The Role of Risk Parameter for Stochastic Model Predictive Control for Autonomous Highway Driving

Scientific work within the Ingenieurspraxis
from the Department of Electrical and Computer Engineering at the Technical University of Munich.

Supervised by Univ.-Prof. Dr.-Ing./Univ. Tokio habil. Martin Buss
M.Sc. Tommaso Benciolini
Chair of Automatic Control Engineering

Submitted by cand. ing. Ivan Nikolovski
Soldauer Straße 27
81927 Munich
+491781861183

Submitted on Munich, 07.06.2021

TECHNISCHE UNIVERSITÄT MÜNCHEN
LEHRSTUHL FÜR STEUERUNGS- UND REGELUNGSTECHNIK
UNIV.-PROF. DR.-ING./UNIV. TOKIO HABIL. MARTIN BUSS
DR.-ING. MARION LEIBOLD

March 11, 2021

I N G E N I E U R P R A X I S
for
Ivan Nikolovski
Student ID 3717921, Degree EI

The Role of the Risk Parameter in Stochastic Model Predictive Control for Autonomous Highway Driving

Problem description:

Model Predictive Control (MPC) is nowadays affirmed as a leading approach for model-based control algorithms for autonomous driving [3]. At each sampling time, the algorithm computes a sequence of control inputs minimizing a cost function over a finite horizon, taking into account the satisfaction of a set of constraints. By properly designing the cost function, desired control goal can be rewarded, whereas safety conditions (for example, maintaining a safety distance from other road participants) can be encoded in the set of constraints.
In practice, however, deterministic safety constraints cannot be formulated, since the behavior of other vehicles is uncertain. For this reason, if a stochastic description of the other vehicles' behavior is known, Stochastic Model Predictive Control (SMPC) can be used [2, 1], requiring that the safety constraints hold at least with a given probability (risk parameter).
The aim of this work is to implement a SMPC controller and to test it in an automated highway driving scenario, discussing the role of the risk parameter in balancing the efficiency and safety of the executed maneuvers.

Work schedule:

- Literature research on MPC and SMPC
- Implementation in Matlab of the SMPC scheme for a highway driving scenario
- Discussion, through numerical simulations, of the role of the risk parameter

[1] T. Brüdigam, M. Olbrich, D. Wollherr, and M. Leibold. Stochastic model predictive control with a safety guarantee for automated driving, 2020. arXiv: 2009.09381.
[2] A. Carvalho, Y. Gao, S. Lefevre, and F. Borrelli. Stochastic predictive control of autonomous vehicles in uncertain environments. In *12th International Symposium on Advanced Vehicle Control*, Tokyo, Japan, 2014.
[3] X. Qian, I. Navarro, A. de La Fortelle, and F. Moutarde. Motion planning for urban autonomous driving using bézier curves and mpc. In *IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 826–833, 2016.

Supervisor:	M. Sc. Tommaso Benciolini
Start:	15.03.2021
Delivery:	17.05.2021

(M. Leibold)
Akad. Rat

Contents

Chapter 1

Goal Of The Project And Starting Point

1.1 Starting Point, Technical Background And Goal Of The Project

The basic idea of the assignment was to develop a toolbox which could successfully control and simulate an autonomous vehicle on the road, which would always keep to the road, avoid collisions with other vehicles and preferably avoid stopping, keep a constant speed, yaw angle and trajectory unless otherwise specified and try to minimize the change of acceleration and steering angle. A visualization of the vehicles on the road as well as plots of all of the states (velocity, lateral and longitudinal position, yaw angle) and their respective change with time as well as all of the inputs (acceleration and steering angle) and their respective change with the time had to also be developed.
The starting point was the NMPC (Nonlinear Model Predictive Control) Algorithm from the book Nonlinear Model Predictive Control: Theory and Algorithms [GP17], which had to further be developed to reach the goal of the project a autonomous highway driving vehicle. This control based algorithm tries to minimize a given function, based on some constraints and then calculates the optimal control input and future states based on these inputs, and always applies just the first input to the initial state (a more detailed explanation will be given in Chapter 2).
The NMPC Algorithm is a suitable tool for designing an automated vehicle, since when we are driving a vehicle we subconsciously predict what the other vehicles involved in the driving scenario will do. Another reason why the NMPC algorithm is a useful tool for automated driving, is the fact that constraints are easily implemented into the NMPC framework. This allows us to make sure that the vehicle doesn't deviate of the road or drive with a high speed.
For this engineers practice only a PC which could run MATLAB was required and a premade NMPC MATLAB toolbox from the book Nonlinear Model Predictive Con-

trol: Theory and Algorithms from Jürgen Pannek and Lars Grüne[GP17]. Except that no other hardware or software was required during working on this engineers practice.

Chapter 2

Solution Concept And Realization Of The Project

2.1 Introduction To Nonlinear Model Predictive Control

Nonlinear model predictive control (NMPC) is an optimization based method for the feedback control of nonlinear systems. Its primary applications are stabilization and tracking problems. Suppose we are given a controlled process whose state $x(n)$ is measured at discrete time instants $t * n$, $n = 0, 1, 2, \dots$. "Controlled"means that at each time instant we can select a control input $u(n)$ which influences the future behavior of the state of the system. In tracking control, the task is to determine the control inputs $u(n)$ such that $x(n)$ follows a given reference $x_{ref}(n)$ as good as possible. This means that if the current state is far away from the reference then we want to control the system towards the reference and if the current state is already close to the reference then we want to keep it there. The goal of the NMPC algorithm is to minimize a cost function J with a given stage cost l, subject to the state function of the system f. The key steps to the NMPC algorithm (as defined in [GP17]) are the following:

Algorithm 1 (basic NMPC algorithm for constant reference x_{ref})

1. Measure the state $x(n)$ of the system
2. Set $x_0 = x(n)$ solve the optimal control problem (OCP)
   minimize
   $$J_N(x_0, u(.)) = \sum_{k=0}^{N-1} l(x_u(k, x_0), u(k)) \tag{2.1}$$
   with respect to $u(.)$
   subject to $x_u(0, x_0) = x_0$, $x_u(k + 1, x_0) = f(x_u(k, x_0), u(k))$

and denote the optimal control sequence $u^*(.)$, with $u(.)$ being a control sequence and x_u the state trajectory (state values of from 0 to N with respect to u).

3. Define the NMPC feedback $\mu(x(n)) = u^*(.)$ and use the control value $u(0)$ in the next sampling period.

The closed loop system resulting from the NMPC algorithm is defined as

$$x^+ = f(x, \mu_N(x)). \tag{2.2}$$

One of the main reasons for the usage of the NMPC algorithm for solving the problem of autonomous driving is the fact that the NMPC algorithm takes constraints into account. The NMPC framework allows for constraints to be included in the OCP. Constraints are useful for autonomous driving since, they can express speed limits, lane boundaries, etc. The constraints in the NMPC algorithm can be applied to both the state and the control input. The fact that we can constrain both the control input and state is particularly useful in the field of autonomous driving since we don't want our car to accelerate or decelerate with a large acceleration and we do not want our car to deviate out of the road either or to drive above the speed limit. The SMPC (Stochastic Model Predictive Control) algorithm allows for further improvements on this idea. It allows the constraints to hold true to a certain probability, this is useful because it decreases the conservativeness of the automated vehicle. However this does not ensure absolute safety, which is definitely a negative for automated driving. One of the approaches to resolve this is the usage of a FTP (Failsafe Trajectory Planning), see [BOWL21].

2.2 Vehicle Models

There will be 2 models used in the code (which were also used in [BOWL21]).

1. Model of the ego vehicle (EV): the kinematic bicycle model.

2. Model of the target vehicles (TV): the TV model.

2.2.1 EV Model

For the EV a discretized kinematic bicycle model was used. The following is the continuous kinematic bicycle model.

$$\dot{s} = v \cos(\phi + \alpha) \tag{2.3}$$

$$\dot{d} = v \sin(\phi + \alpha) \tag{2.4}$$

$$\dot{\phi} = \frac{v}{l_r}\sin(\alpha) \tag{2.5}$$

$$\dot{v} = a \tag{2.6}$$

$$\alpha = \arctan(\frac{l_r}{l_r + l_f}\tan(\delta)). \tag{2.7}$$

Where s is the longitudinal deviation, d is the lateral deviation, ϕ is the yaw angle and v is the velocity, l_r and l_f are the distances from the vehicle center of gravity to the vehicle front and rear axis, δ is the steering angle and a is the acceleration. The input vector is $u = \begin{pmatrix} a & \delta \end{pmatrix}^T$ and the state vector is $x = \begin{pmatrix} s & d & \phi & v \end{pmatrix}^T$.
The main problem with the continuous model is that the NMPC algorithm requires a discrete time model as stated previously. So that is why we have to discretize the continuous model. Which results in the following model

$$x_{k+1} = x_0 + Tf^c(x_0, 0) + A_d(x_k - x_0) + B_d u_k, \tag{2.8}$$

with A_d and B_d being the discretized state and input matrices (see Apendix page 18 of [BOWL21]) and f^c being the continuos time model at state x_0 and control input vector u equal to 0.

2.2.2 TV Model

The TV model is required to ensure the safety of the other vehicles involved in the highway driving scenario and for the creation of a safety rectangle around each TV vehicle, which will be mentioned later. The TV model is described as follows

$$x_{k+1}^{TV} = Ax_k^{TV} + Bu_k^{TV} \tag{2.9}$$

$$u_k^{TV} = \tilde{u}_k^{TV} + w_k^{TV}, \tag{2.10}$$

with $\tilde{u}_k^{TV}$ being

$$\tilde{u}_k^{TV} = K(x_k^{TV} - x_{ref,k}^{TV}), \tag{2.11}$$

A and B being

$$A = \begin{pmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.12}$$

$$B = \begin{pmatrix} 0.5T^2 & 0 \\ T & 0 \\ 0 & 0.5T^2 \\ 0 & T \end{pmatrix} \tag{2.13}$$

and $K = \begin{pmatrix} 0 & k_{12} & 0 & 0 \\ 0 & 0 & k_{21} & k_{22} \end{pmatrix}$. x_k^{TV} is the state of the TV at time point k, u_k^{TV} is the control input of the TV at time point k and $\tilde{u}_k^{TV}$ is the controller for the control input. The state vector x consists of $x = \begin{pmatrix} x^{TV} & v_x^{TV} & y^{TV} & v_y^{TV} \end{pmatrix}^T$ and the input vector u consists of $u = \begin{pmatrix} a_x & a_y \end{pmatrix}^T$.
The perturbation of the input w_k^{TV} is a 0 mean Gaussian noise with covariance matrix Σ_w^{TV} and $\tilde{u}_k^{TV}$ can be calculated as follows

$$\tilde{u}_k^{TV} = K(x_k^{TV} - x_{ref,k}^{TV}), \tag{2.14}$$

with $x_{ref,k}^{TV}$ being the TV reference. The reason why this noise is included is to propagate the uncertainty that may arise from the future movements of the other vehicles involved.

2.2.3 Cost Function For The Simulation

As a cost function for this simulation we will use the following function

$$J = \sum_{k=0}^{N-1} (x - x_{ref})^T Q (x - x_{ref}) + u^T R u. \tag{2.15}$$

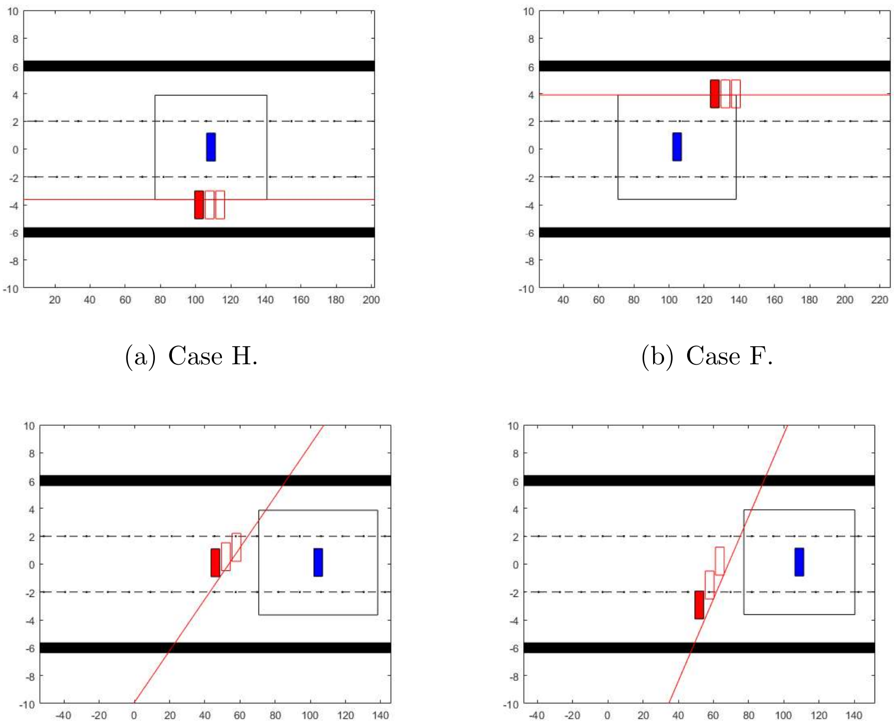
The Q and R matrices are positive-definite diagonal matrices. Each diagonal term describes a degree of value for the cost function. For example if we set Q(3,3)=3 and Q(2,2)=10, it means the algorithm prioritizes keeping the lateral deviation as close as possible to the reference more than keeping the yaw angle as close as possible the references. One thing to note here is that the Q(1,1)=0 since there is no reference for the longitudinal deviation.

2.3 Safety Rectangle And Constraints

In order to ensure the safety of everyone driving we have to create a so called safety rectangle for each of the TVs. The purpose of this is so that our EV will be constrained to never enter this safety rectangle and ensure maximum safety for the other vehicles involved in the driving scenario. This is where we use our previously discussed TV model. To see details on how to develop the safety rectangle, see [BOWL21].
Now that we have the safety rectangle all that is required for a successful simulation are the constraints. We have to constraint the car to stay on the road, then we need to make sure it keeps the speed below speed limit. Both of these properties are easily achievable with the usage of the NMPC toolbox. The most important constraint and the most difficult one to implement is the one based on the other target vehicles. We can write this as

$$0 \geq q_y y_k + q_x x_k + q_t, \tag{2.16}$$

with q_x being the longitudinal constraint and q_y the lateral constraint.
This equation is separated into multiple cases with multiple constraints. As a general rule our EV only overtakes from the right to the left. A detailed overview of how each case functions and how the EV moves can be found in the appendix of [BOWL21]. A few simple cases can be found in Figure 2.1.



(a) Case H.

(b) Case F.

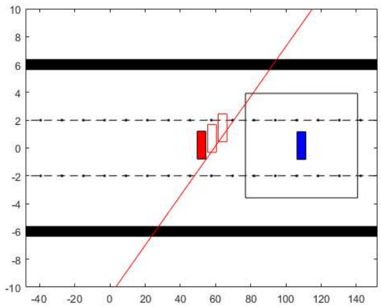
(c) Case D.

(d) Case E.

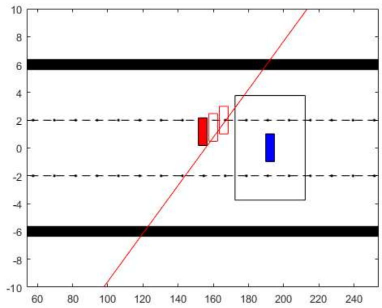
Figure 2.1: Visualization of some of the cases covered.

2.4 A Simple Simulation

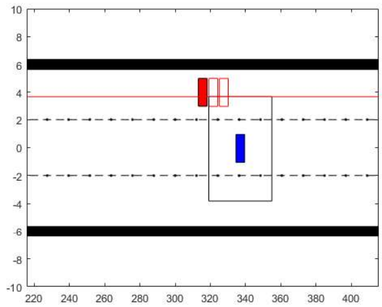
With all of the necessary steps completed it is now possible to run a simple simulation. We will take 2 vehicles into consideration with the following coordinates. $x_0^{EV} = \begin{pmatrix} 40m & 0m & 0rad & 30m/s \end{pmatrix}^T$ and $x_0^{TV} = \begin{pmatrix} 100m & 20m/s & 0m & 0m/s \end{pmatrix}^T$. We will set the safety factor $\beta = 0.7$.
Looking at Figure 2.2 we can see 2 vehicles. The blue vehicle is the TV, while the red vehicle is the EV. The rectangle around the blue vehicle is the safety rectangle, the 2 red rectangles after the red vehicle are the predicted states in the next 2 steps



(a) Before overtaking.



(b) Overtaking.



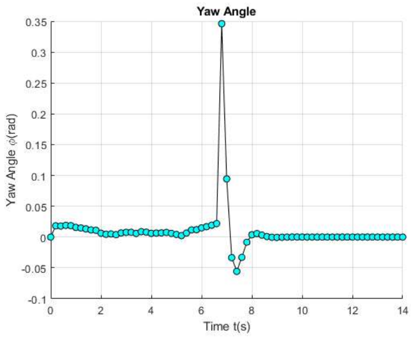
(c) After overtaking.

Figure 2.2: Overtaking maneuver.

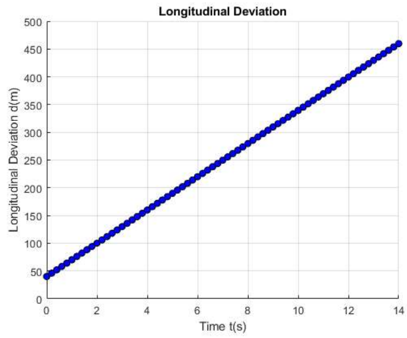
of the SMPC simulation. The red line represents the constraint which has to be satisfied in the next step of the simulation. The speed with which the EV is moving is 30m/s, while the TV is accelerating from a speed of 20m/s to a speed of 27m/s, which results in the change in size of the safety rectangle in sub-figures (b) and (c). The EV must eventually overtake the TV. With the simulation it is visible the EV is able to successfully overtake with regard to the given constraints. In Figure 2.3 we can see how the states change withing the first 14 seconds of the simulation. The results show that the EV is able to keep a constant speed, while changing lateral deviation because of the overtaking maneuver and also change the yaw angle when it overtakes and then return to a almost zero yaw angle.

2.5 Risk Parameter

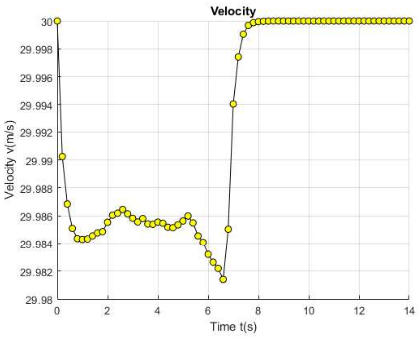
The risk parameter β has a vital influence on the end result of the simulation. Choosing a large risk parameter might cause very conservative driving or in some cases it might even cause the EV to deviate too far from the middle of the lane, wheres a small risk parameter may take dangerous routes. Of course the risk parameter is largely dependent on the covariance matrix so in term choosing relatively large



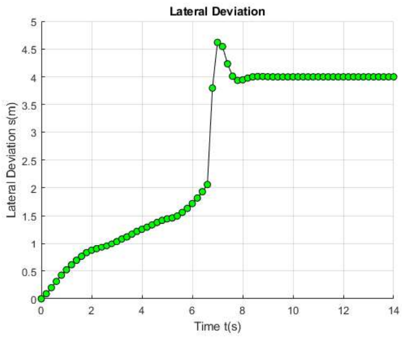
(a) Yaw angle.



(b) Longitudinal deviation.



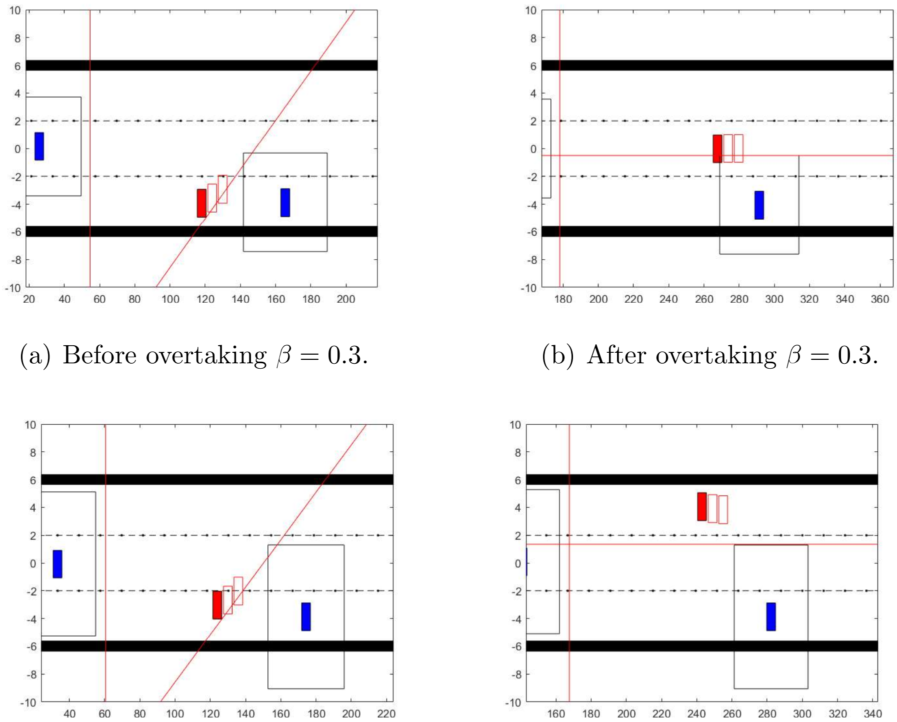
(c) Speed.



(d) Lateral deviation.

Figure 2.3: State Changes with respect to Time.

values for the noise results in an higher importance of the risk parameter, whereas as small values make the risk parameter almost meaningless. Also another important factor is the $\tilde{a_r}$ term, or more precisely the speed of the EV vs the speed of the TV. A large speed of the EV always results in a bigger safety rectangle which decreases the importance of the risk parameter in the simulation, whereas if the speed of the TV is larger then the risk parameter becomes more important. In general the best choice for a risk factor lies somewhere between the range of 0.4 and 0.7 since this ensures more safety for the other vehicles, but also makes sure the EV does not have relatively conservative movements. Taking a look at Figure 2.4 we can see how does the simulation differ when a higher value is chosen for the risk parameter. The upper row has a risk parameter β of 0.3, while the lower row of 0.9. The results align with our expectations, the higher risk factor results in a relatively conservative approach and a bigger safety rectangle area. As expected the risk factor of $\beta = 0.3$ results in a less conservative driving maneuver of switching just 1 lane, whereas the higher risk factor $\beta = 0.9$ results in the EV switching to the farthest lane so that safety is ensured for the other TV involved in highway scenario, which in this case is not necessary.



(a) Before overtaking $\beta = 0.3$.

(b) After overtaking $\beta = 0.3$.

(c) Before overtaking $\beta = 0.9$.

(d) After overtaking $\beta = 0.9$.

Figure 2.4: Simulation results with different safety parameters.

Chapter 3

Further Development And Implementation

The SMPC algorithm for automated driving which was discussed in this project works as it should, but there are some new things which could be developed. One thing which could be added is a new set of constraints. It is noticeable from looking at the table in the appendix of [BOWL21], all the constraints with which the EV overtakes a TV are based on switching to the left lane. So when the EV is at the utmost left lane it will always stay there. A simple way to improve this is to add similar constraints as cases E and D for the right corner of the TV. Another possible implementation is a algorithm which saves a backup of the solution of the SMPC algorithm in case the SMPC algorithm is not solvable in the next steps. This would ensure that the vehicle keeps moving at the safe states although the current iteration does not provide it with a feasible solution. Another idea which could be further developed is involving pedestrians in the simulations and making constraints based on their movement and also making a prediction of there movements as we did with the TVs. This would of course be a more challenging task since pedestrians are not constrained to the limits of a lane or road such as vehicles.

List of Figures

List of Tables

Bibliography

[BOWL21] Tim Brüdigam, Michael Olbrich, Dirk Wollherr, and Marion Leibold. Stochastic model predictive control with a safety guarantee for automated driving. *IEEE Transactions on Intelligent Vehicles*, 2021. doi:10.1109/TIV.2021.3074645.

[GP17] Lars Grüne and Jürgen Pannek. Nonlinear model predictive control. In *Nonlinear Model Predictive Control*. Springer, 2017.

License