Autonom omkörning i motorvägsmiljö med Model Predictive Control

SIMEON ANDERSSON

KIM STRINNHOLM
Autonomous Highway Overtaking With Model Predictive Control

SIMEON ANDERSSON

KIM STRINNHOLM
Sammanfattning


An important step towards making road transportation safer around the world is the development of autonomous vehicles. In this paper a controller for performing autonomous overtaking at highway speeds using Model Predictive Control (MPC) is designed. The MPC framework is designed and tested in a simulated environment in order to evaluate the performance of the controller. Four different MPC frameworks are developed for generating paths for autonomous overtaking and a Proportional Integral Derivative (PID) controller is formulated for a general comparison. The four MPC frameworks all plan trajectories by introducing constraints; they differ in the way they formulate said constraints. From the simulations we conclude that MPC is a better controller choice than PID for the application of controlling autonomous vehicles because of the usability of MPC, even though they might be equally fast. The benefits and drawbacks of the MPC implementations and their characteristics are discussed, and we conclude that the preferred implementation is a linear sloped edge dynamic constraint where a disallowed area around the leading vehicle is explicitly defined outside of the MPC framework.
Autonomous Highway Overtaking With Model Predictive Control
Simeon Andersson and Kim Strinnholm

Abstract—An important step towards making road transportation safer around the world is the development of autonomous vehicles. In this paper a controller for performing autonomous overtaking at highway speeds using Model Predictive Control (MPC) is designed. The MPC framework is designed and tested in a simulated environment in order to evaluate the performance of the controller. Four different MPC frameworks are developed for generating paths for autonomous overtaking and a Proportional Integral Derivative (PID) controller is formulated for a general comparison. The four MPC frameworks all plan trajectories by introducing constraints; they differ in the way they formulate said constraints. From the simulations we conclude that MPC is a better controller choice than PID for the application of controlling autonomous vehicles because of the usability of MPC, even though they might be equally fast. The benefits and drawbacks of the MPC implementations and their characteristics are discussed, and we conclude that the preferred implementation is a linear sloped edge dynamic constraint where a disallowed area around the leading vehicle is explicitly defined outside of the MPC framework.

I. INTRODUCTION

Society today is extremely dependent on its transportation infrastructure and road congestion can be a big problem in many of our cities. The dense traffic means that travelling on highways is associated with risks and one of the most dangerous situations is overtaking another vehicle [1]. Since 90% of all accidents can be attributed to human errors there might be a lot to gain by avoiding human drivers at all [2]. Autonomous vehicles are rapidly becoming more popular and their development has taken a prominent position at the forefront of technological development. A large number of companies focus on the development of self-driving cars and there have been many ground breaking announcements coming from the private sector within the area [3].

The possible benefits of autonomous vehicle isn’t isolated to decreasing the amount of road fatalities but also in decreasing congestion since if you drive in Stockholm during peak-hours you will spend on average an extra 126 hours per year in traffic because of congestion [4]. That is a huge cost for society and one that autonomous cars potentially could radically decrease. According to Tientrakool et al. [5], if all vehicles driving on a highway at 100 km/h were autonomously controlled, equipped with sensors and communication devices for vehicle to vehicle communication the road capacity would increase about 3.7 times.

When a human driver prepares to perform an overtaking the person predicts a path the vehicle would take in order for the overtaking to be safe and controlled. An autonomous vehicle would need to do the same prediction of the near future path, and model how the current decision would effect the future to perform a safe overtaking. One control framework that is able to do that is Model Predictive Control (MPC), which this study will focus on. MPC is the leading advanced control strategy that has been used for about ten years in the industry because MPC algorithms can control large scale systems with many variables and because of the easy expansion of constraints [6]. An MPC controller uses a plant model to predict the future state of the plant for a certain input control value to the plant. The controller predictions are time discrete and it predicts on a finite receding horizon, i.e. a finite number of steps, $N$, into the future [7]. In this work $N$ will be referred to as the prediction horizon. Fundamentally what an MPC controller does, in every prediction step, is determine a control vector that minimizes a given cost function considering a given set of constraints. The main advantage with MPC is that it can optimize the problem in real time while modelling future positions and taking other vehicles into account. Model predictive control is a very popular type of controller due to its usability, easy expansion to more complex problems and ability to deal with several constraints [8]. An MPC controller can be composed of a planning layer and a controlling layer. In this implementation the planning layer will directly control the vehicle as the first element of the input vector will be used as control input to the vehicle.

The objective with this study is to formulate said MPC framework in different ways whom all will be able to perform an overtaking at highway speeds, and then to compare the different results. All formulations will have the same basic structure; a cost will be added to the cost function when the distance from the main lane increase, and the position of the vehicle that will be overtaken, the leading vehicle, will be formulated as constraints on the position of the overtaking vehicle, the ego vehicle. Only the ego vehicle will be actively controlled. The difference between methods will be in the way the constraints are formulated. The main result of this work will be the comparison between the speed of computation between the methods and their performance in terms of stability and speed. The MPC controllers will be implemented in Matlab using the Yalmip toolbox. As well as the different MPC formulations a PID controller will be created for the sake of comparison. Questions that will be discussed in this study is advantages and disadvantages with the both controllers as well as whether MPC is better then PID control.

The difference between this study and similar studies within the area is the focus on different implementation of the constraints and comparisons between the performance and calculation time.

To limit the scope of this study the following assumptions
and simplifications are made:
- Both vehicles are maintaining constant speeds.
- The leading car is driving with a constant direction
- The road is straight and consists of two lanes. The leading car is driving in the right lane and the left lane will be empty and free for the ego car to use.
- The ego car has perfect knowledge of its surroundings. It knows the exact position and speed of itself and the leading vehicle.

A. Scenario

In Fig. 1 the problem environment can be seen. The position of the ego car is described by \((x, y)\), and the position of the leading car is described by \((x_L, y_L)\). The width of each lane is 5 m, and both cars are 1.8 m wide and 4.7 m long. The relative starting distance between the ego vehicle and the leading vehicle is 50 m.

![Fig. 1. Describes the environment the problem is based on where \((x, y)\) describe the ego car position, and \((x_L, y_L)\) describe the leading car position.](image)

The safety aspect is very important to take into consideration when developing autonomous vehicles. The ego vehicle needs to have a certain specified distance to the leading vehicle, both in \(x\) and \(y\) direction. The safety distance in the \(x\) direction will depend on, among others, the velocity \(v\) of the ego car and the friction factor \(f\) between the wheels and the road. The safety distance in the \(x\) direction can be described as

\[
x_s = \frac{v^2}{250f} \tag{1}
\]

according to P. Greibe [9], where \(v\) is the velocity and \(f\) is the friction factor between the wheels and the road. With wheels in good condition on dry asphalt the friction factor is typically 0.8, but can get down to 0.1 on ice. In this study \(f\) will be assumed to be equal to 0.8. This distance \(x_s\) will determine when the controller will commence the overtaking. The safety distance in the \(y\) direction, \(y_s\), will be set to 4.5. All distances related to the vehicles are defined relative to the center-points of the vehicles.

II. SINGLE-TRACK VEHICLE DYNAMICS

The popular single-track vehicle model, also known as a bicycle model, is used to approximate the dynamics of the four wheeled vehicle. The vehicle model is the same as described by Paden et al. [6]. The vehicle kinematics are there described as

\[
\dot{x} = v \cos(\theta), \tag{2a}
\]

\[
\dot{y} = v \sin(\theta), \tag{2b}
\]

\[
\dot{\theta} = \frac{v}{l} \tan(\delta). \tag{2c}
\]

The velocity is denoted \(v\), \(l\) is the length of the vehicle, \(\theta\) is the angle of the rear wheel relative to the direction of the road and \(\delta\) is the angle of the front wheel relative the angle of the rear wheel, i.e. the wheel angle. Because the front and rear wheels have an angle relative each other they will not have the same velocity. Therefore it is valid to describe the motion of the vehicle by either the front or the rear wheel. In this work however, the velocity and direction of the car is described by the motion of the rear wheel. In this work the angle \(\delta\) is the controllers input to the plant model and \(\theta\) is the state controlled by the controllers. The model assumes that no slip will occur, i.e. there is an infinitely large friction coefficient between the tires and roadway. In Fig. 2 the vehicle model configuration is illustrated.

![Fig. 2. Vehicle model configuration.](image)

A bicycle model makes the description of the car more manageable without making an approximation of reality that is not accurate [6]. A more detailed high-fidelity model can be used to describe the response of the vehicle at the expense of computational difficulty and more computationally demanding simulations. Though it is in this work not considered to be necessary to use a more demanding, but accurate, model.

III. PID-CONTROLLER

A PID controller was designed as a reference to evaluate the results from the MPC. One of the goals is to answer the question whether it is worth using the more complex MPC controller instead of a more simple one as the PID and therefor the PID is used to evaluate if the MPC is motivated to use. The PID controller is described with the equation

\[
\delta(t) = K_c e(t) + K_I \int_{t_0}^{t} e(\tau) d\tau + K_d \frac{de(t)}{dt} \tag{3}
\]

where \(\delta(t)\) is the control signal, \(e(t)\) is the error \(y_r(t) - y(t)\) and the parameters \(K_c, K_I, K_d\) [10]. The PID parameters are tuned with Simulink in order to be optimized for a fair comparison. Fig. 3 shows the structure of the system where
F(s) is a PID-controller and G(s) is the plant model. To be able to tune the PID parameters in Simulink eq. (2) needs to be described in the frequency domain. Laplace transforming the input control signal $u = \delta$ and the controlled $y$ position results in

$$U(s) = \Delta(s), \quad Y(s) = Y.$$  \hspace{1cm} (4)

Laplace transforming eq. (2b) and (2c), and linearizing around $\delta = 0$ and $\theta = 0$ we get

$$sY(s) = \mathcal{L}\{v\sin(\theta(t))\} \approx \mathcal{L}\{v\theta(t)\} = v\Theta(s), \quad s\Theta(s) = \mathcal{L}\{\frac{v}{l}\tan(\delta(t))\} \approx \mathcal{L}\{\frac{v}{l}\delta(t)\} = \frac{v}{l}\Delta(s). \quad \hspace{1cm} (5a, b)$$

Combining eq. (4) and (5) we get the following equation that describe the vehicle dynamics in the frequency domain:

$$Y(s) \approx \frac{v^2}{l} \frac{1}{s^2} U(s) \implies G(s) \approx \frac{v^2}{l} \frac{1}{s^2}. \hspace{1cm} (6)$$

The transfer function $G(s)$ can be divided into several parts shown in Fig. 4 and this structure can then be formulated in Simulink to acquire the optimal parameters for the PID controller.

![Diagram](image)

**Fig. 4.** $G(s)$, the vehicle models transfer function, in more detail.

### IV. MPC CONTROLLERS

Model Predictive Control is a method for controlling a plant process while satisfying a number of constraints. The advantage in this application is when the system detects a vehicle ahead of it, the controller predicts a future path while taking a number of constraints into consideration. The MPC controller is based on minimizing a cost function and this cost function can have several terms based on different factors. The constraints and the cost function is then processed in an optimizer. In this section the state function, cost function and constraint formulations are described.

**A. Control-oriented model**

To be able to control the vehicle with a MPC controller a vehicle model in the form of a state function must be formulated. The state function can be defined, according to [11], as

$$z(k+1) = Az(k) + Bu(k) + K,$$  \hspace{1cm} (7)

$$z(k) = \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} \quad \text{and} \quad u(k) = \delta(k). \quad \hspace{1cm} (8)$$

According to the assumption in section 1 that $v$ is constant, and using eq. (2) we can formulate the equations

$$x(k+1) = x(k) + v\cos(\theta(k))\Delta t \approx x(k) + v\Delta t, \quad \hspace{1cm} (9a)$$

$$y(k+1) = y(k) + v\sin(\theta(k))\Delta t \approx y(k) + v\theta(k)\Delta t, \quad \hspace{1cm} (9b)$$

$$\theta(k+1) = \theta(k) + \frac{v}{l}\tan(\delta(k))\Delta t \approx \theta(k) + \frac{v}{l}\delta(k)\Delta t, \quad \hspace{1cm} (9c)$$

where we have linearized around $\theta = 0$ and $\delta = 0$. We perform this linearization to simplify the computations when performing the MPC predictions. The angles $\theta$ and $\delta$ are limited in magnitude since the problem is formulated around driving at highway speeds where large movements aren’t desirable and thus these linearizations does not have a major impact on the accuracy of the MPC predictions. Rewriting eq. (9) we get the following equation on the same form as eq. (7):

$$\begin{bmatrix} x(k+1) \\ y(k+1) \\ \theta(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & v\Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} + \begin{bmatrix} \frac{v}{l}\Delta t \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \delta(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{v\Delta t}{l} \end{bmatrix} \begin{bmatrix} \bar{u}(k) \\ \bar{u}(k) \\ \bar{u}(k) \end{bmatrix}. \hspace{1cm} (10)$$

**B. MPC Formulation**

According to M. Cannon [7] the cost function can be formulated as

$$J(k) = \sum_{i=0}^{N} \left( \|\bar{z}(k+i)\|_Q^2 + \|\bar{u}(k+i)\|_R^2 \right) \hspace{1cm} (11)$$

where the notation $(k+i)$ refers to the predicted state at time $(k+i)$. The predictions are performed at time $k$. In the application in this study the cost function increase when the ego car deviates from the reference values $r(k)$. The cost function is therefore rewritten as

$$J(k) = (\bar{z}(k) - \bar{r}(k))^T \bar{Q}(\bar{z}(k) - \bar{r}(k)), \hspace{1cm} (12)$$

$$\bar{Q} = \begin{pmatrix} Q & 0 \\ 0 & \ldots & Q & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{pmatrix}, \quad Q = \begin{pmatrix} 0 & 0 & 0 \\ 0 & q_{22} & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

$$\bar{r}(k) = \begin{pmatrix} r(k+1) \\ r(k+N) \end{pmatrix}, \quad \bar{z}(k) = \begin{pmatrix} z(k+1) \\ \cdots \end{pmatrix}.$$  \hspace{1cm} (13)

The formulation of the cost function in eq. (12) allows for more variables to be controlled than just the $y$ position. For the application in this study however only the $y$ position are explicitly controlled. Because of this the matrix $Q$ only have one nonzero element positioned in the center as shown in eq. (13). For simplicity when the value of $Q$ is referred to what is meant is the value of $q_{22}$. 

![Diagram](image)
The optimization problem is formulated as
\[
\min_{\mathbf{u}(k)} J(k) \quad (14a)
\]
\[\text{s.t} \quad -\frac{\pi}{8} \leq \delta(k) \leq \frac{\pi}{8} \quad \forall k, \quad (14b)\]
\[1.5 \leq y(k) \leq 8.5 \quad \forall k, \quad (14c)\]
\[
D_{\min} \leq \frac{\delta(k) - \delta(k-1)}{\Delta t} \leq D_{\max} \quad \forall k, \quad (14d)
\]
\[
\theta_{\min} \leq \theta(k) \leq \theta_{\max} \quad \forall k. \quad (14e)
\]
where \(D_{\min}, D_{\max}, \theta_{\min}\) and \(\theta_{\max}\) are scalar valued limits on the problem. The limits on \(\delta\) and \(y\) are physical limits originating from the described problem were the limit on \(\delta\) is set as an arbitrary limit on the wheel angle and the limit on \(y\) is set so that there is about 0.5 meters to the edge of the road from the side of the vehicle as a buffer zone. The undefined limits in eq. (14) will be analyzed during simulation and set after that analysis.

One inherent problem with MPC is that the optimization problems can, because of certain hard constraints, become infeasible [12]. The constraints are defined as hard constraints by default, which means that the constraint are not allowed to be violated. When the controller tries to minimize the cost function it might control the vehicle to balances on the edge of a constraint and if some sort of disturbance is introduced it might be impossible to avoid violating the constraint. If the constraint is hard then the problem will be infeasible and the optimizer will return no solution. For example, a disturbance might be a small numerical error in Matlab. To avoid the problem becoming infeasible the solution is to soften the constraint, which means that it is allowed to violate a constraint but then a large cost will be added to the cost function. To soften a constraint a slack variable \(s(k)\) is added to the constraint. If the constraint isn’t broken the value of the slack variable is zero. After introducing slack variables the cost function is rewritten as
\[
J(k) = (\bar{z}(k) - \bar{r}(k))^T Q (\bar{z}(k) - \bar{r}(k)) + \bar{s}(k)^T W \bar{s}(k), \quad (15)
\]
where
\[
\bar{s}(k) = \begin{pmatrix}
  s(k+1) \\
  \vdots \\
  s(k+N)
\end{pmatrix}, \quad s(k+i) = \begin{pmatrix}
  s_1(k+i) \\
  \vdots \\
  s_m(k+i)
\end{pmatrix},
\]
\[
W = \begin{pmatrix}
  w_1 & 0 & \cdots & 0 \\
  \vdots & \ddots & \vdots & \vdots \\
  0 & \cdots & w_N
\end{pmatrix}. \quad (16)
\]
The matrix \(W\) is an \(mN \times mN\) diagonal matrix where \(m\) is the number of slack variables introduced and \(N\) is the prediction horizon. The value of the scalar \(w_i\) is set according to the importance of each constraint [12]; if it is a critical constraint the value should be larger. In this study every \(w\) is set to 100 000.

C. Different overtaking approaches

The MPC controller can be implemented with many different methods for detecting other vehicles and how the trajectory is determined. A few of them have been implemented in order to get simulated comparisons between the different methods.

Since the MPC controller solves an optimization problem to perform the path planning the computational difficulty is of great importance in practical application, which is not the case for PID control to the same extent. That is why the time it takes for the MPC controller to perform the path planning will be compared between the methods. For simplicity the notation \((k)\) will be omitted in this section and unless a variable is explicitly said to be constant it is assumed to vary with \(k\).

1) Elliptical constraint: The first way to describe the constraint around the leading vehicle is to define the constraint as an elliptical shape around the leading car because it is intuitive. The constraint can be described with a single equation as
\[
1 \leq \frac{(x-x_L)^2}{x_s^2} + \frac{(y-y_L)^2}{y_s^2} + s \quad (17)
\]
where \(x_s, y_s\) are constants that describe the safety distances that need to be kept and \(s\) is a slack variable. In Fig. 5 the general shape of the constraint can be seen. When introducing the elliptical shape the constraint is no longer linear which makes the problem unsolvable with quadratic programming. For the elliptical constraint the Matlab solver \textit{fmincon} is used to solve to prediction problem instead. The non linearly makes the problem more complex and therefore the following methods are also implemented.

2) Mixed integer constraint: Next way to describe the leading vehicle as a constraint is by using mixed integers. Using mixed integer is a way of introducing an "if statement" to the MPC controller. The logical variables \(f_1, f_2\) and \(f_3\) can only take the value 0 or 1. The following implication criteria
between $\phi_1, \phi_2, x$, defined in Fig. 6, and the logical variables must always be met:

$$x \leq \phi_1 \Rightarrow f_1 = 0,$$
$$\phi_1 \leq x \leq \phi_2 \Rightarrow f_2 = 0,$$
$$\phi_2 \leq x \Rightarrow f_3 = 0.$$  \hspace{1cm} (18a, 18b, 18c)

Since one and only one of $f_i$ can be equal to 0 at any time the sum of the logical variables must for every time-step be equal to 2. I.e. the constraint is formulated as

$$f_1 + f_2 + f_3 = 2. \hspace{1cm} (19)$$

The variables $\phi_1$ and $\phi_2$ are defined as safety distances to the leading vehicle where $\phi_1 = x_L - x_s$ and $\phi_2 = x_L + x_s$. The constant variable $x_s$ is introduced in eq. (1). To be able to formulate the implications in a more mathematical way we introduce a large constant scalar $c$ and set the following inequalities as constraints:

$$\phi_1 - x + cf_1 \geq 0, \hspace{1cm} (20a)$$
$$x - \phi_1 + cf_2 \geq 0, \hspace{1cm} (20b)$$
$$x - \phi_2 + cf_3 \geq 0. \hspace{1cm} (20c)$$

The large scalar $c$ must be sufficiently large compared to the size differences between $\phi_1, \phi_2$ and $x$. In this application $c = 1000$ is sufficiently large.

Right now the values of the logical variables mean nothing to the MPC-controller. Similar constraints must be set on the $y$ position of the vehicle to make the ego vehicle change lane. Similar to above we formulate the implications as

$$f_1 = 0 \Rightarrow \gamma_1 \leq y, \hspace{1cm} (21a)$$
$$f_2 = 0 \Rightarrow \gamma_2 \leq y, \hspace{1cm} (21b)$$
$$f_3 = 0 \Rightarrow \gamma_1 \leq y, \hspace{1cm} (21c)$$

where $\gamma_1$ and $\gamma_2$ are defined in Fig. 6. From the implications we get the inequality constraints

$$y - \gamma_1 + cf_1 + s_1 \geq 0, \hspace{1cm} (22a)$$
$$y - \gamma_2 + cf_2 + s_2 \geq 0, \hspace{1cm} (22b)$$
$$y - \gamma_1 + cf_3 + s_3 \geq 0, \hspace{1cm} (22c)$$

where $s_1, s_2$ and $s_3$ are slack variables. With the mixed integer formulation of the constraint the optimization problem contains binary variables and the Matlab solver bnb must therefore be used.

3) Dynamic constraint: Lastly we propose an implementation of the constraint which we call dynamic constraint. Neither of the two aforementioned constraint formulations were linear so the final formulation was defined to be able to use quadratic programming to solve the optimization problem. The objective was to determine if the linear formulation is faster, and if so how much faster. With the dynamic constraint for every prediction step a vector $y_{min}$ is formulated as the smallest value of the constant $\gamma_2$ or the value

$$y_{min} = \gamma_2 + S(x_s - |x - x_L|) \hspace{1cm} (23)$$

where $S$ is constant and describes the steepness of the slope. The general shape of the constraint can be seen in Fig. 7. Two test are done with a dynamic constraint. One where the slope is set to a 22.5 degree angle, and one where the angle is set to 90 degrees, i.e. a straight edge similar to the constraint shown in Fig. 6. This vector is then implemented as an constraint as

$$y + s \geq y_{min} \hspace{1cm} (24)$$

where $s$ is a slack variable.

V. RESULTS

To obtain the results in this study a number of simulations have been performed. A nonlinear model based on the vehicle dynamics was created and used as base for the simulations. The car model itself was updated every 0.01 seconds to represent the real time performance while the controllers had a slower updating rate. As a tool the Matlab toolbox Yalmip was used to formulate the MPC controllers structures with constraints and objectives.

A. Choosing of PID parameters

After the simulation environment was completed a PID controller was formulated as a reference to the MPC controller. A reference to the MPC controller is needed to later be able to argue whether or not the MPC controller is an appropriate controller. For the comparison to be fair the PID need to be optimized so the parameters were tuned to perform as well as possible. Matlabs Simulink has been used to simulate the control system and the tool for tuning the PID parameters in Simulink has been used to design the PID controller.

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>PARAMETERS TO PID CONTROLLER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Value</td>
</tr>
<tr>
<td>$K_c$</td>
<td>0.13</td>
</tr>
<tr>
<td>$K_i$</td>
<td>0.001</td>
</tr>
<tr>
<td>$K_d$</td>
<td>0.035</td>
</tr>
</tbody>
</table>

Simulink the parameters $K_c$, $K_i$ and $K_d$ described in Table I were determined to be optimal and subsequently these values have been used for the PID controller. The sampling time for the PID controller was set to 0.1 seconds to be the same as the sampling time for the MPC controller.
B. Results from MPC and PID Comparison

One of the goals of this study was to compare characteristics of MPC and PID control and to answer which of the two is most suitable for autonomous overtaking. To first be able to fairly compare the speed of the two control methods they were both used as controllers in similar scenarios. In Figure 8 and Figure 9 the result from the PID simulation and different MPC simulations are shown. The controller objective of the simulations is to make the car change lane, i.e. move from \( y = 2.5 \) to \( y = 7.5 \). The other lane, position \( y = 7.5 \), is set as the reference value. When running the PID simulation the only constraint outlined in eq. (14) that was explicitly followed was the constraint on the wheel angle: \( \delta \). For a fair comparison between controllers, the MPC controller was also unconstrained except for the wheel angle, \( \delta \).

Simulations were run for different diagonal matrices \( Q \), sampling times \( T \) and prediction horizons \( N \) and the results are shown in Fig. 8 and Fig. 9. Since only one state, the \( y \) position, is explicitly controlled in this study, \( Q \) is a matrix with only zeros except at row 2, column 2. The value \( Q \) in Fig. 8 reflects that value. The prediction horizon is set to \( N = 10 \) and sampling time \( T = 0.1 \).

To determine what sampling time \( T \) is required for the controller several simulations were run and the result from those simulation are shown in Fig. 9. The prediction horizon is set so that the actual distance that is simulated in every step is the same and the value of \( Q \) is set to 1.

C. Result from Comparison Between Different Methods of MPC

Outlined in this study are four different ways of formulating the overtaking constraint. They are all based on the reference value being the right lane and the other vehicle formulated as constraints. For comparison they were all tested on the same scenario:

- \( v = 30 \text{ m/s}, \ v_L = 20 \text{ m/s} \)
- \( x_L(0) - x(0) = 50 \)
- \( Q = 1 \)
- \( N = 15, \ T = 0.3 \)
- basic constraints according to equation (25)

The results from these simulations can be seen in Fig. 10.

Fig. 9. MPC compared with PID, where \( N \) is the prediction horizon and \( T \) is the sampling time.

Fig. 10. Different overtaking constraint types.

Because of differences in computational complexity when solving the optimization problem the difference in computational speed is important to compare. As a reference the simulation covers twelve second in reality. For the MPC to be able to control the vehicle online on the current computer the calculation time should be less than 12 seconds. In Table II calculation times for the different methods is shown.

After experimental analysis the constraint, values considered
### TABLE II
**DIFFERENCE IN CALCULATION TIME BETWEEN METHODS**

<table>
<thead>
<tr>
<th>Constraint Type</th>
<th>Calculation Time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed Integer</td>
<td>3.22</td>
</tr>
<tr>
<td>Elliptical</td>
<td>20.02</td>
</tr>
<tr>
<td>Dynamic, Straight</td>
<td>1.78</td>
</tr>
<tr>
<td>Dynamic, Sloped</td>
<td>1.85</td>
</tr>
</tbody>
</table>

Appropriate for this application are:

\[
-\frac{\pi}{8} \leq \delta(k) \leq \frac{\pi}{8} \quad \forall k, \quad (25a)
\]

\[
1.5 \leq y(k) \leq 8.5 \quad \forall k, \quad (25b)
\]

\[
-0.05 \leq \frac{\delta(k) - \delta(k-1)}{\Delta t} \leq 0.05 \quad \forall k, \quad (25c)
\]

\[
-\frac{\pi}{4} \leq \theta(k) \leq \frac{\pi}{4} \quad \forall k. \quad (25d)
\]

These constraints were used when simulating the different methods of overtaking.

### VI. DISCUSSION

#### A. Analysis of MPC compared to PID

When comparing PID to MPC in Fig. 8 and Fig. 9 it can be seen that the two control methods are very similar in speed. PID control is slightly faster in the beginning, but has a bit more overshoot. Of course the overshoot can be traded for a slower controller but the overshoot is within acceptable limits. In Fig. 9 it is shown that MPC can be made faster at the expense of computational speed by having a shorter sampling time. Though we would argue that both methods are unreasonably fast; driving at 30 m/s the car changes lane in both cases in around one second. Because of the rapid movements at high speeds the bicycle model used is probably not accurate since it assumes no slip, i.e. an infinitely large friction coefficient, between the tires and roadway. For the model to be less inaccurate a constraint could be set on the maximum speed in the y-direction, or the constraint on the wheel angle could be reduced. The theoretical speed of the controllers are therefore in this application not considered a significant factor in choosing control method, but what is considered a major factor in choosing MPC over PID is the usability of MPC. By being able to add constraints the controller becomes very useful and easily adaptable to many different scenarios while PID control is limited to only following a reference trajectory, it’s not able to plan its trajectory by itself.

For less demanding applications the simplicity in the PID can however be preferred. PID doesn’t at all need the same amount of computational power as MPC does.

From the result in Fig. 8 it can be concluded that the value of the elements in the diagonal matrix \(Q\) does not have a major impact on the performance on the controller. Smaller values makes the lane change slower while very high values makes it behave quite oscillatory. With value 1 the controller seems to be optimal and that is the value used for the rest of the simulations.

When comparing how sampling time effects the results in Fig. 9 it can be concluded that a large sampling time can make the controller behave oscillatory. A smaller sampling time is therefore needed but the increased computations need to be considered.

#### B. Analysis of Different Methods of MPC

When comparing the different methods of formulating the constraints for the controller in Fig. 10 it can be argued that all methods are applicable; they all perform reasonably well. When comparing the mixed integer constraint and the straight edge dynamic constraint it is clear that they are nearly identical. They exhibit some oscillation when approaching a constant value but by adding the sloped edge to the dynamic constraint that oscillatory behaviour is decreased and the controller performs very well. Of the four constraint formulations the one that exhibits least oscillatory behaviour is the elliptical constraint formulation. That was to be expected since there are no sharp edges in the shape of the constraint. Though the problem with the elliptical constraint is its computational difficulty because of its non-linearity. It is shown in Table II that it takes more than 10 times as long to solve the optimization problem with the elliptical constraint as with the dynamic constraints. If computation time was not a factor an elliptical constraint could be a viable option. Though in general when driving at highway speeds the demand on reaction time and computational speed is high and a dynamic or mixed integer constraint would be preferred. A solution to the small problems with oscillation with the sloped edge constraint could be to predefine a smoother transitional curve between lanes.

During simulation it was apparent that the limits described in eq. (25) could have significant effects on the behaviour of the controller. If the controller exhibited oscillatory behaviour that effect could be diminished by decreasing the limits on the speed of which the wheel turned, described in eq. (25c). If the simulation showed that the controller resulted in a response with a significant overshoot that effect could be decreased by decreasing the limits on \(\theta\). However both of these changes contribute to making the resulting overtaking slower, but that might be preferred. Those limits should be set on a case to case basis dependent on traffic situation.

#### C. Future research

Model Predictive Control is an highly usable control method with several possible applications. This study has addressed the overtaking aspect of autonomous driving but there are many other autonomous driving scenarios that can be, and has been addressed with a model predictive controller, for example handling intersections, parking or avoiding fixed obstacles.

In this study a few assumptions and simplifications have been made that cannot be made if the MPC controller is to be implemented in an actual vehicle. Future research that could be done is to develop methods that allow the MPC controller to handle more than one leading vehicle. One of the more difficult would be to add vehicle driving in the opposite direction. Another difficulty to ad would be to have a model that can handle different non-constant speeds.

This study has only worked with a fixed reference track and having the other vehicles formulated as constraint. It might
be more advantageous to also use a dynamically changing reference track. A possible area for future work might be to compare such a solution with the results from this work. The bicycle model that has been used in this study is only applicable up to a certain speed, a more accurate model suitable for future work would be a more dynamic model which takes the friction factor into account.

VII. CONCLUSION

The conclusion drawn from this study is that Model Predictive Control is an efficient control method for performing an autonomous overtaking at highway speeds. MPC is preferred over PID control and there are several ways to implement MPC constraints to perform said overtaking. The preferred way to formulate the constraint considering performance and computational speed is with a sloped edge dynamic constraint. For less demanding applications the simplicity in the PID can however be preferred.

ACKNOWLEDGMENT

We would like to thank our supervisor Yuchao Li for all his guidance and the knowledge of control theory he has provided us. We would also like to thank project group B3A as well as the rest of context B for their cooperation throughout the project.

REFERENCES