$$
$$K_{iv} = 0.55,$$
$$K_{awv} = 0.12.$$

IV. ACTIVE DISTURBANCE REJECTION CONTROL

A. Why we use ADRC, not PID?

We need to deal with the tremendous of disturbance when our autonomous vehicles is running in real world. For PID, although it's really easy to use. However, it can't deal with such disturbance from the outside. The main goal for our whole system is design a non-linear racing car system with great uncertainties. For PID, it's widely used for linear system with linear control. But it's not a good idea if we add more uncertainties. In this way, we will start in non-linear ADRC and check if this can be used in our system.

B. Non-linear ADRC

We firstly designed a non-linear ADRC system. Since our control objectives are the steering angle command and the longitudinal acceleration command, we implement NLADRC in both the lateral and longitudinal dimensions. Based on our model design, we have the road friction, the wind. They can all be counted as "total disturbance" and they can all be observed by the nonlinear extended state observer (NLESO). To improve transient performance for the transient process. Unlike PID which will produce lots of overshoot while facing the step response, we additionally employ a nonlinear tracking differentiator to produce the better transaction. Then, we use the SEF(state error feedback). It can be counted as the advanced-level PD controller. SEF reads the tracking errors and generates the feedback action to complete the closed-loop response. Compared with a conventional PD controller, the nonlinear SEF can provide improved transient performance and robustness.

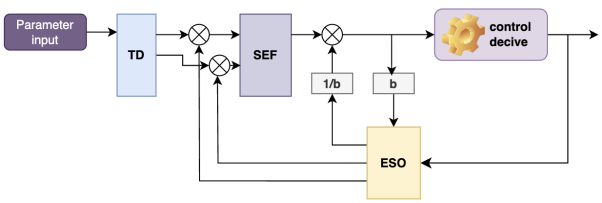

Fig. 5: Lateral non-linear ADRC design

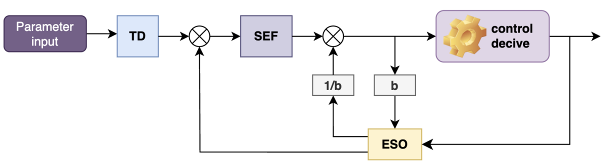

Fig. 6: Longitudinal non-linear ADRC design

C. General non-linear ADRC formula(second order)

Down here are the lateral non-linear ADRC formulas. Longitudinal NLADRC uses a first-order TD and a second-order NLESO with nonlinear SEF. Thus, we just need to decrease the order for the following formulas.

a) Non-linear helping function 1:

$$\mathrm{fal}(e,\alpha,\delta) = \begin{cases} \dfrac{e}{\delta^{1-\alpha}}, & |e| \le \delta, \\ \mathrm{sign}(e)\,|e|^{\alpha}, & |e| > \delta. \end{cases} \tag{18}$$

b) Non-linear helping function 2:

$$\begin{cases} d = r\,h, \qquad d_0 = d\,h, \\ y = x_1 + h\,x_2, \\ a_0 = \sqrt{d^2 + 8r|y|}, \\ a = \begin{cases} x_2 + \dfrac{1}{2}(a_0 - d), & |y| > d_0, \\ x_2 + \dfrac{y}{h}, & |y| \le d_0, \end{cases} \\ \mathrm{fst}(x_1, x_2, r, h) = \begin{cases} -\dfrac{r\,a}{d}, & |a| \le d, \\ -r\,\mathrm{sign}(a), & |a| > d. \end{cases} \end{cases} \tag{19}$$

c) Non-linear TD:

$$v_1[k{+}1] = v_1[k] + T_s\, v_2[k], \tag{20}$$

$$v_2[k{+}1] = v_2[k] + T_s\, \mathrm{fst}\big(v_1[k] - v_0,\; v_2[k],\; r,\; h_0\big) \tag{21}$$

d) Non-linear ESO:

$$b = \frac{v_x^2}{\max(0.5, L_{ff})}, \tag{22}$$

$$\varepsilon[k] = z_1[k] - e_y[k], \tag{23}$$

$$z_1[k{+}1] = z_1[k] + T_s\big(z_2[k] - \beta_1\,\varepsilon[k]\big), \tag{24}$$

$$\xi[k] = z_3[k] - \beta_2\,\mathrm{fal}\big(\varepsilon[k], \alpha_1, \delta_\epsilon\big) + b\,u[k], \tag{25}$$

$$z_2[k{+}1] = z_2[k] + T_s\,\xi[k], \tag{26}$$

$$z_3[k{+}1] = z_3[k] - T_s\,\beta_3\,\mathrm{fal}\big(\varepsilon[k], \alpha_2, \delta_\epsilon\big). \tag{27}$$

e) Non-linear SEF:

$$e_1[k] = v_1[k] - z_1[k], \tag{28}$$

$$e_2[k] = v_2[k] - z_2[k], \tag{29}$$

$$\psi_1[k] = \mathrm{fal}\big(e_1[k], \alpha_{1,\mathrm{SEF}}, \delta_{\mathrm{SEF}}\big), \tag{30}$$

$$\psi_2[k] = \mathrm{fal}\big(e_2[k], \alpha_{2,\mathrm{SEF}}, \delta_{\mathrm{SEF}}\big), \tag{31}$$

$$u_0[k] = \beta_{1,\mathrm{SEF}}\,\psi_1[k] + \beta_{2,\mathrm{SEF}}\,\psi_2[k], \tag{32}$$

$$u[k] = \frac{u_0[k] - z_3[k]}{\max(0.1, b)}. \tag{33}$$

D. The performance of the non-linear ADRC and parameter tuning

For our non-linear ADRC, we can observe tremendous or parameters and gains. To make the whole system perform well should be really hard for us. Our next step is using some algorithm like particle swarm optimization to help us doing the parameter tuning. But currently we observe the LADRC should be the better choice with fewer parameters and easy to do the tuning.

E. Linear ADRC design

To decrease the difficulties of the parameter tuning, we tried to just remove the tracking differentiator and the non-linear helping functions in our non-linear design. And we are still using LADRC consists of a Linear Extended State Observer (LESO) to estimate "total disturbance," and a linear State Error Feedback (SEF) to generate control actions with the help of the error to give us the better feedback. And we are still doing the second order in our lateral dimension and first order on our longitudinal dimension.

F. Performance

With the few tuning steps, we observe the LADRC can work within the bigger initial offset and can have less inputs(contain disturbance). We then put this controller into our whole design.

V. RESULTS

Our results should be in line with our design goals, rapidity, stability and safety indicators. From our index chart, we can find that after our designed controller, the simulated lap time of our racing car in the simulation is faster than before. At the same time, the input of the controller is also smoother than before, which will bring better stability to the system.

At the same time, we observed the safety of the system and found that our design had found a solution that could not only meet the requirements of running speed (lap time) but also significantly enhance the safety of the racing car.

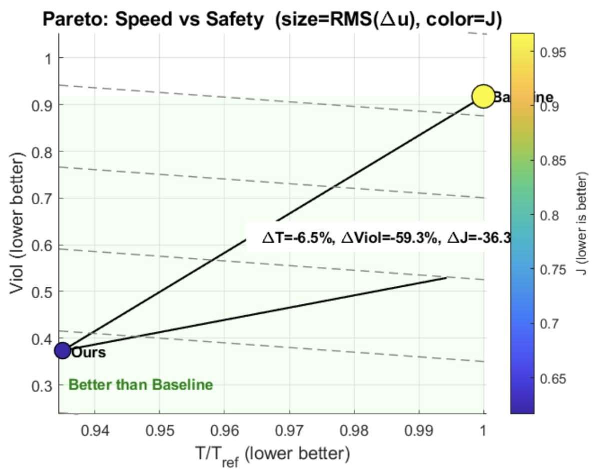

Fig. 7: results graph

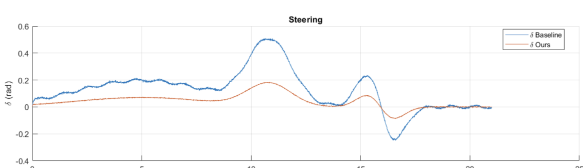

Fig. 8: results graph

Faster: On the speed-safety compromise graph, our method significantly increased lap times, reducing lap times to approximately 93 percent of the baseline (roughly equivalent to an increase in lap times), and overall lap times improved by about 6.5

Safer: The constraint/violation rate has significantly decreased. The improvement shown in the chart is approximately 59percent, indicating that while it is faster, there is no "risk", but rather it is more stable.

Smoother and more executable: The comparison of the steering curves below shows that our control input has smaller fluctuations, lower peaks, and less oscillations, and the actions are more in line with the actual vehicle's achievable handling (the overall cost index has also decreased simultaneously, by approximately 36

VI. CONCLUSIONS

This project has successfully delivered a comprehensive and testable system design pipeline that integrates the physical vehicle model, the controller, and quantitative evaluation within MATLAB. Beginning with a minimal runnable dynamics model, the framework was iteratively enhanced to incorporate more realistic tire and resistance behaviors, as well as considerations for actuator feasibility. This iterative refinement ensured that the simulation remained stable, repeatable, and suitable for closed-loop testing.

The final results demonstrate consistent improvements over the baseline across multiple dimensions. The optimized design achieves enhanced lap performance, reducing lap time to approximately ninety-three percent of the original while significantly lowering the violation rate. This indicates improved safety and better adherence to constraints. Furthermore, control actions have become smoother and more aligned with real execution limits, thereby minimizing unnecessary oscillations and enhancing overall practicality. These outcomes confirm that the proposed system offers an improved speed–safety trade-off along with more deployable control behavior, establishing a robust foundation for ongoing iteration and future on-track validation.

ACKNOWLEDGMENT

The authors would like to thank the Purdue University VIP Program for supporting this project. We also thank Prof. Labi, Mr. Hoyos, Mr. Freed and Mr. Ajagu. In addition, we acknowledge helpful discussions and feedback from our teammates and course staff during the controller design and evaluation.

REFERENCES

[1] D. Yoo, S. S.-T. Yau, and Z. Gao, "On convergence of the linear extended state observer," in *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications and 2006 IEEE International Symposium on Intelligent Control*, Munich, Germany, 4–6 Oct. 2006, pp. 1645–1650.

[2] Z. Wang, *Research on High Performance Control Algorithm of Permanent Magnet Synchronous Electric Drive System Based on Active Disturbance Observer*, M.S. thesis, Hefei University of Technology, Hefei, China, 2020, doi: 10.27101/d.cnki.ghfuq.2020.000326.

[3] J. Han, "From PID technique to active disturbance rejection control," *Control Engineering*, no. 3, pp. 13–18, 2002, doi: 10.14107/j.cnki.kzgc.200203.003.

[4] Z. Li, "Research on self-tuning strategy of permanent magnet synchronous motor active disturbance rejection control parameters," Ph.D. dissertation, Huazhong University of Science and Technology, Wuhan, China, 2021, doi: 10.27157/d.cnki.ghzku.2021.004554.

[5] H. Feng, “Design of control system for permanent magnet synchronous motor based on active disturbance rejection control,” Ph.D. dissertation, Hubei University of Automotive Technology, Shiyan, China, 2023, doi: 10.27847/d.cnki.ghbqc.2023.000050.

[6] Y. Sun, “Design and implementation of ADRC parameter auto-tuning for permanent magnet synchronous motor,” M.S. thesis, Dalian Jiaotong University, Dalian, China, 2020, doi: 10.26990/d.cnki.gsltc.2020.000230.

[7] J. Wen and B. Cao, “Sensorless ADRC speed control system for embedded permanent magnet synchronous motor,” *Proceedings of the CSEE*, vol. 29, no. 30, pp. 58–62, 2009, doi: 10.13334/j.0258-8013.pcsee.2009.30.010.

[8] G. Herbst and R. Madonski, *Active Disturbance Rejection Control: From Principles to Practice*. Cham, Switzerland: Birkhäuser, Springer Nature Switzerland AG, 2025, doi: 10.1007/978-3-031-72687-3.

[9] G. Herbst, “A simulative study on active disturbance rejection control (ADRC) as a control tool for practitioners,” *Electronics*, vol. 2, no. 3, pp. 246–279, Aug. 2013, doi: 10.3390/electronics2030246.

[10] The MathWorks, Inc., “Robotics and autonomous systems,” *MATLAB Simulink Documentation*, 2025. [Online]. Available: `https://www.mathworks.com/help/overview/robotics-and-autonomous-systems.html` [Accessed: 19-Dec-2025].

[11] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, “A survey of autonomous driving: Common practices and emerging technologies,” *IEEE Access*, vol. 8, pp. 58443–58469, 2020, doi: 10.1109/ACCESS.2020.2983149.

[12] A. Nerode and W. Kohn, “An autonomous systems control theory: an overview,” in *Proc. IEEE Symposium on Computer-Aided Control System Design*, Napa, CA, USA, 1992, pp. 204–210, doi: 10.1109/CACSD.1992.274432.

[13] S. M. Veres, L. Molnar, N. K. Lincoln, and C. P. Morice, “Autonomous vehicle control systems — a review of decision making,” *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 225, no. 2, pp. 155–195, 2011, doi: 10.1177/2041304110394727.