Model Predictive Controller for Autonomous Vehicle Overtaking

Dulanjana Perera
Final Project Report
ENTC 689 – Applied Modeling, Controlling and Optimization Techniques in Automation Industries
12/08/2023

Multidisciplinary Engineering
College of Engineering
Texas A&M University
Texas

1 Introduction

Autonomous driving is a well-researched and practically proven concept. The initial idea of autonomous vehicles was coined in 1920 by Pontiac. Since then, this paradigm has developed rapidly with technology to the current state. The primary reason for significant attraction towards autonomy is it increases the productivity and the safety of human beings. A vehicle is a tool that humans regularly use, although it is a leading cause of human death. Early development only focused on controlling the vehicle to track trajectory. However, recent developments try to guarantee the safety of the controller's action. Therefore, reliable sensors and robust controllers are required to ensure safety.

The general hierarchical control systems plan the path and control the vehicle along the path. However, the general path planners, such as RRT* [1], [2], D* lite[3], PRM and APF[4], do not account for the vehicle dynamics, and the vehicle and the controller cannot achieve the resulting path. Therefore, the controller is also vulnerable to unforeseen situations such as the sudden appearance of a vehicle during the overtake or sudden road cross by a pedestrian [5]. These situations need extreme maneuvering and hierarchical controllers are unable to perform better. For this reason, the concept of a single controller that considers the vehicle dynamics and is able to plan the path is used. It can eliminate the middle ground and increase the control frequency, which improves performance. Moreover, the controller's planning and decisions are based on the vehicle dynamics.

The model predictive controller (MPC) is a widely used predictive controller that considers future actions to improve the current action [6]. Many theories have been developed to improve the efficiency of the MPC and the accuracy of the result. MPC formulates the problem into a constrained optimization problem and computes the best possible action for the next time step. Moreover, MPC can generate the action sequence (the path) to follow without violating constraints such as obstacles or road boundaries. It is also able to handle nonlinear systems and nonlinear constraints as well. This controller is popular in the autonomous vehicle field as it can be run in real-time, and performances are significantly better than other methods [7].

1.1 Problem Definition

This project uses nonlinear MPC to derive a control system to guide the vehicle to overtake safely on a two-lane road. Overtaking vehicles is a complex task that requires numerous things to ensure safety [8]. This is mainly due to the high speed and the uncertain behavior of other vehicles. However, in this project, other vehicles are moving at a constant velocity, and these vehicles don't change lanes abruptly. The proposed controller faces several problems, such as multi-vehicle overtaking, incoming traffic, and parking safely at a given location.

2 Literature Review

The controller is a key component of autonomous driving operation. The path planner generates the path, and the controller's responsibility is to keep the vehicle on the planned path. The most fundamental controller is the PID controller, which is widely used in the manufacturing sector with improved versions [9]. However, PID is also used in the autonomous vehicle sector with an adaptive gain system to increase the robustness [10]. PID is primarily suitable for low-order systems with fewer external disturbances. A system like a vehicle undergoes different external disturbances, and the vehicle model itself is highly nonlinear. Therefore, the usage of a PID controller is not suitable for an application like vehicle overtaking. Moreover, PID cannot handle constraints effectively, which is an essential part of safety.

Autonomy is used in many other fields; for example, missiles need a guiding system to navigate to the target. Pure pursuit is an early method and very intuitive algorithm used to guide the missile [11]. The controller looks at the reference path with a fixed distance, called the 'look-ahead' distance. Then, compute the angle at which the moving object has to turn to go directly at the reference trajectory and command the vehicle. This controller uses the vehicle dynamic to compute the angle so that it won't command infeasible actions. The same concept is used in vehicle control as well [12]. The controller is very efficient and suitable for real-time applications with high-frequency control. In addition, it results in a smooth trajectory to follow to meet the reference path. However, the controller cannot handle small curvatures and is not robust enough to overcome dynamic obstacles. Moreover, parameters that affect the control performance, such as overshoot and oscillation, are anticipated for improperly tuned parameters, especially the 'look-ahead' distance [13]. The Stanley Controller was introduced during the DARPA Grand Challenge [14] to overcome many of these issues. This controller looks ahead and checks the cross-tracking error, improving the performance at hard corners [15]. However, the Stanley controller also suffers from dynamic obstacles and rapid changes in the environment.

The aforementioned controllers are primarily used for the lateral control of the vehicle. However, they are very efficient and used for real-time applications. Although they provide good real-time performance, controlling them is not optimal. The linear quadratic regulators (LQG) or linear quadratic Gaussian (LQG) are optimal control methods used in many fields [16]. As the name implies, the system has to be linear, and an objective function, also known as the cost function, is utilized to compute the optimum action to reduce the objective [17]. The important factor is that cost has to be formulated in quadratic form to apply LQR. The LQG follows the same concept, and it can account for uncertainty, which must follow Gaussian distribution. The analytical solution ensures that the result is optimal for the given objective function. Most autonomous problems can be formulated as quadratic problems, and applying LQR provides control over lateral and longitudinal [18]. Generally, LQR is efficient in computation and has been used in trajectory-tracking problems. However, it is unsuitable for highly nonlinear systems where linearization suffers from numerical instability. The Value Iteration (VI) is an optimal control algorithm similar to LQR. However, the original VI takes substantial time to compute the output, which is therefore unsuitable for real-time vehicle control. It is an iterative version of LQG, but optimization is based on Bellmann optimality and Markovian Decision Process (MDP) [19]. It is a brute force algorithm, resulting in the control policy for the given state space. The VI can be used for finite horizon problems as well as infinite horizon problems [20]. However, the general VI and LQR cannot handle constraints effectively. However, there are studies that improved the LQR algorithm so that it runs efficiently and handles constraints such as obstacles [21]. The main disadvantage of constrained iterative LQR (CILQR) in [21] is it requires a reference trajectory, and the performance depends on the trajectory; for example, vehicle overtaking with incoming traffic needs complex trajectory planning, and the proposed CILQR suffers significantly during that.

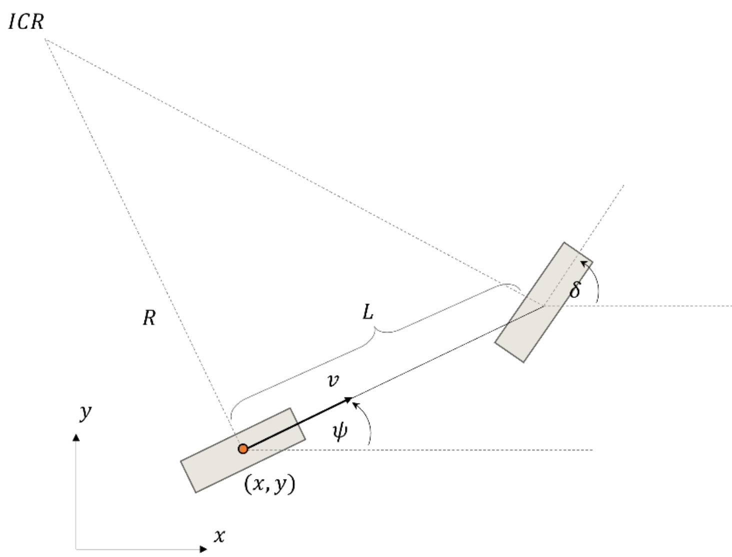
Feedback linearization (FL) is one of the most popular nonlinear controllers, which can handle many dynamic systems. The controller transforms the nonlinear system into a linear through state feedback [22]. However, the controller needs full states that can be obtained from observers and other estimators. The FL can control MIMO systems, such as vehicles, and is used to control the lateral motion of the vehicle for lane change and overtaking [23], [24]. Although it linearizes complex systems, an accurate nonlinear model must be developed. Otherwise, unmodeled dynamics and uncertainties can lead to unstable control.

The model predictive controller is a powerful controller that considers future actions to improve the current action [6]. It uses the system model to predict the future for several steps and checks that the cost associated with it is minimal according to the objective function. Then, a constrained optimization algorithm changes the current and future actions until the objective value reaches the minimum. The method has been used in autonomous driving with extensive research to improve performance [5]. Similar to the FL controller, it can control both longitudinal and lateral direction, and since the system model is employed in the prediction, the outcome is always feasible [25]. Moreover, it can be combined with other techniques to improve the path planning that the controller automatically takes care of. A study combined artificial potential field with MPC to improve the performance of overtaking an obstacle in a narrow gap [26]. Similarly, a vast amount of research has been done on MPC-based autonomous navigation and has proven its excellent performance for real-time applications [8].

In addition to these controllers, machine learning-based controllers have been employed and have shown performance similar to MPC [27]–[30]. This project uses MPC to develop an autonomous vehicle overtaking system. The ability to handle nonlinear systems and constraints drove the decision to select the MPC for the project.

3 System Modeling

A bicycle model of a vehicle is considered to obtain the kinematic model. Here, the velocity at the rear wheel is selected as vehicle velocity. The heading angle rate is equal to the steering angle rate.



$$\dot{x} = v \cdot \cos(\psi)$$
$$\dot{y} = v \cdot \sin(\psi)$$
$$\dot{\psi} = \frac{v}{L} \cdot \tan(\delta)$$
$$\dot{v} = a - \frac{v}{25}$$

- ψ - heading angle
- δ - the steering angle (input)
- v - velocity at the rear wheel
- L - length of the vehicle
- a – acceleration (input)
- x, y - global position of the rear wheel.

The above is the continuous state space model. The $v/25$ represent the damping due to the air and ground friction.

The discrete model is given below.

$$x_{k+1} = x_k + \Delta t \cdot v_k \cdot \cos(\psi_k)$$
$$y_{k+1} = y_k + \Delta t \cdot v_k \cdot \sin(\psi_k)$$
$$\psi_{k+1} = \psi_k + \Delta t \cdot \frac{v_k}{L} \cdot \tan(\delta_k)$$
$$v_{k+1} = v_k + \Delta t \cdot a_k - \Delta t \cdot \frac{v_k}{25}$$

The Jacobian matrices of state space with respect to states and inputs are defined to increase optimization efficiency.

$$J_s = \begin{bmatrix} 1 & 0 & -\Delta t \cdot v_k \cdot \sin(\psi_k) & \Delta t \cdot \cos(\psi_k) \\ 0 & 1 & \Delta t \cdot v_k \cdot \cos(\psi_k) & \Delta t \cdot \sin(\psi_k) \\ 0 & 0 & 1 & \tan(\delta) \cdot \frac{\Delta t}{L} \\ 0 & 0 & 0 & 1 - \frac{\Delta t}{25} \end{bmatrix}$$

$$J_u = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & \Delta t \cdot \frac{v}{L} \cdot (\sec^2(\delta)) \\ \Delta t & 0 \end{bmatrix}$$

The output function is the full state with random noise ω_i as follows.

$$y = \begin{bmatrix} x_k + \omega_1 \\ y_k + \omega_2 \\ \psi_k + \omega_3 \\ v_k + \omega_4 \end{bmatrix}$$

The Jacobian of the output function also defined for fast optimization.

$$J_y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The lane width is 10 meters and vehicle dimensions are 3.5m in length and 2m in width.

4 Controller Design

The system is nonlinear and nonlinear model predictive control method is used. There are a few hard constraints and soft constraints exist.

1. Hard constraints
   a. Lane Width
   b. Obstacle avoidance
   c. Acceleration limits
   d. Steering rate
2. Soft constraints
   a. Final location
   b. Final speed and heading angle

4.1 The cost function

In this project, a reference trajectory is not given. Only the goal location and the input are considered. The $x_H$indicates the final step states.

$$C = \sum_{i=0}^{H-1} \left(x_H^T Q x_H + u_i^T R u_i \right)$$

$$Q = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix}$$

4.2 Constraints

$$|X_i(2)| \leq 2.5$$

$$R_{obs} - \|X_i(1{:}2) - Obstacles\|_2 \leq 0 \quad ; R_{obs} = 3.5$$

$$|u(1)_i| \leq 10$$

$$\left| \frac{u_i(2) - u_{i-1}(2)}{\Delta t} \right| \leq 0.2$$

$$|u_i(2)| \leq \frac{\pi}{3}$$

4.3 Optimizer

Fmincon() is used with the default Sequential Quadratic programming (SQP) algorithm. It approximates the Hessian matrix of the Lagrangian function to compute the search direction. The custom Jacobian functions are used to compute the Hessian matrix and handle the constraints.

5 MATLAB simulation

All the computations and simulations are executed in MATLAB (not Simulink). The nlmpc() tool is used to compute the action for a given horizon. The discrete state functions and constrained functions for obstacles and other hard constraints are defined in separate functions and added to the nlmpc object. Please refer to the MATLAB codes folder.

6 Results

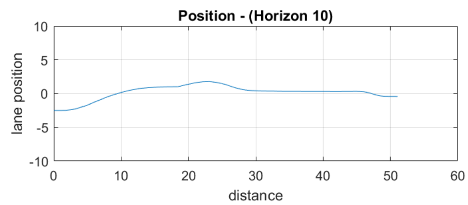
Different scenarios were simulated. However, the lane is only limited to a straight line due to the difficulty of problem formulation. This is discussed in detail in Section 7.

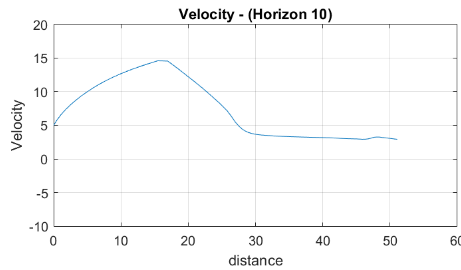
1. Heavy traffic in one direction
2. Two-way traffic
3. Parking

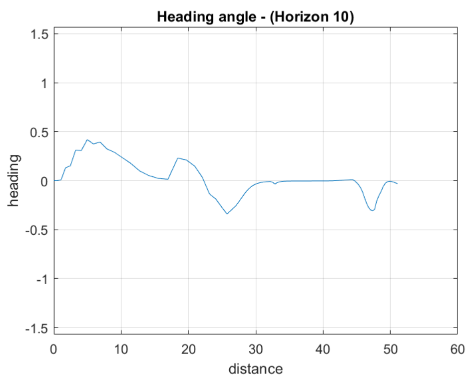
The driving vehicle's initial state is always set to [0, -2.5, 0, 0]. Each iteration took 0.02- 0.015 seconds to compute unless an ill condition was raised (this is discussed in the discussion). The noise is added to each state appropriately. For X and Y, the noise range from (0-10), heading angle; range (0 - 1) and velocity; range (0-10)

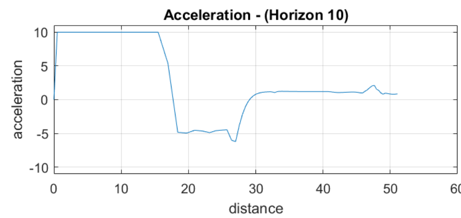
6.1 Heavy traffic one direction

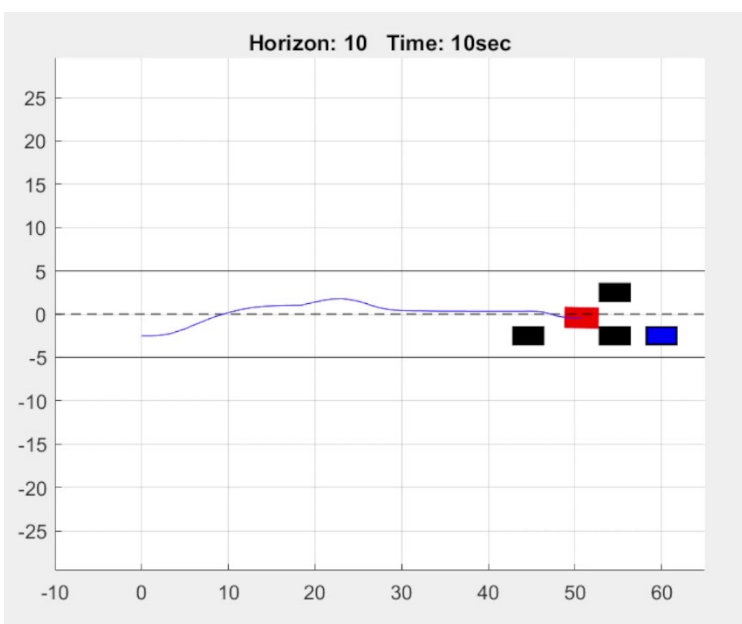
10-step horizon.



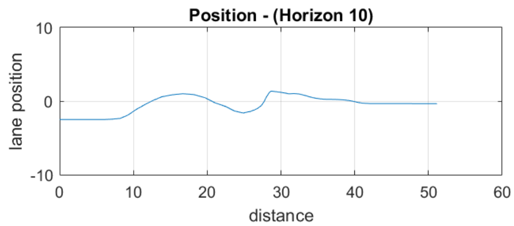


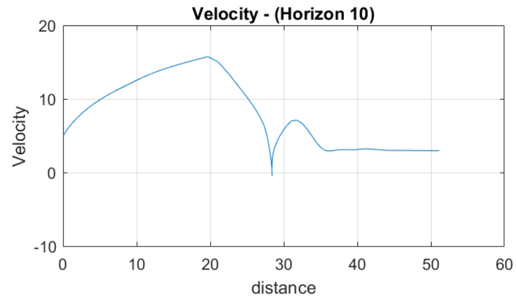


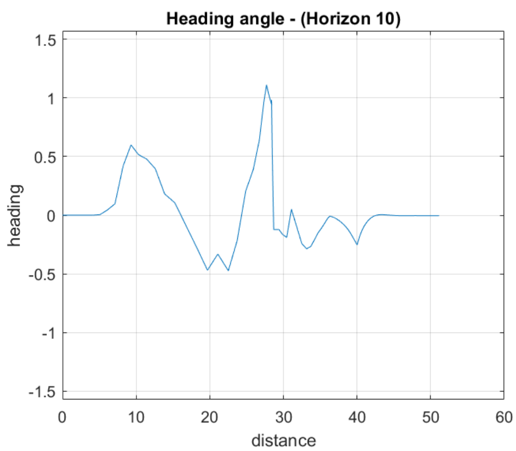


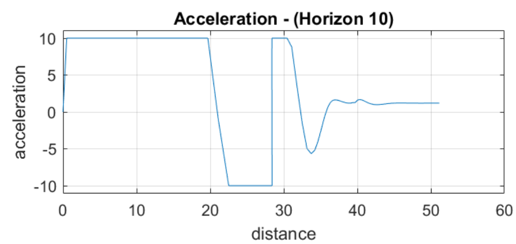


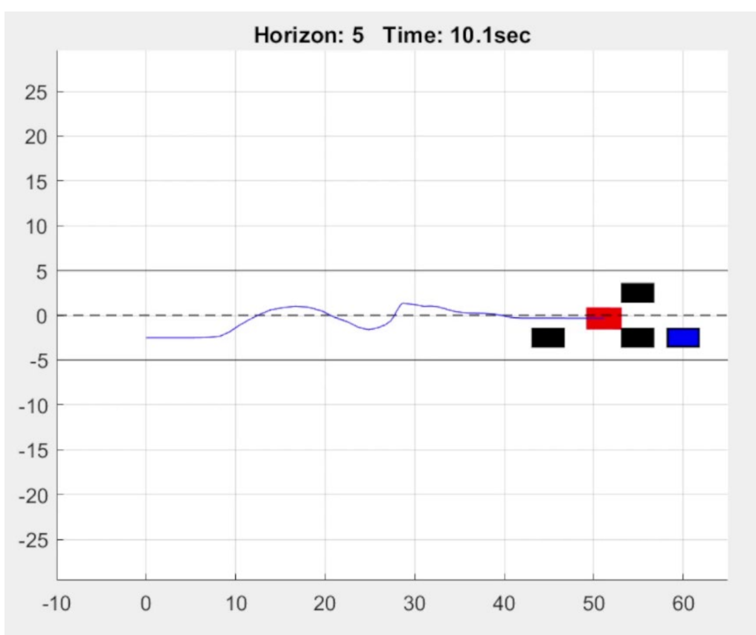
5 step Horizon





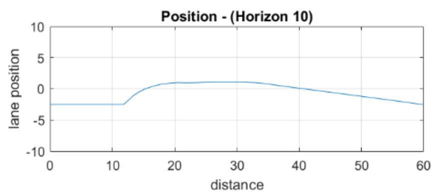


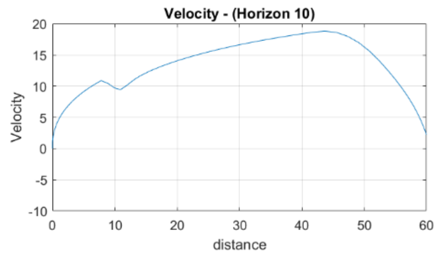


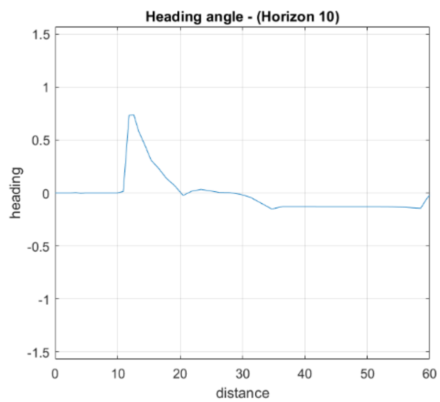


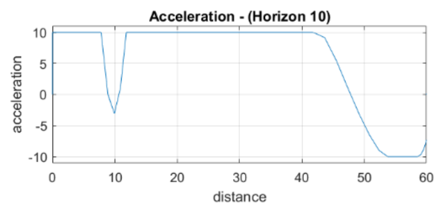
6.2 Two-way traffic

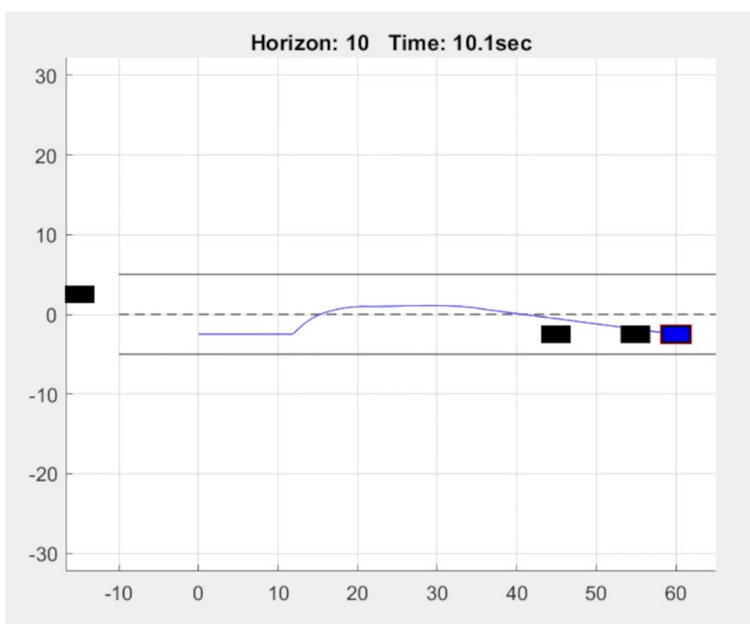
<u>10-step Horizon</u>



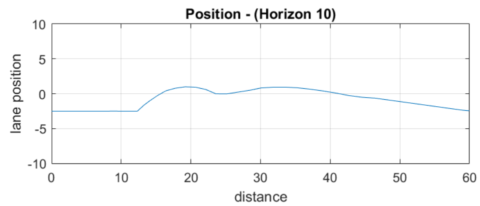


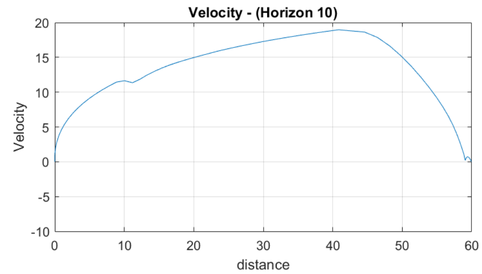


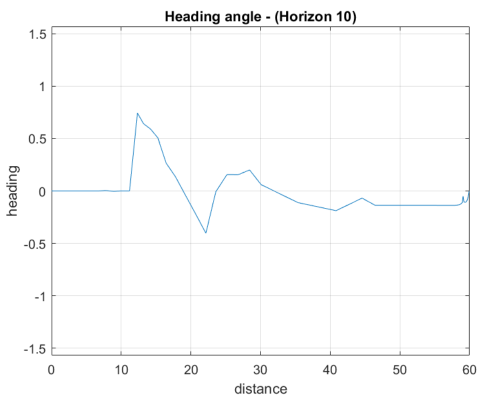


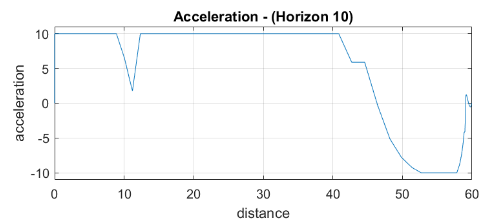


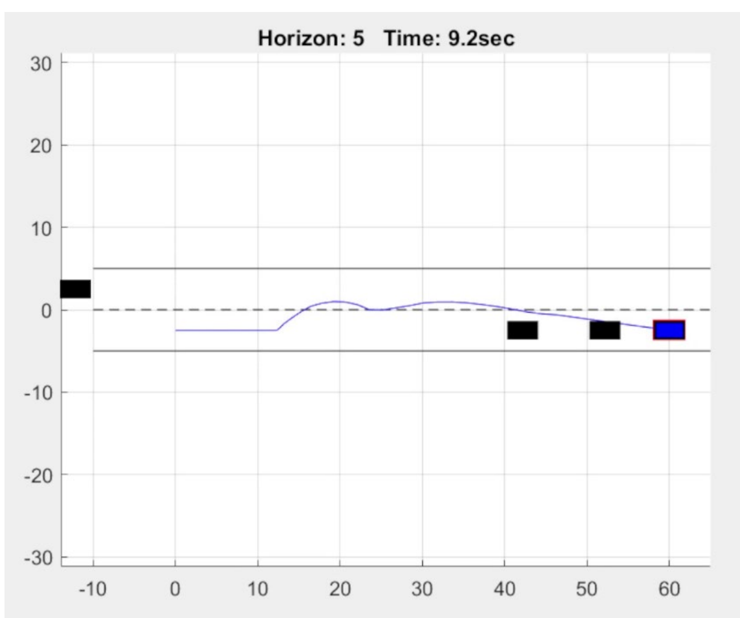
5-step horizon





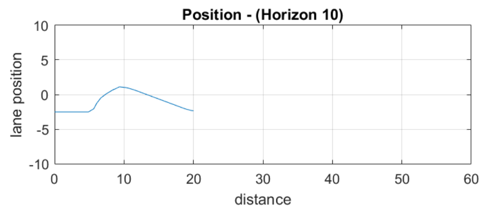


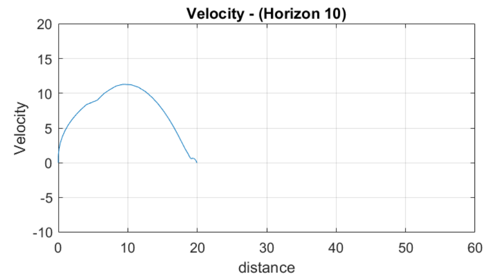


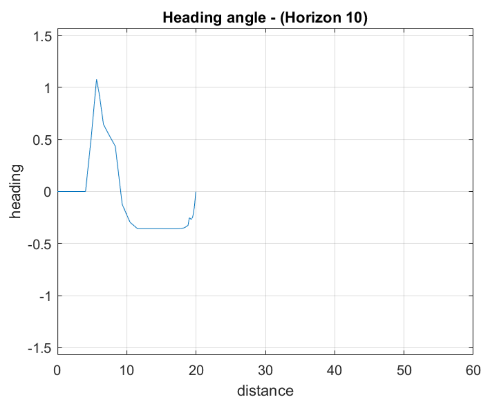


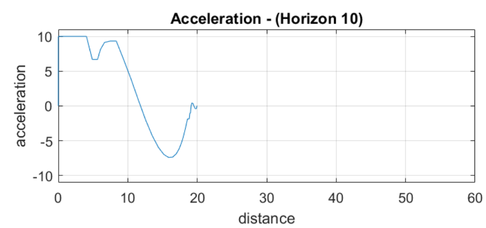
6.3 Parking

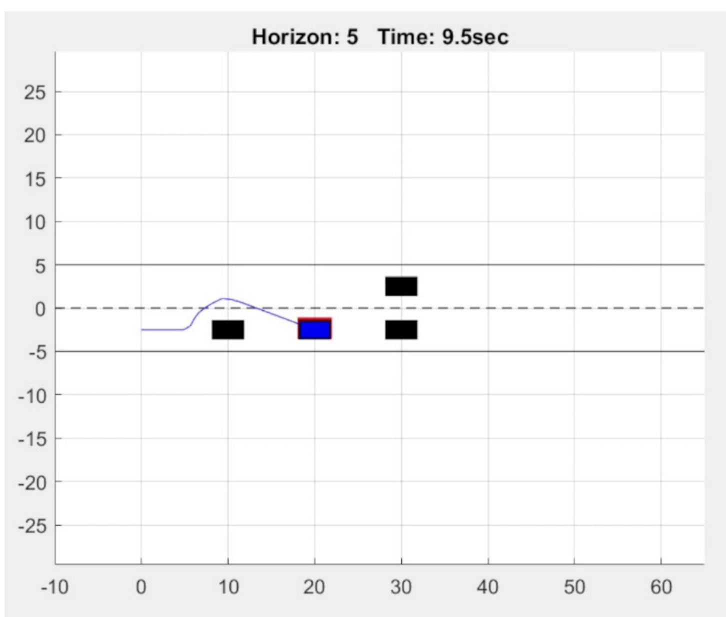
5-step horizon



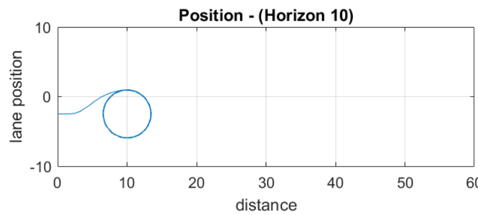


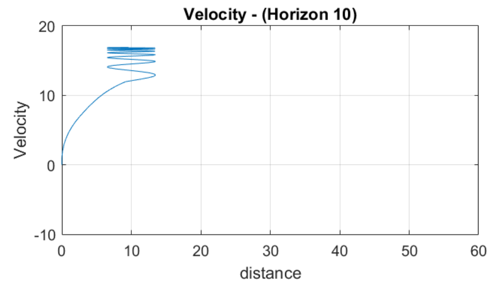


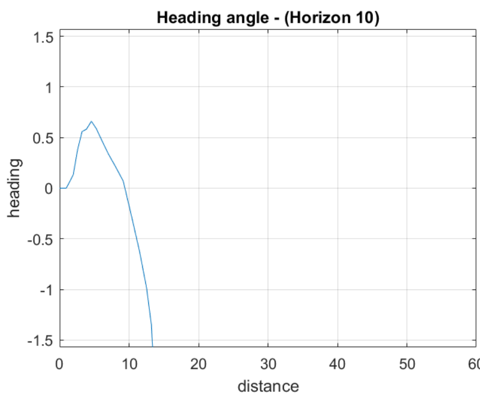


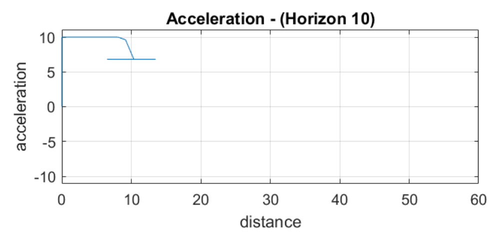


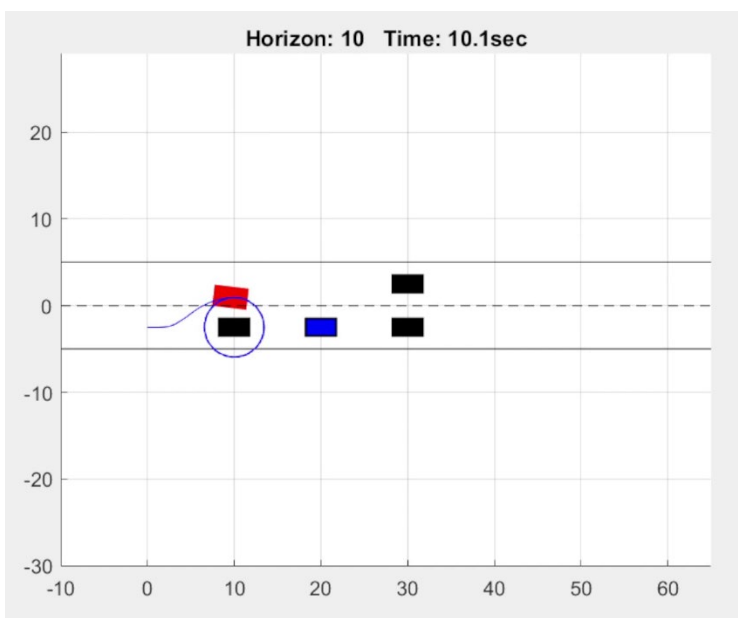
10-step horizon



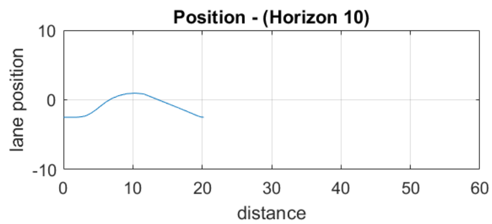


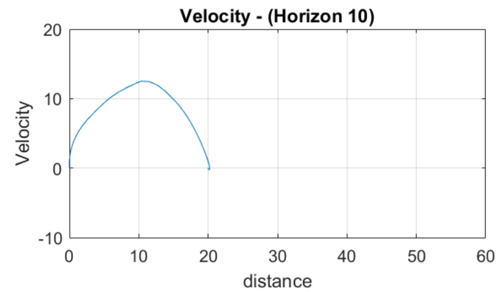


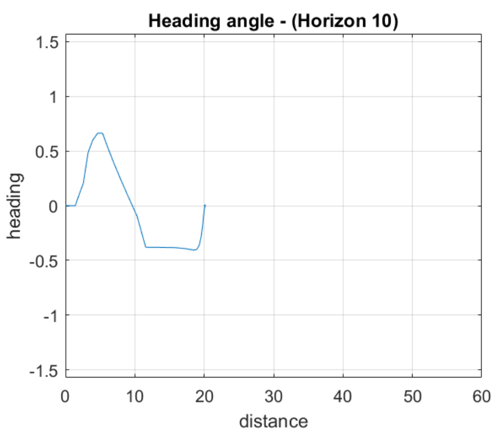


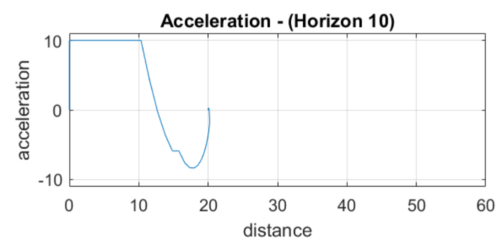


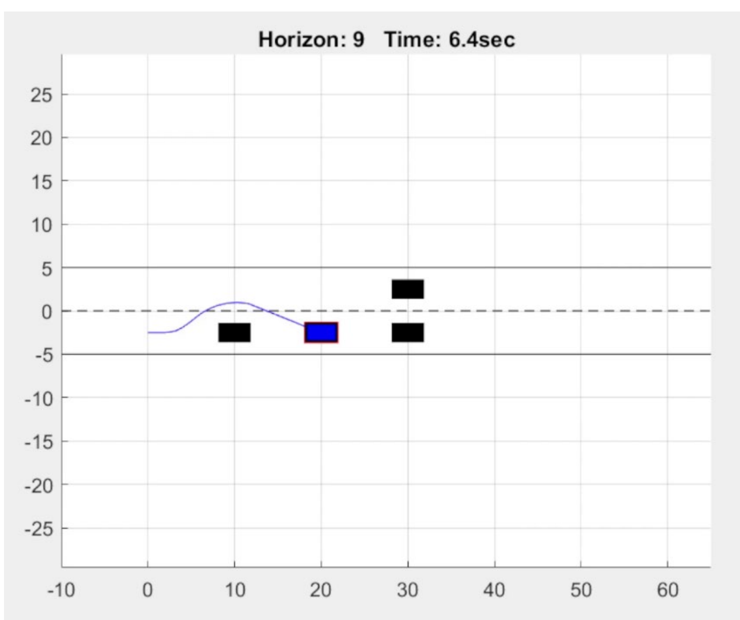
9-step horizon



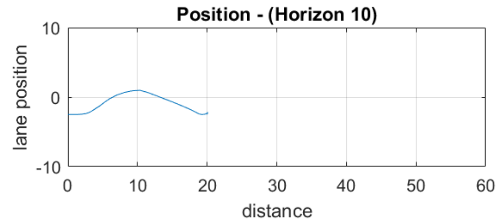


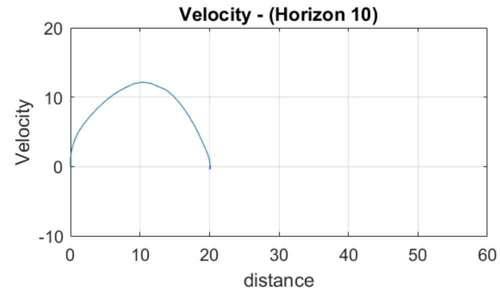


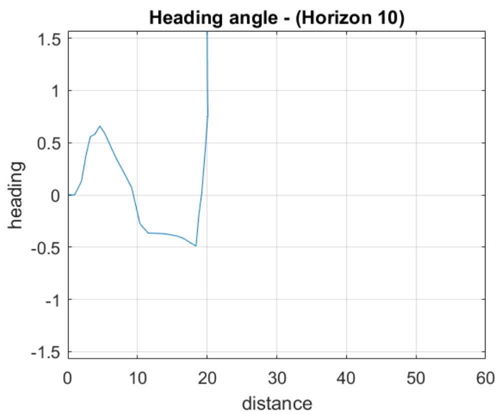


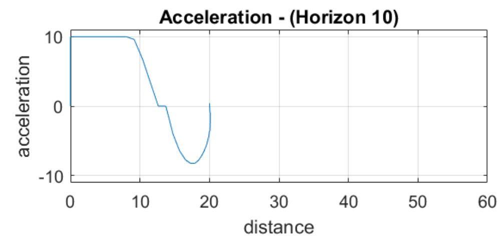


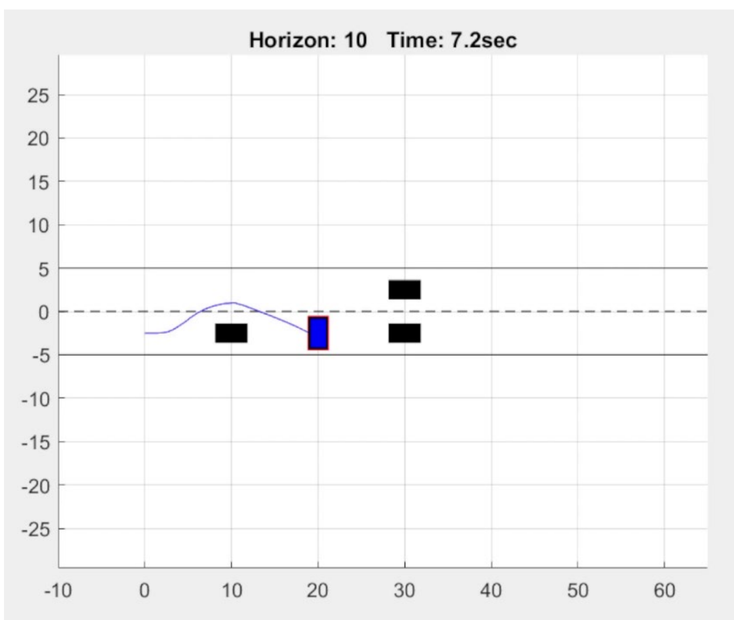
10-step horizon – orientation difference











6.4 Discussion

The heavy traffic in one-way and two-way scenarios performs well. However, a higher horizon results in smooth controlling. It is evident in heading angle variation along the x-axis. Moreover, input constraints are usually saturated since the cost function always tries to reduce the final cost. Therefore, MPC tries to go faster at every possible time. Parking scenario also performs well with large horizons. However, even for a 10-step horizon, MPC only takes 0.01s for the optimization.

An abnormal situation was encountered when a 10-step horizon was used to park the car. Further studies suggested that the 9-step horizon does not have the aforementioned ill-condition. When an ill-condition occurs, the controller takes substantially longer to compute (0.1s), violating all the hard constraints. This is mainly due to numerical stability or suboptimal solutions. It can be due to the poor conditioning in Jacobian as it is used to handle the constraints and searching direction. Since I add noise to the output, it can create infeasible states and unstable the system.

Furthermore, I tried formulating the curved-road problem for the MPC. However, setting up the hard constraints at each horizon calculation is challenging. Transforming the limits using a rotation matrix using the angle of tangent raises limit issues when the angle is higher than $\pi/2$.

7 Learning Objectives

This is the first time I developed an MPC controller. I have tried my own version of the MPC controller with for loops and fmincon() with different algorithms. The controller performs *OK*, but it is not as robust as nlmpc(). Also, I had difficulty adding the constraints to my own MPC.

However, this project gave me good insight into the MPC, especially how to formulate a problem for an MPC. Even though I didn't formulate the problem in quadratic form, setting up discrete state transition functions and custom constraints for obstacles helped me understand the controller better. After completing the project, I gained confidence in developing an MPC controller without using the MATLAB toolboxes. Moreover, I didn't use the Simulink blocks since they hide the MPC formulation. Since I wanted to learn in-depth, I used MATLAB nlmpc() tool and developed my own simulation environment.

8 Conclusion

MPC is a powerful controller which can be used in any problem if properly formulated. The developed controller does not use any reference trajectory but maintains the vehicle position within boundaries. The ability to change the gains and other parameters improves the performance, such as smooth motion and robustness to the unseen inputs (refer videos). By formulating the problem properly for more complex scenarios such as curve paths, and sensor malfunctions, one can obtain very robust controller with guaranteed safety.

9 Support Materials

Video: https://youtu.be/wrtBWJXG60M

GitHub repo: https://github.com/DulanjanaPerera/MPC_vehicleOvertaking.git

10 References

[1] Q. Ge, A. Li, S. Li, H. Du, X. Huang, and C. Niu, "Improved Bidirectional RRT* Path Planning Methods for Smart Vehicle," *Mathematical Problems in Engineering*, vol. 2021, p. e6669728, May 2021, doi: 10.1155/2021/6669728.

[2] I.-B. Jeong, S.-J. Lee, and J.-H. Kim, "Quick-RRT*: Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate," *Expert Systems with Applications*, vol. 123, pp. 82–90, Jun. 2019, doi: 10.1016/j.eswa.2019.01.032.

[3] Y. Yao *et al.*, "Path Planning Method Based on D* lite Algorithm for Unmanned Surface Vehicles in Complex Environments," *China Ocean Eng*, vol. 35, no. 3, pp. 372–383, Jul. 2021, doi: 10.1007/s13344-021-0034-z.

[4] P. Wang, S. Gao, L. Li, B. Sun, and S. Cheng, "Obstacle Avoidance Path Planning Design for Autonomous Driving Vehicles Based on an Improved Artificial Potential Field Algorithm," *Energies*, vol. 12, no. 12, Art. no. 12, Jan. 2019, doi: 10.3390/en12122342.

[5] S. Yu, M. Hirche, Y. Huang, H. Chen, and F. Allgöwer, "Model predictive control for autonomous ground vehicles: a review," *Auton. Intell. Syst.*, vol. 1, no. 1, p. 4, Aug. 2021, doi: 10.1007/s43684-021-00005-z.

[6] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*, vol. 2. Nob Hill Publishing Madison, WI, 2017. Accessed: Dec. 05, 2023. [Online]. Available: http://www.mtas.ru/search/search_results.php?publication_id=21880

[7] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, Mar. 2016, doi: 10.1109/TIV.2016.2578706.

[8] B. Németh, T. Hegedűs, and P. Gáspár, "Model Predictive Control Design for Overtaking Maneuvers for Multi-Vehicle Scenarios," in *2019 18th European Control Conference (ECC)*, Jun. 2019, pp. 744–749. doi: 10.23919/ECC.2019.8796013.

[9] G. K. McMillan, "Industrial Applications of PID Control," in *PID Control in the Third Millennium: Lessons Learned and New Approaches*, R. Vilanova and A. Visioli, Eds., in Advances in Industrial Control. , London: Springer, 2012, pp. 415–461. doi: 10.1007/978-1-4471-2425-2_14.

[10] V. Vartika, S. Singh, S. Das, S. K. Mishra, and S. S. Sahu, "A Review on Intelligent PID Controllers in Autonomous Vehicle," in *Advances in Smart Grid Automation and Industry 4.0*, M. J. B. Reddy, D. Kr. Mohanta, D. Kumar, and D. Ghosh, Eds., in Lecture Notes in Electrical Engineering. Singapore: Springer, 2021, pp. 391–399. doi: 10.1007/978-981-15-7675-1_39.

[11] M. Breivik and T. I. Fossen, "Applying missile guidance concepts to motion control of marine craft," *IFAC Proceedings Volumes*, vol. 40, no. 17, pp. 349–354, Jan. 2007, doi: 10.3182/20070919-3-HR-3904.00061.

[12] M.-W. Park, S.-W. Lee, and W.-Y. Han, "Development of lateral control system for autonomous vehicle based on adaptive pure pursuit algorithm," in *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*, Oct. 2014, pp. 1443–1447. doi: 10.1109/ICCAS.2014.6987787.

[13] A. AbdElmoniem, Y. T. Afif, S. A. Maged, M. Abdelaziz, and S. Hammad, "Adaptive Pure-Pursuit Controller Based on Particle Swarm Optimization (PSO-Pure-Pursuit)," in *2021 16th International Conference on Computer Engineering and Systems (ICCES)*, Dec. 2021, pp. 1–6. doi: 10.1109/ICCES54031.2021.9686157.

[14] S. Thrun *et al.*, "Stanley: The Robot That Won the DARPA Grand Challenge," in *The 2005 DARPA Grand Challenge: The Great Robot Race*, M. Buehler, K. Iagnemma, and S. Singh, Eds., in Springer Tracts in Advanced Robotics. , Berlin, Heidelberg: Springer, 2007, pp. 1–43. doi: 10.1007/978-3-540-73429-1_1.

[15] A. AbdElmoniem, A. Osama, M. Abdelaziz, and S. A. Maged, "A path-tracking algorithm using predictive Stanley lateral controller," *International Journal of Advanced Robotic Systems*, vol. 17, no. 6, p. 1729881420974852, Nov. 2020, doi: 10.1177/1729881420974852.

[16] A. Hasanzadeh, C. S. Edrington, B. Maghsoudlou, and H. Mokhtari, "Optimal LQR-based multi-loop linear control strategy for UPS inverter applications using resonant controller," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, Dec. 2011, pp. 3080–3085. doi: 10.1109/CDC.2011.6161192.

[17] N. Lehtomaki, N. Sandell, and M. Athans, "Robustness results in linear-quadratic Gaussian based multivariable control designs," *IEEE Transactions on Automatic Control*, vol. 26, no. 1, pp. 75–93, Feb. 1981, doi: 10.1109/TAC.1981.1102565.

[18] H. Li, P. Li, L. Yang, J. Zou, and Q. Li, "Safety research on stabilization of autonomous vehicles based on improved-LQR control," *AIP Advances*, vol. 12, no. 1, p. 015313, Jan. 2022, doi: 10.1063/5.0078950.

[19] R. Bellman, "A Markovian Decision Process," *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, 1957.

[20] S. Yang, J. Li, J. Wang, Z. Liu, and F. Yang, "Learning Urban Navigation via Value Iteration Network," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2018, pp. 800–805. doi: 10.1109/IVS.2018.8500517.

[21] J. Chen, W. Zhan, and M. Tomizuka, "Autonomous Driving Motion Planning With Constrained Iterative LQR," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 2, pp. 244–254, Jun. 2019, doi: 10.1109/TIV.2019.2904385.

[22] B. Charlet, J. Lévine, and R. Marino, "On dynamic feedback linearization," *Systems & Control Letters*, vol. 13, no. 2, pp. 143–151, Aug. 1989, doi: 10.1016/0167-6911(89)90031-5.

[23] D.-C. Liaw and W.-C. Chung, "A feedback linearization design for the control of vehicle's lateral dynamics," *Nonlinear Dyn*, vol. 52, no. 4, pp. 313–329, Jun. 2008, doi: 10.1007/s11071-007-9280-8.

[24] M. Eghtesad and D. S. Necsulescu, "Experimental study of the dynamic based feedback linearization of an autonomous wheeled ground vehicle," *Robotics and Autonomous Systems*, vol. 47, no. 1, pp. 47–63, May 2004, doi: 10.1016/j.robot.2004.02.003.

[25] A. Bertipaglia, M. Alirezaei, R. Happee, and B. Shyrokau, "Model Predictive Contouring Control for Vehicle Obstacle Avoidance at the Limit of Handling." arXiv, Aug. 13, 2023. doi: 10.48550/arXiv.2308.06742.

[26] K. Shibata, N. Shibata, K. Nonaka, and K. Sekiguchi, "Model Predictive Obstacle Avoidance Control for Vehicles with Automatic Velocity Suppression using Artificial Potential Field," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 313–318, Jan. 2018, doi: 10.1016/j.ifacol.2018.11.050.

[27] P. Wang, C.-Y. Chan, and A. de La Fortelle, "A Reinforcement Learning Based Approach for Automated Lane Change Maneuvers," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2018, pp. 1379–1384. doi: 10.1109/IVS.2018.8500556.

[28] C. Lamini, S. Benhlima, and A. Elbekri, "Genetic Algorithm Based Approach for Autonomous Mobile Robot Path Planning," *Procedia Computer Science*, vol. 127, pp. 180–189, Jan. 2018, doi: 10.1016/j.procs.2018.01.113.

[29] N. Yu, C. Wang, F. Mo, and J. Cai, "Dynamic environment path planning based on Q-learning algorithm and genetic algorithm," *Journal of Beijing University of technology*, vol. 43, no. 7, Art. no. 7, 2017.

[30] H. Bae, G. Kim, J. Kim, D. Qian, and S. Lee, “Multi-Robot Path Planning Method Using Reinforcement Learning,” *Applied Sciences*, vol. 9, no. 15, Art. no. 15, Jan. 2019, doi: 10.3390/app9153057.