DEGREE PROJECT IN TECHNOLOGY,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, *SWEDEN 2018*

Autonomous Overtaking Using Reachability Analysis and MPC

JOHN FRIBERG

FILIP KLAESSON

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ENGINEERING SCIENCES

EXAMENSARBETE INOM TEKNIK,
GRUNDNIVÅ, 15 HP

STOCKHOLM, *SVERIGE 2018*

Autonom omkörning med nåbarhetsanalys och MPC

JOHN FRIBERG

FILIP KLAESSON

KTH
SKOLAN FÖR TEKNIKVETENSKAP

Sammanfattning

Autonoma bilar är på frammarsch. När förare inte längre har kontroll över ratten är det avgörande att bilarna själva kan garantera säkerheten för alla trafikanter. Denna studie syftar till att utforma ett komplett styrsystem som kan utföra en säker omkörning. Omkörningen planeras med hjälp av ramverket för modell-prediktiv reglering. För att garantera säkerhet används nåbarhetsanalys. Slutligen utformas en modifierad proportionell regulator för att följa den planerade omkörningsvägen. Styrsystemet har implementerats i MATLAB och hela omkörningen har simulerats. Resultaten visar att det konstruerade styrsystemet utför omkörningen på en rak motorväg med två filer på ett säkert och framgångsrikt sätt.

Autonomous Overtaking Using Reachability Analysis and MPC

John Friberg and Filip Klaesson

***Abstract*—The era of autonomous cars is on the rise. As drivers lose control of the steering wheel, it is crucial that the cars themselves can guarantee safety for all traffic participants. This study aims to design a complete control system that can safely perform an overtaking maneuver. To guarantee safety of the maneuver, reachability calculations will be carried out and analyzed. The overtaking will be planned by using the model predictive control, MPC, framework. To complete the control system a modified proportional controller will be designed to track the planned path. The control system is implemented in MATLAB and the entire overtaking maneuver is simulated. The results show that the designed control framework successfully performs the overtaking on a straight two-lane highway in a safe manner.**

I. Introduction

Autonomous vehicles are anticipated to revolutionize the transportation system as we know it today. In the near future we will see self driving cars as a part of our everyday life. Among its many advantages are increased safety, higher fuel efficiency and faster transportation.

Although great advancements have already been made in the field, many tricky traffic scenarios has to be handled in order to guarantee safety. Features that handle some of these situations have already been launched on commercial cars, such as autonomous parking and emergency braking, but a lot of other features are still in development. One such feature, which handles a traffic situation that according to [1] accounts for 4-10% of accidents in traffic, is autonomous overtaking. Therefore, autonomous overtaking is of great importance to study.

As the overtaking maneuver is a particularly dangerous traffic situation, it is a top priority to ensure that an autonomous car is always able to reach a safe state, even if something unexpected would occur.

Many studies regarding trajectory planning of the overtaking scenario have already been carried out, such as [2] and [3]. These papers show that model predictive control is a sufficient tool. Equivalently, the safe state analysis has been researched thoroughly in [4] and [5], where reachability has been proven successful.

The main goal of this study is to design a complete control system which can safely perform an overtaking maneuver on a straight two-lane highway. The resulting controller will be implemented and simulated.

The remainder of this work is organized as follows. Section II covers the basic idea of our approach. Section III discloses the mathematical model and theory for the entire study. In Section IV, a brief description of the implementation of the theory described in Section III is given. All simulation results will be presented in Section V. These results will be further discussed in Section VI. Lastly, conclusions based on the discussion will be presented in Section VII.

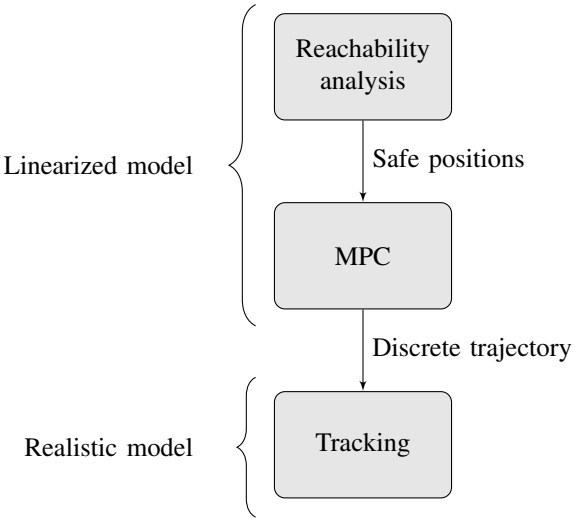

Fig. 1: Flowchart describing the basic idea of the approach.

II. Basic idea, Limitations and Assumptions

The approach that will be used to create the control system can be divided into three main parts; calculations of safe states using reachability analysis, trajectory planning using model predictive control and trajectory tracking.

In this study the overtaking maneuver will be carried out on a straight two-lane highway with two traffic participants. It will be assumed that the vehicle which is being overtaken stays in its initial lane. In an overtaking scenario it is also reasonable to assume that the speed of the car performing the overtaking is significantly higher than the speed of the vehicle being overtaken.

In order to ensure that no collision will occur, the first step is to find the positions on the road where the overtaking car can be but not the car being overtaken given their initial states. These positions are free to use as the travel path.

The next step is to find an optimal and safe trajectory which entirely lies within these positions. This is done with consideration to traffic rules and appropriate overtaking behavior. Such behavior includes accelerating before changing lane and keeping sufficient distance to the other vehicle during the whole maneuver.

This means that the entire overtaking maneuver will be planned, from the initial states of the two cars, before it is executed. The last part is to design a simple tracking controller that makes the car safely perform the planned overtaking based on the real nonlinear model. The basic idea of the approach is shown in Fig. 1.

III. Mathematical modeling

A. Modeling of vehicles

There are two different vehicles included in this study; the vehicle performing the overtaking maneuver, which will be referred to as the ego vehicle, and the vehicle that is being overtaken, which will be referred to as the other vehicle. In reality there is no physical distinction between these two vehicles and therefore they will be modelled in the same way. As the papers [2] and [6] have mentioned, an accurate way to mathematically model a car is through the bicycle model. The same notation as in Fig. 2 from [2] will be used in this paper.

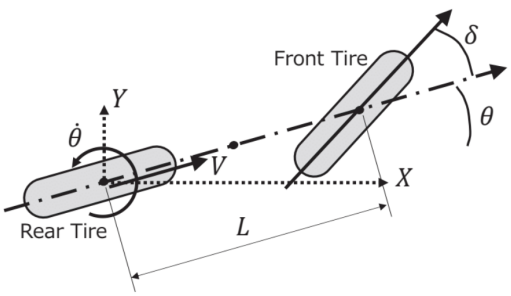

Fig. 2: Bicycle model used to model the cars. The notation will be used throughout this paper.

The dynamics of the car can be described as follows:

$$\dot{X} = V\cos\theta \tag{1}$$
$$\dot{Y} = V\sin\theta \tag{2}$$
$$\dot{\theta} = \frac{V}{L}\tan\delta \tag{3}$$

In order to carry out the calculations in this study, this system will be linearized around certain values on θ, V and δ. These values will be denoted θ_0, V_0 and δ_0. As the overtaking in this case takes place at a straight highway with high speeds, the angles involved will at all times be very small. This also motivates why the linearization will still model the system accurately. Therefore the following values and constraints will be used:

$$\theta_0 = 0 \text{ and } |\theta| \leq 10^\circ \tag{4}$$
$$\delta_0 = 0 \text{ and } |\delta| \leq 10^\circ \tag{5}$$

The speed of the cars will at all times be close to their initial speed and therefore V_0 is chosen as their initial speed. The numerical value will be chosen later in this section. In order to ensure that the linearization is a good approximation, the constraints on V_0 is given by:

$$0.9V_0 \leq V \leq 1.1V_0 \tag{6}$$

The linearized system can be described by the following equations:

$$\dot{X} = V \tag{7}$$
$$\dot{Y} = V_0\theta \tag{8}$$
$$\dot{\theta} = \frac{V_0}{L}\delta \tag{9}$$

From these equations the state vector x and the control input u for the cars are constructed.

$$x = [X, Y, \theta]^T \tag{10}$$
$$u = [V, \delta]^T \tag{11}$$

The linearized system can easily be discretized and can then be written as follows:

$$x(kh + h) = Ax(kh) + Bu(kh) \tag{12}$$

where h is the time step, k is a positive integer parameter and the matrices A and B are given below:

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & V_0 h \\ 0 & 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} h & 0 \\ 0 & 0 \\ 0 & \frac{V_0 h}{L} \end{bmatrix} \tag{13}$$

So far no distinction between the autonomous ego car and the other car has been done. However, the numerical values on V_0 and the constraints for the cars, both on the states and on the inputs, are still to be defined. These will be treated separately.

1) Constraints for the ego vehicle: First, it must be guaranteed that the car stays on the road. This gives a constraint on the Y-coordinate of the car. The middle of the highway will be defined as $Y = 0$. No such constraint can be enforced on the X-coordinate, however the car will begin in $X = 0$ and move forward, hence the state constraints of the ego vehicle can be written as:

$$X_{ego} \geq 0 \text{ m} \tag{14}$$
$$|Y_{ego}| \leq 5 \text{ m} \tag{15}$$
$$|\theta_{ego}| \leq 10^\circ \tag{16}$$

The value on V_0^{ego} and the input constraints also need to be defined. The ego vehicle drives initially at a speed of 90 km/h, therefore the following values are obtained:

$$V_0^{ego} = 90 \text{ km/h} \tag{17}$$
$$81 \text{ km/h} \leq V_{ego} \leq 99 \text{ km/h} \tag{18}$$
$$|\delta_{ego}| \leq 10^\circ \tag{19}$$

2) Constraints for the other vehicle: The state constraints for the other vehicle is similar to the ego vehicle's, with the difference being that the other vehicle will be assumed to stay in the right lane and start 40 m ahead of the ego vehicle.

$$X_{other} \geq 40 \text{ m} \tag{20}$$
$$-5 \text{ m} \leq Y_{other} \leq 0 \text{ m} \tag{21}$$
$$|\theta_{other}| \leq 10^\circ \tag{22}$$

The linearization speed, V_0^{other} is chosen as the initial speed of the other vehicle. The following values are obtained:

$$V_0^{other} = 70 \text{ km/h} \tag{23}$$

$$63 \text{ km/h} \leq V_{other} \leq 77 \text{ km/h} \tag{24}$$

$$|\delta_{other}| \leq 10^\circ \tag{25}$$

The aforementioned models of the cars is used to plan the safe trajectory. When this trajectory is tracked in a simulation, a more physically correct model is needed. As it is impossible to change the velocity and the front tire angle instantly, acceleration and front tire angular velocity is used as control inputs to the ego car. The state and input become:

$$x = [X, Y, \theta, V, \delta]^T \tag{26}$$

$$u = [a, \dot{\delta}]^T \tag{27}$$

where a is the acceleration and $\dot{\delta}$ is the front tire angular velocity. Note that in this case a linearization of the model must not be used, as this is supposed to simulate reality.

The constraints on a are set with passenger convenience in mind, thus being bounded by:

$$|a| \leq 4 \text{ m/s}^2 \tag{28}$$

No such constraint is imposed on $\dot{\delta}$ as δ is already bounded to be very small.

The modelling and constraints of the cars are now finished. The next step is to create a framework from where calculations of the safe states of the cars can be conducted.

B. Reachable Safe States

In order to find a safe overtaking maneuver, it must be guaranteed that the ego vehicle can not at any time in the future be in the same position as the other vehicle. First of all, the reader will be reminded of the definitions of Minkowsky sum, difference and intersection of sets before the definition of a reachable set and the corresponding safe set will be presented.

Definition 1: The Minkowsky sum of the sets A and B is defined by: $A + B = \{x + y | x \in A, y \in B\}$.

Definition 2: The set difference between the sets A and B is defined by: $A \setminus B = \{x | x \in A, x \notin B\}$

Definition 3: The intersection of the sets A and B is defined by: $A \cap B = \{x | x \in A \wedge x \in B\}$

Definition 4: $R_i(kh)$ is the polyhedron containing all reachable state for vehicle i at time step k.

Definition 5: The safe set for the ego vehicle at time step k is defined by:

$$S_{ego}(kh) = R_{ego}(kh) \setminus R_{other}(kh)$$

i.e. the part of the reachable set for the ego vehicle that does not intersect with the reachable set of the other vehicle.

In order to take into account the dimensions of the cars, which in this study are chosen as 4.7m×1.8m, the reachable set of states at time step $k = 0$ is defined as the set of positions covered by the car when it is centered at the initial position. For the ego car, with the initial position chosen as $(X_0^{ego}, Y_0^{ego}) = (2.35, -2.5)$ this gives:

$$\begin{aligned} R(0)_{ego} = \{&X, Y, \theta | \, 0 \text{ m} \leq X \leq 4.7 \text{ m}, \\ &-3.4 \text{ m} \leq Y \leq -1.6 \text{ m}, \theta = 0^\circ\} \end{aligned}$$

The initial set of the other vehicle is then given by the following set.

$$\begin{aligned} R(0)_{other} = \{&X, Y, \theta | \, 40 \text{ m} \leq X \leq 44.7 \text{ m}, \\ &-3.4 \text{ m} \leq Y \leq -1.6 \text{ m}, \theta = 0^\circ\} \end{aligned}$$

The set of possible states and the set of possible inputs are limited by the constraints of the vehicles, which are given in the previous subsection. These sets are denoted as L and U.

$$L_{ego} = \{X, Y, \theta | \, X \geq 0, |Y| \leq 5, |\theta| \leq 10^\circ\} \tag{29}$$

$$U_{ego} = \{V, \delta | \, 81 \text{ km/h} \leq V \leq 99 \text{ km/h}, |\delta| \leq 10^\circ\} \tag{30}$$

$$L_{other} = \{X, Y, \theta | \, X \geq 0, -5 \leq Y \leq 0, \theta \leq 10^\circ\} \tag{31}$$

$$U_{other} = \{V, \delta | \, 63 \text{ km/h} \leq V \leq 77 \text{ km/h}, |\delta| \leq 10^\circ\} \tag{32}$$

The reachable set of states of vehicle i at time step k can be calculated through the following algorithm:

$$R(kh + h)_i = (A_i R(kh)_i + B_i U_i) \cap L_i \tag{33}$$

where A and B are the matrices defined in Eq. 13. The reachable safe states of the ego vehicle at each time step can then be calculated from Definition 5, i.e. from the set difference of $R(kh)_{ego}$ and $R(kh)_{other}$. As it is now known where the ego car safely can be positioned at each time step, it is time to move on to the trajectory planning part.

C. Trajectory planning

As the approach is to plan the entire trajectory from the initial state of the cars, this section consists of essentially two different tasks. The first is the task of computing the next control input signal to the ego vehicle, given the current state and the safe states from the previous subsection. The second part is to calculate the new state given the aforementioned input, virtually move the car accordingly and then recalculate the safe states based on the new position. This procedure will be repeated until the overtaking trajectory is fully planned.

To plan the trajectory, model predictive control, MPC, framework and the notation from [7] is used. The strength of MPC is that one can optimize the current input, while taking future states into account. In order to do this, a cost function is used to assign costs to deviations of certain reference values of the current and future state and input values. The number of future states taken into account is called the prediction horizon. As the cost function assigns penalties to a state, the cost function should be minimized at all times, under the constraint that the car is always positioned in the safe states calculated in the previous subsection. The optimization will generate an input sequence over the prediction horizon, where only the

calculated input for the current state will be used as input to the system. The procedure will be demonstrated below.

The first step in the MPC framework is to define a predicted input and output sequence at time step k. These are defined as follows:

$$\mathbf{u}(k) = [u(k|k), u(k+1|k), ..., u(k+N-1|k)]^T \quad (34)$$

$$\mathbf{x}(k) = [x(k+1|k), ..., x(k+N|k)]^T \quad (35)$$

where $x(k+i|k)$ and $u(k+i|k)$ denotes the state and input vectors at time $k+i$ predicted at time k. N is the prediction horizon, which in this paper is chosen as $N = 5$.

The next step is to design a cost function, $J(\mathbf{x}(k), \mathbf{u}(k))$. In this study the cost function will be on the form below.

$$J = \sum_{i=1}^{N} (x_i - x_i^{ref})^T Q (x_i - x_i^{ref}) + \quad (36)$$

$$(u_i - u_i^{ref})^T R (u_i - u_i^{ref}) \quad (37)$$

where $x_i = x(k+i|k)$ and $u_i = u(k+i-1|k)$ are defined to simplify the notation and Q and R are square matrices called weights. The reference values, x_i^{ref} and u_i^{ref} represents the desired values of the state and input. Before the overtaking maneuver starts, the ego vehicle must stay in the middle of the right lane, i.e. $y^{ref} = -2.5$ m. When the ego vehicle gets sufficiently close to the other vehicle the ego vehicle should move to the middle of the left lane, i.e. $y^{ref} = 2.5$ m. The change in y^{ref} is triggered by a condition on the distance between the ego car and the closest possible position of the other car. In this study, if the distance of X-coordinate between the ego vehicle and the closest possible position of the other vehicle is less than 10 m then $y^{ref} = 2.5$ m, otherwise $y^{ref} = -2.5$ m.

To make a more time efficient overtaking, the ego car is desired to speed up before starting the overtaking maneuver. Therefore the same condition as before will be used, with the distinction that the change in speed is triggered at the distance of 20 m. If the distance is closer than 20 m the ego vehicle should obtain $1.1V_0$, which is 99 km/h, and otherwise keep the speed at 90 km/h. The ego car should also have both the angle of the car, θ, and its steering angle, δ, close to zero. Therefore $\theta^{ref} = \delta^{ref} = 0$ is used.

In the cost function there are two weight matrices, Q and R. These are used to assign a certain cost to each variable. As no penalty is enforced on the X-coordinate the weight corresponding to the X-penalty is chosen to 0. The weight matrices are chosen based on simulations as:

$$Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 5 \end{bmatrix} \qquad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (38)$$

The next step is to assign constraints to the optimization process. As stated in the earlier sections, the following constraints must hold at all times.

$$x(k+i+1) = Ax(k+i) + Bu(k+i) \quad (39)$$

$$x(k+i|k) \in S_{ego}(k+i) \quad (40)$$

$$u(k+i|k) \in U_{ego} \quad (41)$$

where $i = 0, 1, 2, ..., N$. To make the model more physically correct, another constraint on the velocity to bound the acceleration of the car has to be added. This constrain is chosen as:

$$\left| \frac{u(k+i|k) - u(k+i-1|k)}{h} \right| \leq 4 \text{ m/s}^2 \quad (42)$$

From Eq. 39 it is clear that the cost function is a function of the current state $x(k)$, which is known, and the input sequence $\mathbf{u}(k)$. The input sequence that minimizes the cost function is denoted $\mathbf{u}^*(k)$. From this optimal input sequence only the first element, $u^*(k|k)$, will be used as the input signal to the ego car. In this study the input signals from the MPC will therefore be denoted:

$$u^*(k|k) = [V^*(k|k), \delta^*(k|k)]^T \quad (43)$$

When the optimal input signal at the current state is known, the ego vehicle has to be virtually moved accordingly and the safe states in the new prediction horizon has to be recalculated, based on the new position. This means resetting $R(0)_{ego}$ to be centered at the new position given by Eq. 12. This is performed at each time step k.

This entire process goes on until the overtaking maneuver is finished, i.e. until the ego vehicle has overtaken the other vehicle and returned to the right lane. Every position used for the ego vehicle, and its corresponding time stamp, will form a discrete trajectory of the maneuver.

The trajectory planning is now finished and the result from this section is a discrete trajectory of positions with a specified time, that ensures safety.

D. Trajectory tracking

Now when the trajectory is planned, the next step is to design a simple controller that will enable the real car to track the trajectory. It is of great importance that the ego car follows the trajectory close for safety to be guaranteed. The approach to solve this problem is to design two controllers, one controlling the acceleration and one controlling the front tire angular velocity.

Due to the complexity of multiple controllers working simultaneously the controllers are designed as if the other controller behaves ideally, i.e. the controller which controls the acceleration will assume that the car has the right angle and the controller which controls the steering angle will assume that the car has the right velocity, although this might not always be true.

In [4] a P-controller was shown to successfully track a predefined trajectory, therefore it is motivated to use a similar controller in this study. However in this case, following the trajectory sufficiently close is necessary but not enough. Since each point on the calculated trajectory is associated with a certain time the controllers must ensure that the car reaches each point at the right time. In order to achieve this, a P-controller which applies a control input based on the difference between the current state of the car and the desired sate of the car is used.

As a first attempt the input acceleration is proportional to the difference between the desired speed and the current speed.

The desired speed is defined as the constant speed which is needed in order for the car to reach the trajectory point at the right time.

$$a = K_a \Delta v = K_a(v_{des} - v_c) \tag{44}$$

where the desired speed is denoted v_{des} and v_c is the current speed of the ego vehicle. Since the desired speed is the constant speed needed to reach the next trajectory point at the correct time, it can be written as:

$$v_{des} = \frac{\sqrt{(x_{des} - x_c)^2 + (y_{des} - y_c)^2}}{t_{des} - t_c} \tag{45}$$

where (x_{des}, y_{des}) is the position of the next trajectory point, t_{des} is its corresponding time, (x_c, y_c) is the current position of the ego vehicle and t_c is the current time. The input acceleration can now be written as:

$$a = K_a\Big(\frac{\sqrt{(x_{des} - x_c)^2 + (y_{des} - y_c)^2}}{t_{des} - t_c} - v_c\Big) \tag{46}$$

In order to choose the proportional constant K_a, simple mechanics will be used. Assuming the car traveling in a straight line with constant acceleration a and initial speed v, the distance d traveled during the time Δt is:

$$d = v\Delta t + a\Delta t^2 \tag{47}$$

Solving for a gives:

$$a = \frac{2}{\Delta t}(\frac{d}{\Delta t} - v) \tag{48}$$

By comparing Eq. 46 with Eq. 48 it can be realized that in order to reach the trajectory point at the right time the following choice is appropriate:

$$K_a = \frac{2}{t_{des} - t_c} \tag{49}$$

As K_a is not a constant, this controller is not strictly a P-controller but a modified version of it.

Now the attention is turned to the front tire angular velocity. This controller will also be a proportional controller. In this case the input signal will be proportional to the difference between the desired and the current front tire angle.

$$\dot{\delta} = K_\delta \Delta\delta = K_\delta(\delta_{des} - \delta_c) \tag{50}$$

The desired front tire angle is the constant angle that will place the car in the desired direction during a time Δt. The desired direction of the car is the angle for which the car drives straight towards the next discrete trajectory point. From the bicycle model Eq. 3, the following relation is obtained:

$$\delta_{des} = \tan^{-1}\Big(\frac{L_{ego}}{V_{ego}}\dot{\theta}_{des}\Big) \tag{51}$$

In order to calculate $\dot{\theta}$, the following approximation is made:

$$\dot{\theta}_{des} = \frac{\theta_{des} - \theta_{ego}}{t_{des} - t_c} = \frac{\tan^{-1}(\frac{\Delta y}{\Delta x}) - \theta_{ego}}{t_{des} - t_c} \tag{52}$$

where $\Delta x = x_{des} - x_c$ and $\Delta y = y_{des} - y_c$. This means that θ_{des} is chosen as the desired angle of the vehicle. The input angular velocity can now be written as:

$$\dot{\delta} = K_\delta(\tan^{-1}\Big(\frac{L_{ego}}{V_{ego}}\frac{\tan^{-1}(\frac{\Delta y}{\Delta x}) - \theta_{ego}}{t_{des} - t_c}\Big) - \delta_c) \tag{53}$$

The next step is to choose the constant K_δ. This value is chosen through simulations as 50.

In order to track the discrete trajectory, the tracking controller will run multiple times before changing the reference point. In this study, the tracking controller runs 10 times between every reference update. As the discrete trajectory points are separated by the time step h, defined in Eq. 12, the controller will run periodically with period $h/10$.

IV. Implementation

To perform all of the calculations, Matlab and additional toolboxes are used. In this section a brief description of the procedure is presented.

The time step h is the time step used in the reachability calculation and the planning of the trajectory. In the implementation of the control system the numerical value is set to $h = 0.08$ as it gives a reasonable calculation time.

A. Reachable safe states

As described in the previous section, the mathematics behind the calculations for the reachable safe states relies heavily on set operations. To easily define sets and perform simple set operations, the toolbox described in [8], called MPT3, is used. This toolbox represents sets as polyhedra. These polyhedra are easily used with the predefined set operations in the toolbox. All set operations used in the study are all defined in the previous section.

The first part of the implementation was to calculate the safe reachable states for the ego vehicle at a certain time step. To demonstrate this procedure a plot of the safe states in the prediction horizon at a certain time is shown in Fig. 3.

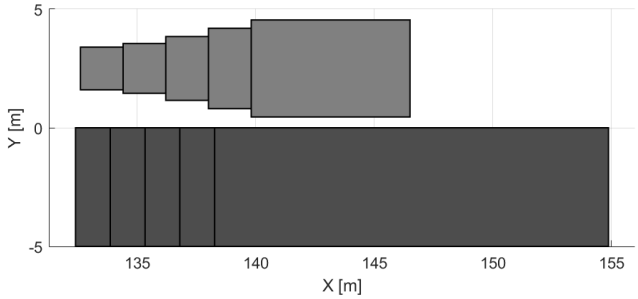

Fig. 3: Reachable safe positions for the ego car in one prediction horizon are shown in light grey. The corresponding reachable set of positions for the other vehicle are shown in dark grey.

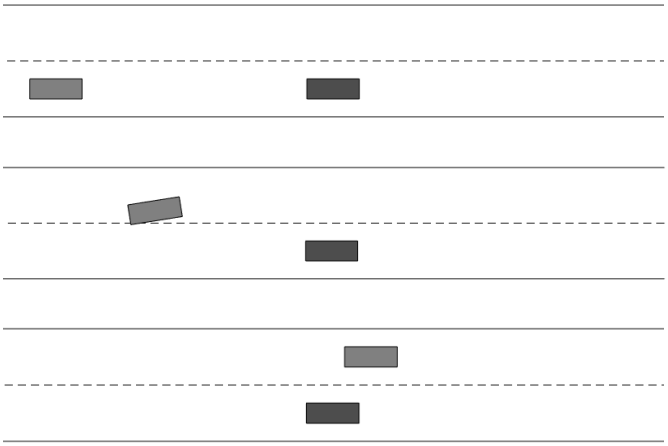

Fig. 4: The ego vehicle in light grey overtaking the other vehicle in dark grey visualized by the simulation environment.

B. MPC

To perform the calculations needed in the MPC framework it is convenient to use the toolbox described in [9]. This toolbox, called Yalmip, provides a simple way to define the prediction horizon, the cost function, the constraints and to calculate the optimized solution. As the MPT3 and Yalmip toolboxes are very compatible with each other it is easy to use the safe sets, defined as polyhedra, as constraints in the MPC calculations.

C. Simulation environment

In order to simulate the behaviour of the ego vehicle a simulation environment was created. The simulation environment is supposed to simulate a real car and its surroundings. From the simulation program the actual trajectory and input of the car can be extracted. To make the simulation program more realistic, the acceleration and front tire angular velocity are used as inputs, i.e. the more complex model from the mathematical modelling of the vehicles is used. Figure 4 shows how the simulation environment displays an overtaking scenario.

V. Simulation results

The main result from the first two parts of the study, i.e. the reachable safe states and the trajectory planning, is the discrete positions describing the safe trajectory, which are shown in Fig. 5, together with their corresponding time stamp. After implementing the tracking controller, the overtaking maneuver was simulated. The obtained trajectory is shown in Fig. 6. From Fig. 5 and Fig. 6 overshoot is seen around e.g. $X = 100$ m. Fig. 7 shows this behaviour more closely.

The behaviour of the car is satisfactory if the car reaches its desired position at the right time. One way to study this is to look at the distance between the desired position and the actual position of the car at each time stamp. This distance is shown in Fig. 8.

From the trajectory planning the calculated input data from the MPC is given, i.e. the speed V^* and the front tire angle δ^* of the ego vehicle. Although these are not used as inputs to the ego car in the simulation, these values can give us information about the tracking behaviour. Both the calculated speed and the speed obtained from the simulation are shown in Fig. 9. The corresponding information about the front tire angle are shown in Fig. 10.

In the simulation the input signal to the car is the acceleration and front tire angular velocity. These input signals are shown in Fig. 11.

VI. Discussion

The main focus of this study was to create a complete control system that can guarantee safety of an overtaking maneuver. However, all the limitations and assumptions made simplifies the problem. In this section, a discussion of the models used and how these might have affected the result are presented. Also, the results from the previous section and how these relate to the model is discussed. The last part of this section covers the possible applications of the method used in this study and how it might be implemented in the future.

A. Model

In this study, the bicycle model from Fig. 2 is used which results in Eq. 1, Eq. 2 and Eq. 3. These equations are then linearized to give Eq. 7, Eq. 8 and Eq. 9. Since they are linearized around $(V_0, \theta_0, \delta_0)$ the equations are approximations for any values (V, θ, δ) different from the ones linearized around. The approximation error increases when the difference between these tuples increases, i.e. the model is only a good approximation when the ego vehicles dynamics are close enough to the dynamics linearized around.

From this, one can draw a couple of conclusions. The first is that it can be expected that the linearized model accurately describes the ego vehicle in the beginning and in the end of the overtaking as the values on (V, θ, δ) are $(V_0, \theta_0, \delta_0)$ in these regions. However when the ego vehicle actually changes lane and speeds up, a less accurate result is expected. For example it can be seen that while the ego car has a constant speed of V and a constant angle of $\theta > 0$, Eq. 7 gives that the car should move forward along the X-axis with speed V, while Eq. 8 shows that the car is moving along the Y-axis with speed $V_0\theta$. This gives a total speed of $\sqrt{V^2 + (V_0\theta)^2}$ instead of the actual speed V.

The second conclusion is that in the studied case, the vehicles are assumed to travel at high speeds which means that the angles θ and δ of the cars are always close to 0. Therefore the linearized model should be a good approximation. In lower speed scenarios, for example urban driving where the angles θ and δ might significantly differ from 0, the linearized model is probably not satisfied. Essentially this means that our methods should not be naively applied to urban driving scenarios.

B. Results

The results are essentially of two different types; results connected to the trajectory and results connected to the input signals.

1) Trajectory analysis: From the MPC, the discrete trajectory displayed in figure 5 is obtained. From the reachability analysis it is guaranteed that the generated trajectory is safe. In addition to this, the length of the overtaking is approximately 350 m, which is reasonable at the high speeds used. This might seem like a long overtaking, but bear in mind that the ego vehicle overtakes all possible position of the other vehicle independent of the actual speed of the other vehicle. It is also seen that the car strives to stay in the middle of its reference lane and that the angle of the car is within the valid interval. This is the behaviour that was demanded from the MPC.

As has been mentioned, each position displayed by a point also corresponds to a certain time stamp. This means that the points on the trajectory should be closer together when the ego car has the lower speed and further apart when the ego car has the higher speed. This behaviour is hard to see in figure 5, but will be verified through the tracking, where it is seen that the car speeds up and still stays close to the reference trajectory points. This means that the planned trajectory possesses the appropriate behavior and the result is therefore satisfactory.

The planning part is now finished and left is the tracking of the planned trajectory. The actual trajectory of the car, shown in figure 6, is position-wise very similar to the planned trajectory. Some overshoot is seen at the end of the lane changes, which is expected since a simple P-controller is used. From figure 7 it can be seen that the overshoot is less than 0.2 m, which is a very small distance in this context.

Although the positions of the car seems to be good enough, no conclusions can be drawn from figures 5, 6 and 7 regarding whether the car actually follows the discrete trajectory as the time aspect is not yet considered. In order to analyze this, figure 8 is useful. One can see that the distance between the actual position of the car and the reference point is always less than 0.15 m. This verifies that the car follows the trajectory, including the time aspect. This also gives an upper bound on the overshoot mentioned earlier. The result from the tracking part verifies that the P-controllers were sufficient, at least in simulations. In reality however, there are a lot of disturbances which probably would require a more advanced controller.

Another interesting thing to mention here is that the worst behaviour, i.e. the largest distance, is seen when the car turns. This will be further discussed in the input analysis.

2) Input analysis: In addition to the discrete trajectory, the calculated optimal input (V^*, δ^*) is obtained from the MPC. As the input to the car in the simulation are $(a, \dot{\delta})$, the input (V^*, δ^*) could have been treated as references to track. However, this would not be sufficient as safety needs to be ensured, i.e. that the car is at the right position at the right time, not that the car has a certain speed and front tire angle. The reason that the car would not be at the right position at the right time if (V^*, δ^*) would have been tracked is due to the fact that these are calculated with the linearized model and this is not the way the real car would behave.

Although the speed V^* and front tire angle δ^* of the modelled car is not used as a reference trajectory, it is very interesting to compare the values generated from the MPC and the actual values from the simulated car. The speeds are shown in figure 9 and the front tire angles are shown in figure 10.

As can be seen, the speed of the ego vehicle follows V^* closely except of two shorter time intervals. In these intervals, the speed is significantly higher than V^*. From Fig. 10 it is seen that this occurs when the car turns. This means that the car speeds up, in order to reach each trajectory point in the right time, when it turns. As already mentioned in the *Model* discussion, the linearized model of the car will generate a trajectory based on the velocity $\sqrt{V^{*2} + (V_0\theta)^2}$, while the car still has velocity V^* in the model. With the values used in this study, this means that the trajectory is generated with a velocity of approximately 27.8 m/s which agrees with the actual speed of the car shown in Fig. 9. Therefore the actual speed of the ego vehicle is expected.

The front tire angle input is very similar to δ^* as seen in

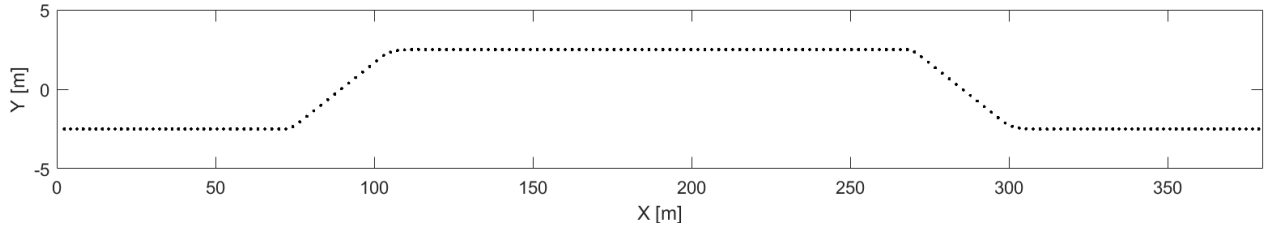

Fig. 5: Discrete trajectory of positions on the road. Each point is correlated with a certain time stamp. Note the gradation on the axes.

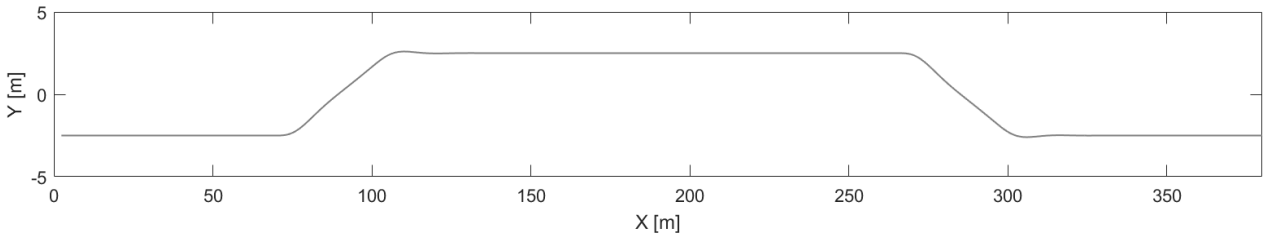

Fig. 6: The actual trajectory of the ego vehicle when performing the maneuver. Note the gradation on the axes.

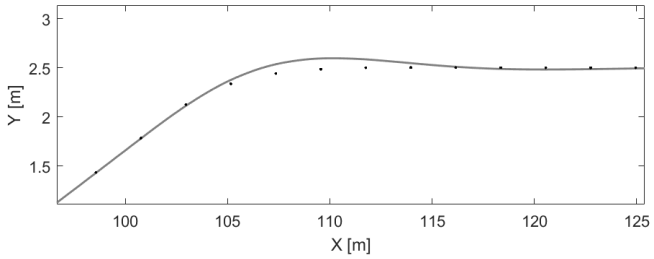

Fig. 7: The discrete trajectory together with the actual trajectory capturing the overshoot. Note the gradation on the axes.

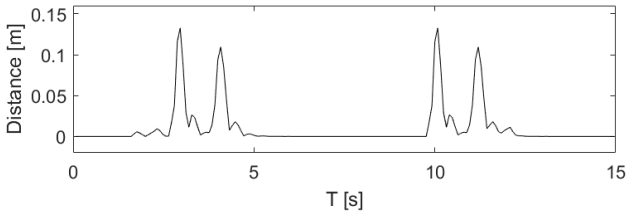

Fig. 8: The distance between the desired position and the actual position of the ego vehicle at each time stamp.

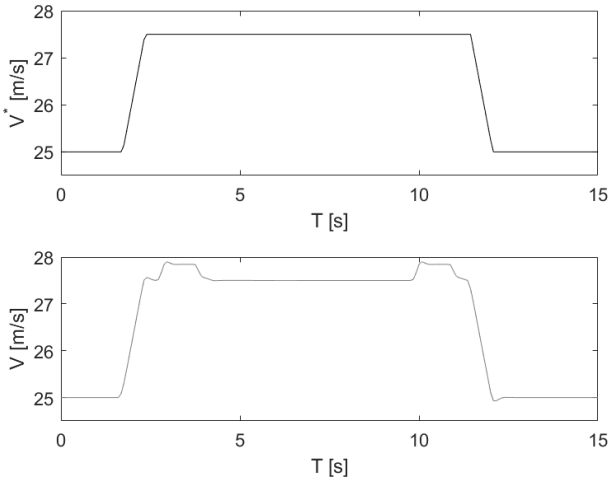

Fig. 9: Input speed calculated in the MPC, V^*, and the actual speed of the ego vehicle in simulation, V.

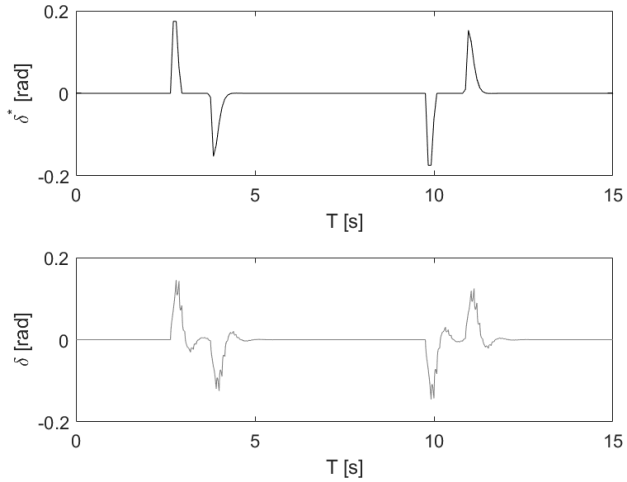

Fig. 10: Front tire angle input calculated in the MPC, δ^*, and the actual front tire angle of the ego vehicle in simulation, δ.

Fig. 10. The difference between δ^* and the actual δ is that δ has some overshoot, oscillations and is not as smooth as δ^*. This is expected since a simple P-controller is used for the front tire angular velocity input.

The real input to the car is the acceleration and the front tire angular velocity, shown in Fig. 11. The acceleration input signal is as expected, based on the speed discussion and the constraint of ± 4 m/s^2. The two wide peaks which reaches the constraint acceleration corresponds to when the car obtains a new speed reference. The four lower peaks corresponds to the speed change needed when the car turns as discussed above.

The interesting part from the front tire angular velocity input is its drastic and oscillatory changes. This is a consequence of how the P-controller is designed and how the reference is updated. However, notice that this is not a strange behaviour since this is the angular velocity of the steering wheel and that the updates are discrete in time with a very small time step. The time step is so small that a human passenger would probably not be able feel the oscillation. A human would rather feel the steering wheel angle which is much smoother.

Overall we can see that all inputs has the expected behaviour which is reflected by the great tracking result. This verifies that the P-controller was sufficient for the tracking problem.

C. Methods

In this subsection the methods used in this study will be discussed.

1) Reachable safe states: The aim of the reachability analysis was to generate sets representing the safe states within a prediction horizon. The safe sets are expected to successively expand and never overlap the reachable set of the other vehicle. Fig. 3 shows the sets for a certain prediction horizon and shows that it holds the desired behavior.

The safe reachable states for the entire overtake are calculated from the initial state of the two cars. Although this means that the overtaking maneuver might be unnecessary long, it is crucial to know that the entire overtaking can safely be performed before even starting the maneuver. This can only be guaranteed through calculations based on the initial state of the cars. To solve the problem of an unnecessarily long overtake, a more advanced method could be used. One such method is to gather data about the other vehicle during the overtake and recalculate the future safe sets. The MPC would then calculate a trajectory based on the current state of the system rather than its initial state.

2) MPC: The MPC framework uses predicted states in order to calculate the optimal input. This is very useful in traffic situation as the desired behavior of a traffic participant highly depends on possible future states of a traffic system. In this study this is utilized as the future safe sets are known and

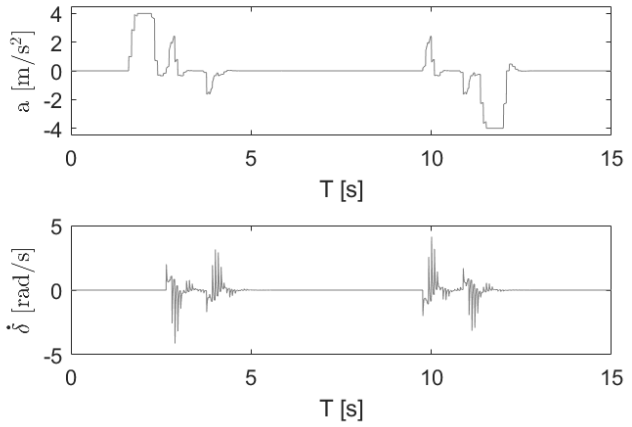

Fig. 11: The two input signals a and $\dot{\delta}$ to the ego vehicle in the overtaking simulation.

the main focus is on creating a safe trajectory. From the safe set constraints it is known that a generated trajectory is indeed safe, but the shape of it is determined by the cost function. One might add additional costs to the cost function presented in this paper in order to get another desired behaviour such as a faster, a more passenger friendly or a more eco-friendly overtake.

D. Applications

As already mentioned, the overtake situation in this study is conducted with constraints that in most cases are realistic, but hard to guarantee. One such example is the assumption that the other vehicle will always stay in its lane and not speed up significantly. In today's traffic this can of course not be ensured and although it is still possible to calculate the trajectory, safety can not be guaranteed.

Even though constraints like the aforementioned are hard to ensure in today's traffic, it might not at all be hard to ensure if the other car is also autonomous. In a smart traffic system the ego and other vehicle could exploit vehicle-to-vehicle, V2V, communication to accomplish this. However there are still a lot of challenges for such communication which is shown in [10].

Another interesting point to mention is that overtaking might not be the only or best fitted application for the methods presented in this study. A lot of other traffic scenarios, such as lane change, intersection and roundabout situations could be handled in a similar way.

E. Future work

The methods used in this paper would benefit from the development of V2V communication. A repetitive problem is that the ego vehicle is not able the foresee other vehicles behaviour which might be solved using V2V communication. A contract based communication language between vehicles which bounds the possible behaviour of each vehicle might be a solution.

As mentioned in the *Methods* discussion, calculating the safe sets based solely on the initial state is problematic. To gather data and recalculate the safe sets during the overtake would generate an improved trajectory and strengthen the overall performance of our complete control system.

Other obvious extensions of this study that would greatly increase the applicability would be to study more complex traffic scenarios with the same approach. Such scenarios should include multiple other traffic participants, oncoming traffic and non-straight roads. Also the limitations imposed by the linearization done in order to plan the trajectory has been discussed. A study treating the procedure for the non-linear system is of great importance in order to make this applicable to situations of greater velocity changes or sharper turns, which is usually the case in urban driving.

VII. Conclusion

This study aimed to create a complete control system for an overtaking maneuver on a straight two-lane highway that can also guarantee safety. The procedure was divided into three main parts; calculations of safe states using reachability analysis, trajectory planning using the framework of model predictive control and trajectory tracking. With the methods used it can be guaranteed that a planned trajectory is safe.

The result from the trajectory planning reveals that the used procedure works as intended. The planned discrete trajectory shows a very reasonable overtaking path, which is also successful in simulation.

The tracking used to follow this path is designed to ensure that the car stays sufficiently close to the planned safe trajectory at all times. From the error analysis it is seen that the controller used is successful.

Overall the complete control system plans and performs a safe overtaking maneuver in a satisfying way. However, the overtaking situation in this study is simplified from reality and from the discussion we can conclude that improvements has to be made before a real implementation is possible.

Acknowledgment

The authors would like to express their great gratitude to the project supervisor, Yulong Gao. His support, encouragement and guidance throughout the project has been highly appreciated.

References

[1] G. Hegeman, K. Brookhuis, S. Hoogendoorn *et al.*, "Opportunities of advanced driver assistance systems towards overtaking," *EJTIR*, vol. 5, no. 4, pp. 281–296, May 2005.

[2] M. Obayashi, K. Uto, and G. Takano, "Appropriate overtaking motion generating method using predictive control with suitable car dynamics," in *Proc. of 55th IEEE Conference on Decision and Control (CDC)*, Dec 2016, pp. 4992–4997.

[3] J. Nilsson, P. Falcone, M. Ali, and J. Sjoberg, "Receding horizon maneuver generation for automated highway driving," *Control Engineering Practice*, vol. 41, pp. 124–133, Aug 2015.

[4] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, Aug 2014.

[5] M. Althoff and S. Magdici, "Set-based prediction of traffic participants on arbitrary road networks," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 2, pp. 187–202, Jun 2016.

[6] H. Andersen, W. Schwarting, F. Naser, Y. H. Eng, M. H. Ang, D. Rus, and J. Alonso-Mora, "Trajectory optimization for autonomous overtaking with visibility maximization," in *Proc. of 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Oct 2017, pp. 1–8.
[7] B. Kouvaritakis and M. Cannon, *Model Predictive Control: Classical, Robust and Stochastic*. Springer, 2016.
[8] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proc. of the European Control Conference*, Jul 2013, pp. 502–510.
[9] J. Löfberg, "Yalmip : A toolbox for modeling and optimization in matlab," in *Proc. of the CACSD Conference*, 2004, pp. 284–289.
[10] S. Lefevre, J. Petit, R. Bajcsy, C. Laugier, and F. Kargl, "Impact of v2x privacy strategies on intersection collision avoidance systems," in *Proc. of 2013 IEEE Vehicular Networking Conference (VNC)*, Dec 2013, pp. 71–78.