NTNU
Kunnskap for en bedre verden

Department of Engineering Cybernetics

TTK4551 - Engineering Cybernetics, Specialization Project

Adaptive Model Predictive Control and Path Planning for an Autonomous Car

Author:
Erik Thallaug Fagerli

Supervisor:
Thor Inge Fossen
6th June 2022

Abstract

Autonomous vehicles are getting more attraction and with that comes further development of control strategies. One of these controls is vehicle control. Vehicle control plays the role of assuring comfort and safety when guiding the vehicle. In this paper, the study of model predictive control (MPC) to control lateral movement of a 2 degree of freedom (DOF) bicycle model is researched. The controller uses a adaptive MPC to control the lateral deviation and relative steering angle with respect to desired trajectory. The controller is tested in a double lane change scenario. The system is simulated with the help of MATLAB, Simulink and Model Predictive Control Toolbox. Results show that the controller is effective as long as the speed does not drop to below four m/s. At this speed, the system becomes unstable.

Sammendrag

Autonome kjøretøy får større og større oppmerksomhet, og med det følger nye kontrollstrategier. En av disse er kontroll av baneplanlegging. Baneplanlegging forsikrer om at kjøretøyet følger en bane så sikkert og komfortabelt som mulig. I denne rapporten, blir "model predictive control" (MPC) studert for å kontrollere lateral bevegelse av en to frihetsgrad dynamisk sykkelmodell. Kontrollstrategien som blir brukt bruker en adaptiv MPC to å kontrollere lateral avvik og relative gir vinkel i forhold til ønsket bane. Kontrollstrategien er implementert i MATLAB og Simulink med hjelp av Model Predictive Control Toolbox. Resultatene viser at kontrolleren er effektiv til så lenge hastigheten ikke er under fire m/s. Ved denne hastighetene begyner systemet å bli ustabilt.

Preface

This paper is written in conjunction with the course TTK4551 - Engineering Cybernetics, Specialization Project. The course serves as an introduction to writing a master's thesis and how it should be structured. Throughout the course, extensive knowledge in current problems is obtained, good knowledge in related topics, and good knowledge about the relevant scientific literature.

I want to express my gratitude to Prof. Thor I. Fossen, supervisor of this paper, for supporting me with his knowledge and availability. I would also like to thank DNV FuelFighter for allowing me to be part of their team and giving me this problem to solve.

Contents

Figures

Chapter 1

Introduction

1.1 Motivation

As autonomous cars are becoming a more significant part of society's infrastructure, competitions around the globe are being held to see who can produce the best autonomous car. One of these competitions is the "Shell Eco-marathon". "Shell Eco-marathon" is a global academic program focused on energy optimization of cars [1]. NTNU has been part of this competition since 2008 with their group "DNV FuelFighter" [2]. DNV FuelFighter consists of several subgroups, such as mechanical, electrical, software, design, and has, as of 2018, created the subgroup autonomous. The autonomous group focuses on the autonomous part of the vehicle. With the combination of ROS (Robot Operating System, [3]), sensor, perception, path planning, and control, the car will be able to drive without the help of humans. Figure 1.1 shows how the different subsystems within the autonomous group connect. The car's position and surroundings are fetched into SLAM to create a map of the environment, then a goal is created, which is fed into the path planning algorithm. The path planning algorithm creates a suitable path with a reference speed and angle. The speed and angle are then transformed into voltages and fed into steering angle and steering brake control. Due to the ongoing pandemic, DNV FuelFighter has not been able to create or compete with the autonomous car fully but hopes to have a functional autonomous car by the end of June 2022.

This paper serves as a pre-phase for a master's thesis, where Adaptive Model Predictive Control (AMPC) will be researched, and its feasibility of controlling the lateral movement of the car will be studied. AMPC is a control strategy that originated from model predictive control (MPC) and is a control strategy for computing desired input based on previous output. The AMPC will be tested by seeing how it performs during a double lane change maneuver at varying speeds. If the AMPC is found feasible, it will later be implemented in ROS and simulated in Gazebo [4]. If AMPC shows promise in the Gazebo simulation, it will be used in the DNV FuelFighters car.

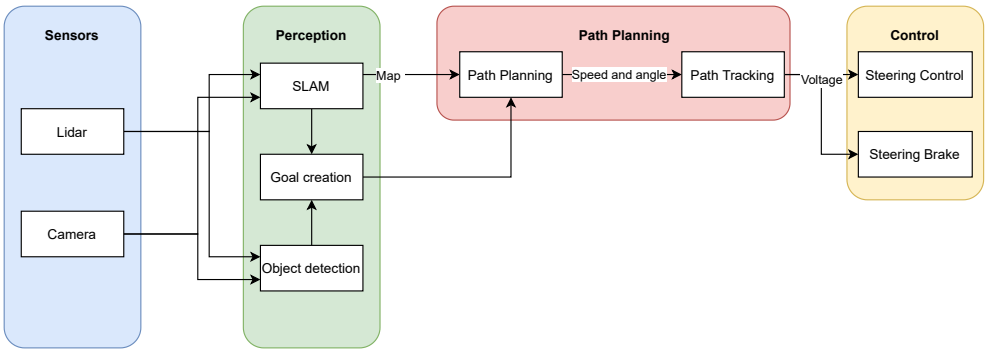

Figure 1.1: Description of how the different subsystems within the autonomous car work with each other

1.2 Previous work

The use of MPC to control lateral movement has already been researched on. Research done by [5] shows that even though the vehicle model can be linear time-varying (LTV), LTV-MPC can be an excellent tool for solving these kinds of problems. Their research shows that the vehicle follows the given trajectory well under varying speed conditions. [6] talks about the usage of Robust Adaptive MPC for lane-keeping, where it shows that the vehicle follows a circular arc curvature and manages to stay within the constraints even with model biases. [7] focuses on how well a nonlinear MPC and linear MPC handles a trajectory tracking problem for an unmanned ground vehicle. It concludes that the nonlinear MPC solves the tracking problem best in terms of performance but requires more computational time. However, the linear MPC handles the problem reasonably well and has reduced computational time. A study done by [8] compares the linear time-invariant (LTI) model and LTV model of a vehicle and sees how they perform when integrated with an AMPC. It concludes that they both perform pretty well, but the AMPC with the LTV model achieves better lane-keeping performances. [9] compares MPC to PD controller and concludes that MPC achieves better performance under the same conditions as the PD controller. [10] compares the use of the kinematic bicycle model and dynamic bicycle model with MPC at various velocities. It concludes that the kinematic bicycle model works well in straight lines and at very low speeds, but when lateral acceleration increases, the kinematic model becomes unreliable. The dynamic bicycle model works well at higher speeds and when lateral acceleration increases.

1.3 Problem formulation

As mentioned previously, in this paper, we will look at how AMPC controls an autonomous car's lateral movement. When looking at Figure 1.1, AMPC is implemented in the "Path Planning" subsystem. Here the AMPC will be fed the de-

sired trajectory and, based on that trajectory, spit out the desired speed and angle needed for the autonomous car to reach that trajectory. From previous work, AMPC has shown promise when handling varying speeds and does well at trajectory following. We will be looking at how the AMPC handles various velocities in a double lane change maneuver. A double lane change maneuver consists of sharp turns, thus making it more vulnerable to instability. We will also be looking at how the AMPC handle low velocities when used with a dynamic bicycle model.

For the sake of simplicity and limitations, we will in this paper assume that desired double lane change maneuver trajectory is already computed, the model describing the autonomous car is perfect, and the longitudinal velocity is constant. This is not applicable in real life but will indicate if AMPC is something we will want to pursue further.

1.4 Research questions

Throughout this paper, the following questions will be answered

- What type of vehicle model will be used to represent lateral motion?
- How does AMPC handle double lane change maneuvers?

1.5 Organization of report

The paper is structured as following:

- **Chapter 2**: A two-degree of freedom linear vehicle model with a linear tire model is presented and used to describe the lateral movement of the vehicle.
- **Chapter 3**: An overview of what MPC is and how it is used to predict future inputs. Later AMPC is presented and how this is implemented into the problem.
- **Chapter 4**: The AMPC is evaluated based on how well it performs a double lane change maneuver.
- **Chapter 5**: Conclusion and future work is presented.

Chapter 2

Automotive Bicycle and Tire Models

In this chapter, the vehicle's mathematical model is derived. The control of the vehicle is affected by how complex the vehicle model is. If a model is complex, it may give accurate predictions, but it may be computational heavy. On the other side, a simple model is less computational heavy but may not give accurate predictions. In this paper, a linearized two-degree of freedom dynamic bicycle model is used to define the lateral dynamics of the vehicle. The lateral dynamics will then control the error in lateral displacement and yaw angle. For simplicity, the longitudinal speed is held constant.

2.1 Dynamic bicycle model

Modeling autonomous cars "perfectly" can be time-consuming and complex. It is easier to use simple models for simulation purposes that can indicate how the autonomous car will behave. In [10] two simple models are discussed, a kinematic bicycle model and a dynamic bicycle model. It concludes that the kinematic bicycle model is best suited for very low speeds (0-6 m/s), and dynamic is best for higher speeds. Since DNV Fuelfighter will mainly operate at higher speeds than six m/s, the testing model will be the dynamic bicycle model. At high vehicle speeds, the velocity at each wheel is no longer in the direction of the wheel [11]. A dynamic model for lateral motion needs to be implemented in these situations. This is depicted in Figure 2.1. The degrees of freedom are represented as lateral position **y** and yaw angle ψ. Lateral motion is usually represented in three degrees of freedom, sway, roll, and yaw. In this paper, the roll is neglected. It is assumed that the car does not reach high enough speeds to roll.

The forces applied to the autonomous car along the y-axis are described using Newton's second law of motion.

$$ma_y = F_{yf} + F_{yr}, \tag{2.1}$$

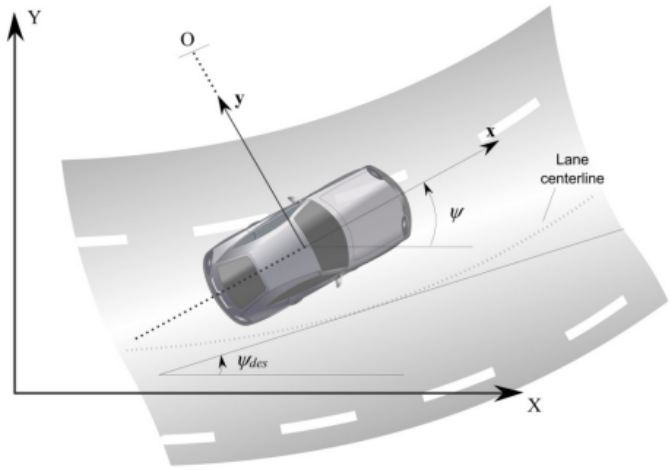

Figure 2.1: Lateral vehicle dynamics (source: [11])

where a_y is the acceleration of the vehicle at the center of gravity in the y axis, m is the mass of the car, F_{yf} and F_{yr} are the lateral tire forces of the front and rear wheels. How F_{yf} and F_{yr} are derived is explained in Section 2.2.1. The terms that contribute to an acceleration in a_y are the acceleration $\ddot{y}$ due to movement along the y-axis and the centripetal acceleration $V_x\dot{\psi}$, where V_x is the longitudinal velocity and $\dot{\psi}$ is the yaw rate. This gives the equation

$$a_y = \ddot{y} + V_x\dot{\psi}. \tag{2.2}$$

By inserting (2.2) into (2.1), the equation for lateral motion of the vehicle is obtained.

$$m(\ddot{y} + \dot{\psi}V_x) = F_{yf} + F_{yr}. \tag{2.3}$$

The equation for yaw dynamics is obtained by looking at the moment balance about the z-axis.

$$I_z\ddot{\psi} = l_f F_{yf} - l_r F_{yr} \tag{2.4}$$

where l_f and l_r is the distance from the front and rear tire to the center of gravity and I_z is the yaw moment of inertia. Now the equation for later motion and yaw dynamics are presented. The next thing is to derive the tire forces.

2.2 Tire models

When slip ratio and sideslip angle are large, the tire forces behave nonlinearly and need to be modeled to represent the vehicle's dynamics. Such situations happen in, e.g., racing. When slip ratio and sideslip are small, the tire forces can have linear behavior as shown in Figure 2.2. Since the vehicle will operate under "normal speeds", a linear tire model will be good enough.

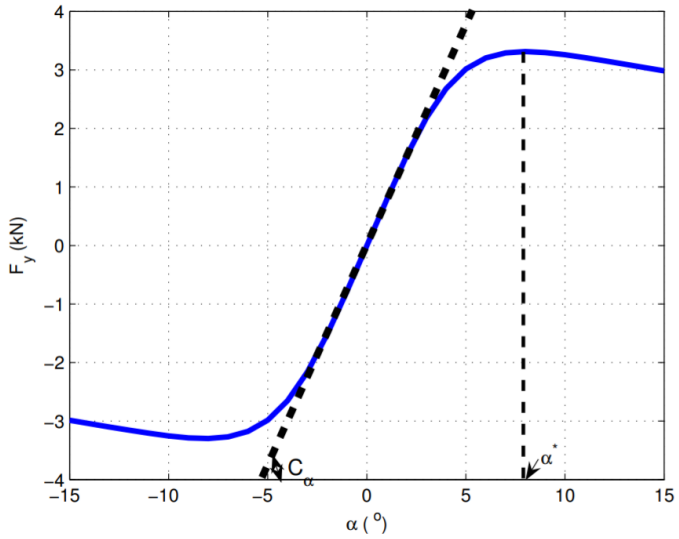

Figure 2.2: Lateral forces on tire when sideslip angle is low (source: [12])

2.2.1 Linear tire model

Experimental results have shown that for small sideslip, the lateral forces on the tires are proportional to the sideslip angle [11]. The sideslip angle is the angle between the orientation of the tire and the velocity vector of the wheel, as shown in Figure 2.3. With these assumptions, the lateral forces on the tires can be expressed as

$$F = C_\alpha \alpha, \tag{2.5}$$

where C_α is the cornering stiffness and α is the sideslip angle. The sideslip angle is given by

$$\alpha_f = \delta - \theta_{Vf}. \tag{2.6}$$

δ is the angle of the front wheel, and θ_{Vf} is the angle of the velocity vector compared to the longitudinal axis. The sideslip angle of the rear tire is

$$\alpha_r = -\theta_{Vr} \tag{2.7}$$

where θ_{Vf} is the angle of the rear wheel compared to the longitudinal axis.
By inserting (2.6) and (2.7) into (2.5) we get:

$$F_{yf} = 2C_{\alpha f}(\delta - \theta_{Vf}) \tag{2.8}$$
$$F_{yr} = 2C_{\alpha r}(-\theta_{Vr}) \tag{2.9}$$

The reason for the factor 2 in (2.8) and (2.9) is to take into account two front- and rear wheels. The equations for the angle of the wheels compared to the longitudinal axis are given by

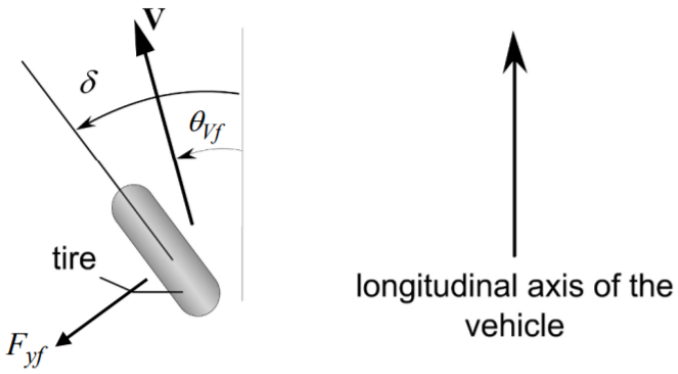

Figure 2.3: Sideslip angle and lateral force at front tire (source: [11])

$$\tan(\theta_{Vf}) = \frac{V_y + l_f\dot{\psi}}{V_x} \tag{2.10}$$

$$\tan(\theta_{Vr}) = \frac{V_y + l_r\dot{\psi}}{V_x} \tag{2.11}$$

By using small angle approximation and noticing that $V_f = \dot{y}$ we get:

$$\theta_{Vf} = \frac{\dot{y} + l_f\dot{\psi}}{V_x} \tag{2.12}$$

$$\theta_{Vr} = \frac{\dot{y} + l_r\dot{\psi}}{V_x} \tag{2.13}$$

2.3 Vehicle model

By inserting (2.8), (2.9), (2.12) and (2.13) into (2.3) and (2.4), and represent it in state-space form we get

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{2C_{\alpha f}+2C_{\alpha r}}{mV_x} & 0 & -V_x - \frac{2C_{\alpha f}l_f - 2C_{\alpha r}l_r}{mV_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{2l_fC_{\alpha f} - 2l_rC_{\alpha r}}{I_zV_x} & 0 & -\frac{2l_f^2C_{\alpha f} + 2l_r^2C_{\alpha r}}{I_zV_x} \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{2C_{\alpha f}}{m} \\ 0 \\ \frac{2l_fC_{\alpha f}}{I_z} \end{bmatrix} \delta \tag{2.14}$$

(2.14) represent the lateral dynamics of the linearized dynamic bicycle model. One thing to note is that the model has a singularity. This is when $V_x = 0$, this means that the vehicle model is only valid for $V_x > 0$. (2.14) is the model that will be used in MATLAB. The values for the parameters are listed below. These values are taken from [13].

- $m = 1575$, Mass of the car (kg).
- $I_z = 2875$, Yaw moment of inertia (kgm^2).
- $l_f = 1.2$, Longitudinal distance from center of gravity to the front tires (m).
- $l_r = 1.6$, Longitudinal distance from the center of gravity to the rear tires (m).
- $C_f = 19000$, Cornering stiffness of the front tires (N/rad).
- $C_r = 33000$, Cornering stiffness of the rear tires (N/rad)

Chapter 3

Model Predictive Control

This section presents an introduction to what model predictive control is and how it works. Later we introduce adaptive model predictive control and how it works compared to standard MPC. Lastly, the AMPC objective function is formulated and explains what we minimize.

3.1 Overview of model predictive control

Model predictive control is an advanced control technique used for multi-variable control problems [14] that can be traced back to the 1960s, but the interest in the field started in the 1980s [15]. The concept of MPC is predicting future output values based on an accurate model of the process. The prediction is made at each time step. This is useful because it changes its prediction if a disturbance occurs. For example, if a gust of wind hits the car, the position might change since the model does not account for such disturbances. This is not a problem for the MPC since it will change the future control actions based on the car's new position. Figure 3.1 shows an example of a single input single output control. At the sampling instant k, the MPC calculates a set of future inputs. The future inputs are calculated such that the future outputs $\hat{y}$ reach the target most optimally. The further the control horizon M and prediction horizon P are, the more optimal the trajectory becomes, but comes with the cost of computational time. The prediction horizon P is often limited by sensors and how far they can see. The objective of an MPC is summarized by [16]:

- Prevent violating constraints (input and output).
- Reach optimal output set points.
- Prevent excessive input movement.
- Control as many process variables as possible when a sensor or actuator is not available.

Figure 3.2 shows how the MPC block works. The reference is fed into the MPC. The MPC uses an optimizer to calculate the different scenarios and choose the optimal one. This is then fed into the plant, which is then looped back to be

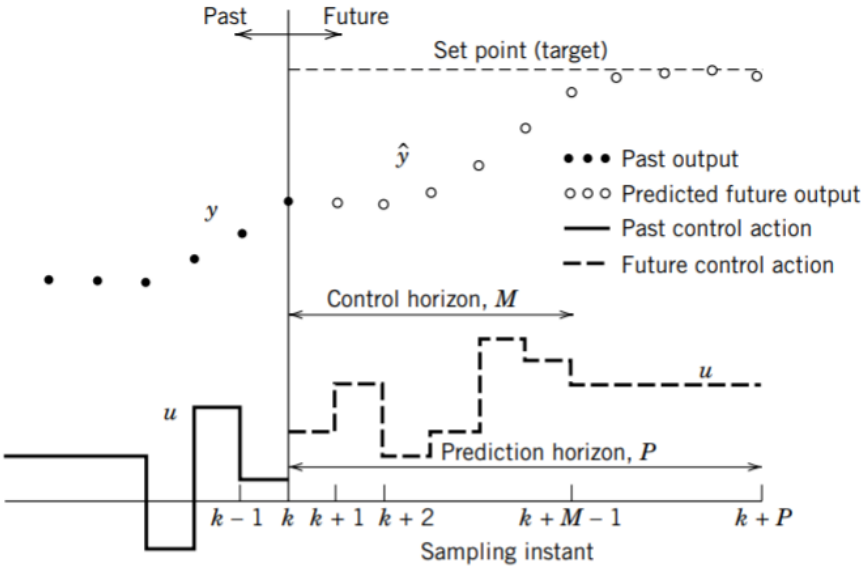

Figure 3.1: SISO MPC (Source: [14])

compared to the reference.

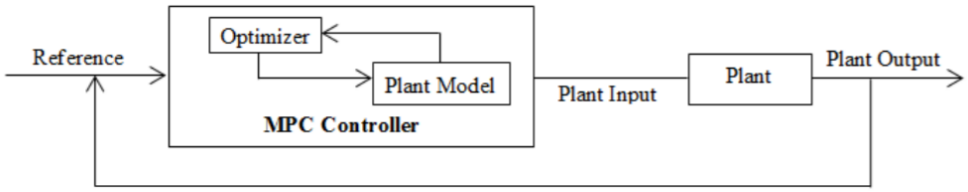

Figure 3.2: MPC diagram (source: [17])

3.2 Adaptive model predictive control

The MPC controller mentioned earlier is bad at handling changing dynamics, since its internal plant is constant. In a situation where the car increases or decreases its speed, the accuracy of the internal plant will be affected, and the performance might become unstable. To counter this, Adaptive MPC can be used. AMPC handles varying dynamics by changing the operating point at each time step. By doing so, the AMPC can handle various speeds. Since it changes operating point, the model must be represented as a discrete-time state-space model

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k), \end{aligned} \tag{3.1}$$

where A is the state matrix, B is the input matrix, and C is the output matrix.

3.2.1 Problem formulation

The AMPC is used to deal with the lateral motion of the autonomous car. The steering angle is controlled so that it manages to stay as close to the given trajectory as possible. The AMPC objective is to minimize error in lateral deviation (e_1) and relative angle (e_2) as depicted in Figure 3.3. This is done by computing the most optimal angle. To achieve this, the AMPC minimizes the cost function

$$\begin{aligned} \min_u J = \sum_{j=1}^{N} ||y_p(k+j|k) - y_{ref}(k+j|k)||_{Q_y} + \sum_{j=0}^{M-1} ||u(k+j|k)||_{R_u} \\ s.t. \quad x(k+j+1|k) = Ax(k+j|k) + Bu(j+k|k) \\ x(k|k) = x(k) \\ y(k+j|k) = Cx(k+j|k) \\ u_{lower} \leq |u(k+j|k) \leq u_{upper} \end{aligned} \tag{3.2}$$

where u is the manipulated variable, in this case the angle of the front wheel δ. Q_y and R_u represent the weight matrices. y_{ref} is the desired reference vector and y_p is the predicted output vector. The state vector $y_p = [e_1\ e_2]^\top$ while $y_{ref} = [0\ 0]^\top$. $\dot{e_1}$ and $\dot{e_2}$ are given by the equations

$$\dot{e_1} = V_x e_2 + V_y \tag{3.3}$$

$$\dot{e_2} = \dot{\psi} - \dot{\psi}_{ref}, \tag{3.4}$$

where

$$\dot{\psi}_{ref} = V_x \kappa. \tag{3.5}$$

κ is the road curvature. In this paper we assume that the road curvature is already known.

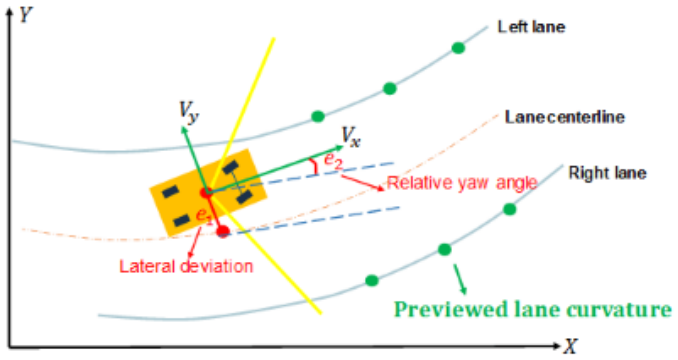

Figure 3.3: Lateral deviation and relative yaw angle (source: [6])

The predicted output $y(k)$ can be computed if we know the following values

- Present output measurement $y(k|k) = y(k)$
- Input $u(k)$
- Set of predicted input values $u(k+j),\ j = 0, 1, 2..., N$.

The state can then be defined as:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ x(k+2) &= Ax(k+1) + u(k+1) \\ &= A^2x(k) + AB(k) + Bu(k+1) \\ &\vdots \\ x(k+N) &= A^N x(k) + A^{N-1}Bu(k) + A^{N-2}Bu(k+1) + ... \\ &\quad + A^{N-M}Bu(k+M-1) \end{aligned} \tag{3.6}$$

The output can be defined as:

$$\begin{aligned} y(k) &= Cx(k) \\ y(k+1) &= Cx(k+1) \\ &\vdots \\ y(k+N) &= Cx(k+N) \end{aligned} \tag{3.7}$$

By defining u and y as:

$$\tilde{y}(k) = \begin{bmatrix} y(k+1) \\ y(k+2) \\ y(k+3) \\ \vdots \\ y(k+N) \end{bmatrix}, \tilde{u}(k) = \begin{bmatrix} u(k) \\ u(k+1) \\ u(k+2) \\ \vdots \\ u(k+M-1) \end{bmatrix} \tag{3.8}$$

we can rewrite the prediction output as:

$$\tilde{y}(k) = \tilde{A}x(k) + \tilde{B}\tilde{u}(k), \tag{3.9}$$

where

$$\tilde{A} = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^N \end{bmatrix},\ \tilde{B} = \begin{bmatrix} CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ CA^2B & CAB & CB & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{N-1}B & CA^{N-2}B & CA^{N-3}B & \cdots & CA^{N-M}B \end{bmatrix}. \tag{3.10}$$

This control strategy minimizes actual trajectory compared to desired trajectory by minimizing lateral deviation and yaw angle.

Chapter 4

Results and Discussion

The double lane change maneuver is simulated in MATLAB/Simulink with the help of the Model Predictive Control toolbox [18]. A double lane change maneuver is shown in Figure 4.1, where the car switches between three lanes. Using this maneuver when simulating is because it includes sharp turns, which are more difficult to optimize for than straight maneuvers. The MATLAB code and Simulink diagram can be found in Appendix A and B. Since the car will not only be moving at high velocities, the scenario is simulated at 3, 5, 10, and 20 m/s. The results are shown in Figure 4.2 and 4.3. From Figure 4.2 it looks like the AMPC does not work since there are no changes in steering angle, but this is proven wrong in Figure 4.3. The reason for instability when simulating at three m/s may be because of singularity in the model. As mentioned in Section 2.3, (2.14) has a singularity when $V_x = 0$. The closer V_x is to zero, the higher the chance of instability may happen. When simulating at different velocities, the threshold between stability and instability was around 3-4 m/s. This is further proven in Figure 4.4. Here it shows that the steering angle starts to oscillate when it reaches the double lane change maneuver and does not stop. A stability analysis may be useful to prove instability at these velocities, but this was not done due to time constraints.

One thing that is important to note from Figure 4.3 is that the car enters the double lane change maneuver at different periods of the simulation. When the simulation runs at five m/s, it starts to steer later compared to the other velocities. What we can see from Figure 4.3 is that the car manages to follow the trajectory reasonably well with a maximum lateral deviation of ±0.1m and a relative yaw angle maximum of 3 degrees. We see that at 20 m/s, the car does a sharp turn, something that is not desirable in the real world. The reason for this is that the longitudinal velocity is kept constant, thus meaning that when the car reaches the double lane change maneuver, it does not slow down but rather speeds up since it now moves in both x- and y-axis. Nevertheless, even with high speeds, the car does not become unstable.

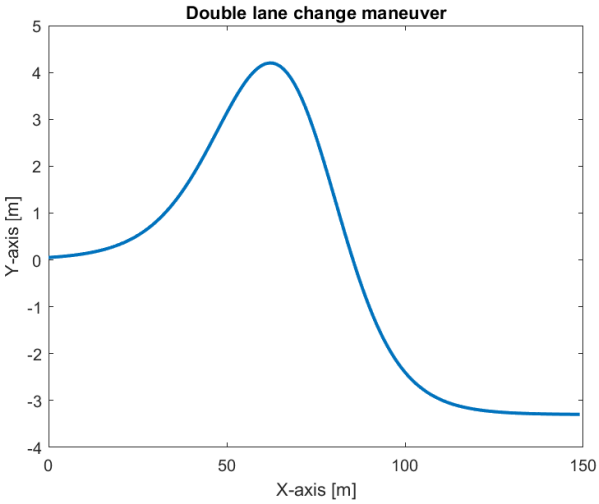

Figure 4.1: A double lane change maneuver

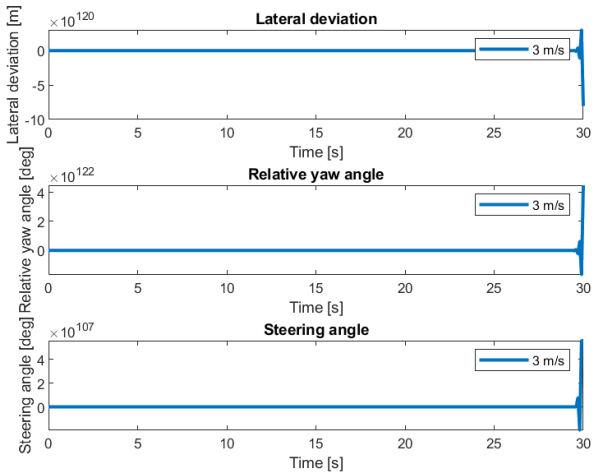

Figure 4.2: Simulated result of double lane change maneuver at 3 m/s. It shows no change in controls until the very last second where it becomes unstable.

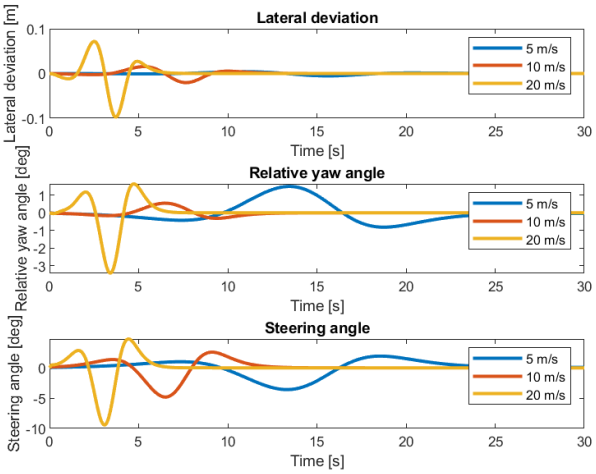

Figure 4.3: Simulated results of a double lane change maneuver at 5, 10 and 15 m/s. At the various speeds the lateral deviation and relative yaw angle eventually goes to zero.

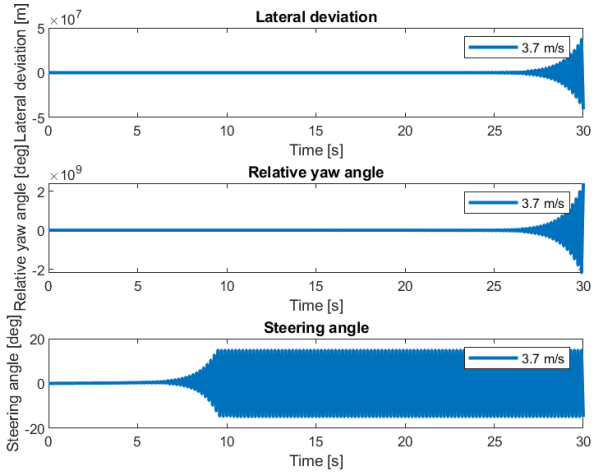

Figure 4.4: Simulated results of double lane change maneuver at 3.7 m/s. The system becomes unstable and starts oscillating

Chapter 5

Conclusion and Future Work

In this paper, A model for lateral motion and yaw movement was presented. A two-degree of freedom dynamic bicycle model with a linear tire model was introduced. An Adaptive Model Predictive Control was designed to control the autonomous car in a double lane change scenario. The adaptive controller minimized the error in lateral deviation and relative yaw angle, making the car follow the given trajectory well. The adaptive controller was simulated in MATLAB and Simulink at different longitudinal speeds. The simulations showed that the autonomous car held the lateral deviation and relative yaw angle within acceptable ranges. It was shown that AMPC is applicable to handle the lateral movement of the car at different speeds, but when reaching as low speeds as 3-4 m/s, the system becomes unstable, and the AMPC does not behave as desired.

As the dynamic model becomes unstable when reaching low speeds, one thing to improve for further studies is the model. The model used in this paper was chosen to keep simplicity and computation expenses as low as possible. By introducing a more complex model that does not have singularities, we will better compare how the car will behave when comparing simulations to real-life scenarios but may come with the cost of computational time. As mentioned in [10], the kinematic bicycle model is better suited at low speeds. A switching algorithm between the dynamic bicycle model and the kinematic bicycle model may be an alternative to handle both low and high speeds.

After finding a more suitable model for lateral dynamics, a model that describes longitudinal dynamics should be researched. As mentioned in chapter 4, the velocity increases when the autonomous car enters a turn. This is not desirable since it is customary to brake when entering a turn. This can be implemented by looking at the road's curvature and finding the desired velocity based on curvature, comfort, and constraints.

Testing on DNV Fuelfighters real car is time-consuming. It would be wise to simulate it in Gazebo, as it gives a better understanding of how the AMPC works in real life. If the AMPC functions as intended, it can later be implemented into the car for final testing. However, if AMPC is not found suitable, another algorithm that is similar to AMPC can be researched; multi-parametric quadratic program-

ming (mp-QP) [19]. Mp-QP works the same as AMPC as it is a control algorithm that computes a manipulated variable from a linear process model. The difference is that all the on-line computation can be moved off-line [20]. This makes the controller less computationally demanding, something that may be necessary.

Bibliography

[1] S. Eco, *Shell eco-marathon: Brilliant minds coming together to help build a lower carbon world*. [Online]. Available: https://www.makethefuture.shell/en-gb/shell-eco-marathon, (accessed: 30.09.2021).

[2] FuelFighter, *History*. [Online]. Available: https://www.fuelfighter.no/history, (accessed: 30.09.2021).

[3] *Ros*, https://www.ros.org/, (accessed: 2021-12-14).

[4] *Gazebo*, http://gazebosim.org/, (accessed: 2021-12-14).

[5] Y. Xu, B. Chen, X. Shan, W. Jia, Z. Lu and G. Xu, 'Model predictive control for lane keeping system in autonomous vehicle,' in *2017 7th International Conference on Power Electronics Systems and Applications - Smart Mobility, Power Transfer Security (PESA)*, 2017, pp. 1–5. DOI: 10.1109/PESA.2017.8277758.

[6] M. Bujarbaruah, X. Zhang, E. Tseng and F. Borrelli, 'Adaptive mpc for autonomous lane keeping,' Feb. 2018.

[7] E. Kayacan, W. Saeys, H. Ramon, C. Belta and J. M. Peschel, 'Experimental validation of linear and nonlinear mpc on an articulated unmanned ground vehicle,' *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 5, pp. 2023–2030, 2018. DOI: 10.1109/TMECH.2018.2854877.

[8] B.-C. Chen, B.-C. Luan and K. Lee, 'Design of lane keeping system using adaptive model predictive control,' in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, 2014, pp. 922–926. DOI: 10.1109/CoASE.2014.6899436.

[9] M. Samuel, M. Mohamad, M. Hussein and S. M. Saad, *Lane keeping maneuvers using proportional integral derivative (pid) and model predictive control (mpc)*, 2021.

[10] J. Kong, M. Pfeiffer, G. Schildbach and F. Borrelli, 'Kinematic and dynamic vehicle models for autonomous driving control design,' Jun. 2015, pp. 1094–1099. DOI: 10.1109/IVS.2015.7225830.

[11] R. Rajamani, *Vehicle Dynamics and Control*, ser. Mechanical Engineering Series. 2012, ISBN: 9781489985460.

[12] S. Baslamisli, I. Polat and I. Kose, 'Gain scheduled active steering control based on a parametric bicycle model,' Jul. 2007, pp. 1168–1173. DOI: 10.1109/IVS.2007.4290276.

[13] Mathworks, *Lane keeping assist system using model predictive control*. [Online]. Available: https://se.mathworks.com/help/mpc/ug/lane-keeping-assist-system-using-model-predictive-control.html, (accessed: 23.11.2021).

[14] D. E. Seborg, D. A. Mellichamp and T. F. Edgar, *Process Dynamics and Control*, Third, ser. Wylie Series in Chemical Engineering. John Wiley & Sons, 2011, ISBN: 9780470646106. [Online]. Available: http://www.worldcat.org/isbn/9780470646106.

[15] M. Morari and J. H. Lee, 'Model predictive control: Past, present and future,' *Computers Chemical Engineering*, vol. 23, no. 4, pp. 667–682, 1999, ISSN: 0098-1354. DOI: https://doi.org/10.1016/S0098-1354(98)00301-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0098135498003019.

[16] J. Qin and T. Badgwell, 'A survey of industrial model predictive control technology,' *Control engineering practice*, vol. 11, pp. 733–764, Jul. 2003. DOI: 10.1016/S0967-0661(02)00186-7.

[17] A. Reda, A. Bouzid and J. Vásárhelyi, 'Model predictive control for automated vehicle steering,' *Acta Polytechnica Hungarica*, vol. 17, pp. 163–182, Sep. 2020. DOI: 10.12700/APH.17.7.2020.7.9.

[18] *Model predictive control toolbox*, https://se.mathworks.com/products/model-predictive-control.html, (accessed: 2021-12-17).

[19] P. Tøndel, T. A. Johansen and A. Bemporad, 'An algorithm for multi-parametric quadratic programming and explicit mpc solutions,' *Automatica*, vol. 39, no. 3, pp. 489–497, 2003, ISSN: 0005-1098. DOI: https://doi.org/10.1016/S0005-1098(02)00250-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109802002509.

[20] A. Bemporad, M. Morari, V. Dua and E. N. Pistikopoulos, 'The explicit linear quadratic regulator for constrained systems,' *Automatica*, vol. 38, no. 1, pp. 3–20, 2002, ISSN: 0005-1098. DOI: https://doi.org/10.1016/S0005-1098(01)00174-1. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109801001741.

Appendix A

MATLAB Code

```
%% Sample time and simulation duration

Ts = 0.1; %Sample time
T  = 15;  %Simulation duration

%% Parameters

m = 1575;   % Mass (kg)
Iz = 2875;  % Yaw moment of inertia (Kg*m^2)
lf = 1.2;   % Longitudinal distance from center of gravity to the front tires
lr = 1.6;   % Longitudinal distance from center of gravity to the rear tires
Cf = 19000; % Cornering stiffness of the front tires (N/rad)
Cr = 33000; % Cornering stiffness of the rear tires (N/rad)

Vx = 5;    % Longitudinal velocity (m/s)

%% State-space model

A = [-(2*Cf+2*Cr)/m/Vx        ,  -Vx-(2*Cf*lf-2*Cr*lr)/m/Vx ;   %Lateral
     -(2*Cf*lf-2*Cr*lr)/Iz/Vx , -(2*Cf*lf^2+2*Cr*lr^2)/Iz/Vx];  % Yaw
B = [2*Cf/m, 2*Cf*lf/Iz]'; % Steering
C = eye(2);
G = ss(A,B,C,0);

%% Prediction Horizon

PredictionHorizon = 10;

time = 0:0.1:25;
[md, Xref, Yref] = getCurvature_test(Vx,time); % Double lane change manuever.

%% Constraints

u_min = -0.5; % Steering angle [rad/s]
u_max = 0.5;  % Steering angle [rad/s]
```

```
function [md, Xref, Yref] = getCurvature_test(Vx,time)
% Get previewed curvature from desired X and Y positions for LKA
%
% Inputs:
%   Vx: longitudinal velocity
%   time: time vector
%
% Outputs:
%   md: previewed curvature
%   Xref: Desired x values
%   Yref: Desired y values

% Desired X position
Xref = Vx*time;


% Desired Y position
z1 = (2.4/50)*(Xref-27.19)-1.2;
z2 = (2.4/43.9)*(Xref-56.46)-1.2;
Yref = 8.1/2*(1+tanh(z1)) - 11.4/2*(1+tanh(z2));

% Desired curvature
DX = gradient(Xref,0.1);
DY = gradient(Yref,0.1);
D2Y = gradient(DY,0.1);
curvature = DX.*D2Y./(DX.^2+DY.^2).^(3/2);

% Stored curvature
md.time = time;
md.signals.values = curvature';
```

Appendix B

Simulink Diagram

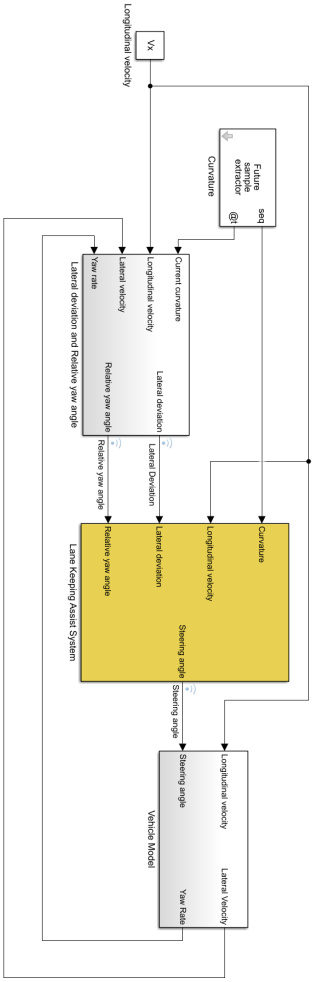

Figure B.1: Simulink diagram