BSc Thesis Applied Mathematics

Path and Trajectory Planning by Model Predictive Control in Autonomous Racing

A.V. van Sprang

Supervisors: Prof. Dr. P.J.F. Lucas, Prof. Dr. H.J. Zwart, M. de Bruin (RDW), G. Hekkelman (RDW), R. Hulshoff (RDW)

February, 2021

Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science

RDW

UNIVERSITY OF TWENTE.

Preface

This research thesis is submitted for partial fulfillment of the degrees Bachelor of Science in Applied Mathematics and Technical Computer Science (University of Twente). Furthermore, this has been combined with an internship at the Netherlands Vehicle Authority (Dienst Wegverkeer, RDW).

Acknowledgements are due to all who contributed to the work presented in this thesis. First, I would like to thank my supervisors Hans Zwart and Peter Lucas, for reading through all concept versions and always providing helpful feedback, even for subjects neither of you were really familiar with. Secondly, I would like to thank the team Data Science and Car & IT at RDW for letting me join your team. I enjoyed (and learned from) joining your stand-ups, sprint plannings and digital pubquizzes. Especially, I would like to thank my supervisors Marc de Bruin and Gerard Hekkelman, who have guided me along the way, read through concept versions and provided technical support. Finally, I would like to thank my friends and family who supported me in many ways. Some of you have helped me directly by reading the report and thinking along, others have digitized my sketches (thank you, Mies) and all have cared for me during the process (especially my boyfriend, Milan). All of your help was much appreciated and have contributed to this thesis.

In addition, it is merely appropriate to state this research has been based on and inspired by the Master's Research Thesis of S. De Boer (Spatial-Based Model Predictive Control for Trajectory Generation in Autonomous Racing). Both his report and source code were available to the RDW and both have been used in the research. I have also consulted him once during the research and I am thankful for his contributions, which have absolutely contributed to this thesis.

Path and Trajectory Planning by Model Predictive Control in Autonomous Racing

Angela van Sprang

February, 2021

Abstract

Driverless vehicles could have a tremendous impact on society, by making the mobility system safer, more accessible and more sustainable. However, there are legal challenges to overcome before fully automated vehicles will drive on public roads. The Netherlands Vehicle Authority (Dienst Wegverkeer, RDW) evaluates vehicles and is interested in the capabilities of self-driving vehicles. The goal of this thesis is to help provide insights in the capabilities of self-driving vehicles, by determining the fastest path and corresponding velocities of autonomous go-karts on a given track, using Model Predictive Control. First, the road and the go-kart are modelled and a state representation is given. The go-kart is modelled by a kinematic bicycle model and the time- and space-derivatives are determined. Then, a controller algorithm is formulated that uses (a simplification of) the vehicle model to calculate the optimal trajectory based on the road and vehicle constraints. The objective of the controller is to maximise (a simplification of) the progress along the center-line of the road. Finally, the MPC algorithm is programmed and tested for different race tracks and in different scenarios. The objective function of the controller behaves as expected, although the controller is not robust with respect to all testing parameters in the MPC algorithm. Considering that approximations have been made in this thesis with regard to the vehicle model, cost function and vehicle dynamics, the proposed MPC algorithm determines the fastest path and corresponding velocities reasonably well.

Keywords: path planning, trajectory planning, model predictive control, self-driving vehicles, autonomous racing

Contents

1 Introduction

Driverless vehicles will shape the future of mobility and could have a tremendous impact on road users and the mobility system as a whole [1]. Human error is estimated to play a role in 94% of accidents, giving driverless vehicles the opportunity to improve road safety [2]. They could also bring mobility to people who cannot drive themselves (e.g. elderly or disabled people) and revolutionise urban planning by freeing up space wasted in parking. Such technology (although limited in full self-driving capabilities) is already on the market [3]. However, there is still a long way to go before fully automated vehicles will be on public roads, including technical and legal unsolved challenges [1], [4].

1.1 Research Motivation

The Netherlands Vehicle Authority (Dienst Wegverkeer, RDW) is an organization that approves and licenses vehicles and vehicle parts in the Netherlands. Her mission is to provide safety, sustainability and legal security in mobility [5]. RDW has an increasing focus on software in vehicles, including Advanced Driving Assist Systems (ADAS), which assist drivers in parking and driving tasks. It is important for the RDW to be able to evaluate these, in order to permit driverless cars on the public road in the future.

In order to gain knowledge on ADAS, the RDW recently launched a self-driving challenge in which a go-kart is equipped with an onboard vehicle computer (NVIDIA DRIVE). The objective of this challenge was to let the go-kart drive autonomously and as fast as possible on a track by means of line detection (or possibly any other functionality such as object and weather detection). This computer is equivalent to the technology present in current self-driving cars [6]. Therefore, programming this computer such that a go-kart drives autonomously gives insight in the capabilities of the current car technology in the real world. However, this link to the real world is not yet made in the challenge. It is not evaluated whether the driven path is actually the fastest and whether the go-kart drives as fast as physically possible and so whether the program of the computer could be improved. This thesis aims to evaluate the race performance of the go-karts as programmed in the self-driving challenge, by calculating the optimal race behaviour on a track with Model Predictive Control (MPC).

1.2 Research Problem

This thesis describes the optimal racing behaviour of a go-kart on a track in terms of the optimal trajectory on the track and the optimal speed at each point to travel the track as fast as possible. To this end, path and trajectory planning will be performed by MPC.

A *path* is a geometric trace that the vehicle should follow to reach its destination whilst adhering to the motion constraints such as road boundaries [7]. The configuration vector defines the set of independent attributes in the coordinate system that together define the position and orientation of the vehicle. Path planning is therefore the problem of finding a geometric path from an initial configuration to a final configuration such that each intermediate configuration is valid and feasible.

On the other hand, a *trajectory* is defined as a sequence of states visited by the vehicle, parametrised by time or velocity [7]. So, a trajectory is a path plus a schedule, describing how quickly a vehicle should move along the path and on what time the vehicle should be at a certain position. These states could be defined in multiple ways, by different quantities, but it is important that the quantities together, uniquely, indicate a specific configuration of the system. For example, the position on the road, the current velocity vector and its direction. Therefore, trajectory planning (or trajectory generation or motion planning) is the problem of planning the vehicle's transition from one state to the next whilst adhering to the kinematic limits of the vehicle and the motion constraints as road boundaries.

MPC is a method in control engineering and it is suitable for path and trajectory planning, since it uses a kinematic or dynamic model of the vehicle to retrieve samples of the vehicle's future motion (controller inputs) [7]. From the model and controller inputs, the best trajectory for the vehicle is obtained. A weakness of MPC is that it gets harder to optimise trajectories when more variables are used to model the vehicle. On the other hand, a strength of MPC is that the model could be extended in the future to also handle other opponents or obstacles on the road to evaluate races which are more complex than one-player races (which is the scope of this thesis).

This leads to the following main question:

How can the fastest path and the corresponding velocities of autonomous go-karts be determined on a given track using Model Predictive Control?

1.3 Contributions of the Thesis

In this thesis a framework is provided to determine the optimal path and maximal attainable speed on a given race course by path and trajectory planning using MPC. First, a kinematic bicycle model is presented, which simulates the vehicles responses and is used in the derivation of the controller algorithm. Next, the controller algorithm is presented and implemented in a computer program to obtain the best trajectory for the vehicle.

This can be translated to the following contributions:

- First, a kinematic bicycle model is formulated;
- Next, a controller algorithm is formulated that calculates the optimal trajectory based on road and vehicle constraints;
- Finally, the implication of the trajectory generating algorithm is visualized by programming the algorithm and executing it on a manually created race track.

2 Theory

This section provides relevant theory to help understand the other chapters of the thesis.

2.1 Vehicle Model Strategy

Vehicle modelling is often done in the vehicle axis system as presented in Figure 1 (note that some letters are used differently in the rest of this thesis) [8]. The origin in this system is formed by the center of gravity (CG). The three axes (roll, pitch and yaw) coincide at the center of gravity.

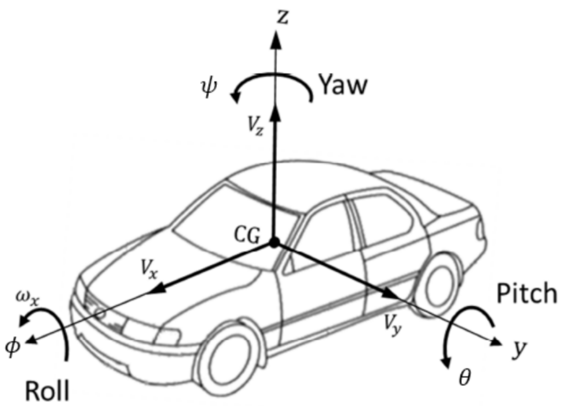


Figure 1: Vehicle Axis System from the ISO 8855-2011 as given in [8]

Vehicle models can generally be categorised as geometric, kinematic or dynamic. *Geometric models* only consider geometric dimensions and are considered to be the most simple of the three types. *Kinematic models* study the motion of the vehicle using its geometry, regardless of the forces and torque that cause it. *Dynamic models*, on the other hand, study the relation between the applied forces/ torques and the resulting motion.

This thesis will use a kinematic model of the vehicle. In general, a kinematic model provides a mathematical description of the vehicle's motion without considering the forces that affect the motion, but solely considering the vehicle's geometry. Despite its simplicity, the model is able to describe important aspects like vehicle velocity, lateral acceleration and yaw motion in terms of the local and global axes. Therefore, this modelling type can be found in most of the studies on path tracking control [9]. However, neglecting the forces that affect the motion is equivalent to assuming there is no resistance on the wheels in another orientation than the driving direction, i.e. there is no slip [10]. This is reasonable for low speed motion of the vehicle (e.g. for speeds less than 5 m/s), but not for racing conditions. Hence, a dynamical model would be more appropriate for a vehicle in racing conditions, where the tire forces are significant [11]. In Appendix C an elaboration can be found on slip.

In all categories, the kart can be modelled as full vehicle or half vehicle (so-called bicycle model). The bicycle model is a model of the go-kart in which the two front and rear wheels are replaced by one front and rear wheel, respectively. Even though it is not uncommon for four-wheeled vehicles to be described by a kinematic full vehicle model [12], this thesis will describe a kinematic bicycle model. Namely, kinematic bicycle models are less complex and still sufficient for path and trajectory planning by MCP [13]. They are commonly used to design a controller in autonomous driving.

2.2 Relevant (Vector) Calculus and Analysis

This section provides an overview of relevant mathematical concepts in (Vector) Calculus and Analysis that will be used and referred to in the coming chapters.

2.2.1 Parametrizations of Plane Curves

The curve or path traced by a particle moving in the xy-plane is not always the graph of a function or single equation $y = f(x)$, see for example Figure 2. Here, multiple y values belong to a specific x value. In this case, there is the need to represent the curve differently, namely by a pair of parametric equations:

Definition 2.1 (Parametric curve and equations)**.** If x and y are given as functions

$$x = x(t), y = y(t)$$

over an interval I of t-values, then the set of points $(x, y) = (x(t), y(t))$ defined by these equations is a parametric curve C. [14]

□

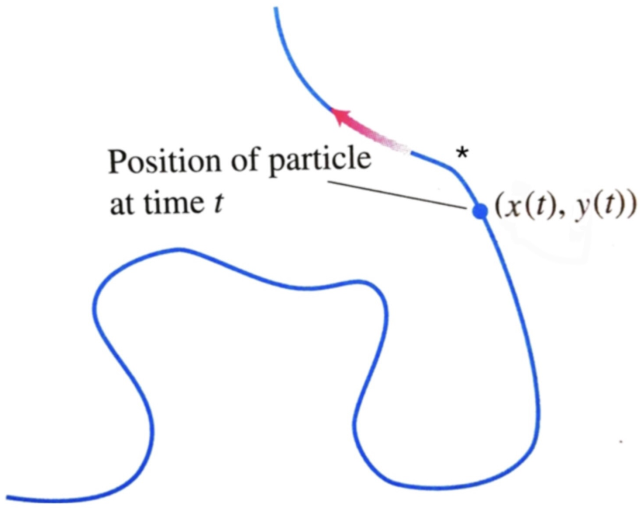


FIGURE 2: The curve or path traced by a particle is not always the graph of a function or single equation [14]

The variable t is a parameter for the curve, and its domain I is the *parameter interval*. Let $I = [a, b]$ be a closed interval, then $(x(a), y(a))$ is the *initial point* of the curve and $(x(b), y(b))$ is the *terminal point* of the curve C specified by $(x(t), y(t)), t \in I$. The equations and interval together constitute a *parametrization* of the curve. When studying motion, t usually denotes time.

Now, if a particle moves around in two-dimensional space, its motion can be described by the 2 coordinates of its position as functions of time t: $x = x(t)$ and $y = y(t)$. It could be more convenient, however, to replace these two equations by a single vector equation:

$$\mathbf{\Gamma} = \mathbf{\Gamma}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} \quad (t \in I). \tag{1}$$

In terms of the standard basis vectors $\mathbf{i}$ and $\mathbf{j}$, the position of the particle at time t is:

$$\mathbf{\Gamma} = \mathbf{\Gamma}(t) = x(t)\mathbf{i} + y(t)\mathbf{j}. \tag{2}$$

If the component functions $x(t)$ and $y(t)$ are continuous functions of t, the particle follows a continuous curve and $\mathbf{\Gamma}(t)$ is a continuous vector function of t. Continuous curves possess interesting properties, such as smooth curves which are defined as follows:

Definition 2.2 (Smooth curve)**.** Let C be a curve with parametrisation $\Gamma(t)$, $t \in I$, where $x(t)$ and $y(t)$ are *continuously differentiable* (meaning they have continuous first derivatives) on the interval I and the derivatives $x'(t)$ and $x'(t)$ are not simultaneously zero, then C is a *smooth curve* [14].

□

Smooth curves are free of corners and cusps, see Figure 3.

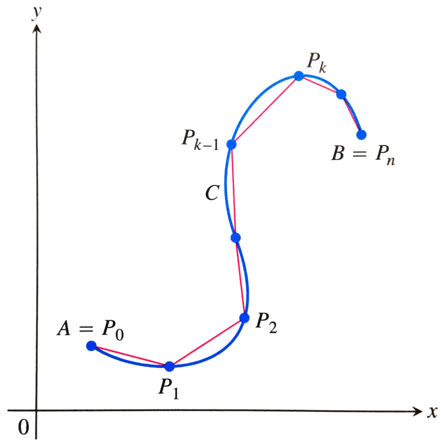


FIGURE 3: The curve or path traced by a particle is not always the graph of a function or single equation [14]

In the figure above, a smooth curve C is presented and the path (or arc) AB is subdivided into n pieces at points $A = P_0, P_1, P_2, ..., P_n = B$. The length of AB can be approximated by the sum of the straight line segments between these points. Each of these line segments have length (see Figure 4):

$$\ell_k = \sqrt{(\Delta x_k)^2 + (\Delta y_k)^2}. \tag{3}$$

Therefore, when the number of segments $n \to \infty$, it is reasonable to interpret the sum of line segments as the arc length and so to define the length of the curve from A to B as a definite integral.

Definition 2.3 (Arc length)**.** If a smooth curve C is defined parametrically by $x = x(t)$ and $y = y(t), a \leq t \leq b$, then the *length of* C is the definite integral:

$$\ell = \int_a^b \sqrt{[x'(t)]^2 + [y'(t)]^2} dt,$$

or in Leibniz notation:

$$\ell = \int_a^b \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2} dt.$$

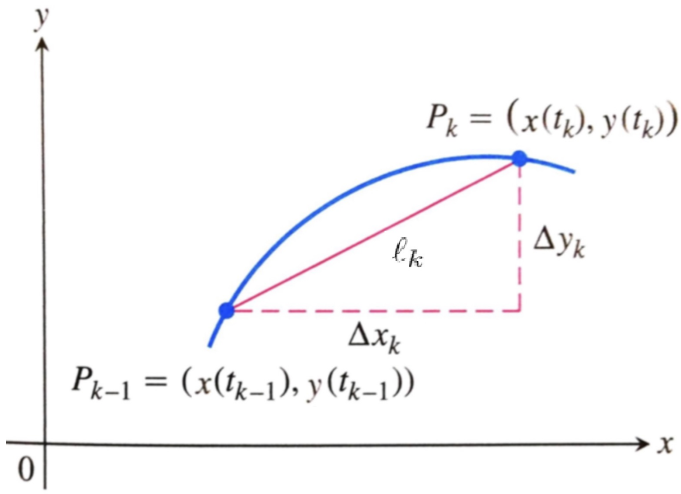


FIGURE 4: The curve or path traced by a particle is not always the graph of a function or single equation

The arc length function for the parametrically defined curve C can be defined by:

$$s(t) = \int_a^t \sqrt{\left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2} d\tau. \tag{4}$$

□

2.2.2 Curvature

The concept of *curvature* can be interpreted as how much the curve direction changes over a small distance travelled on the curve. This makes sense intuitively, as we say from two circles with a different radius that the smaller circle is "more curved". Hence, the intuitive concept of curvature from a circle can be defined as follows:

Definition 2.4 (Curvature circle)**.** The curvature κ of a circle with radius R is:

$$\kappa = \frac{1}{R}.$$

□

Consider Figure 5, where a circle is presented with the tangents at P and P_1. Denote the arc length measured from A by s. Then, the length of the arc PP_1 is Δs. Furthermore, denote the angle between the tangent at P and the horizontal axis by $\alpha(s)$. If the change of direction from the center M to the circle at P and P_1 is $\Delta\alpha(s)$, then it follows from $\Delta s = R\Delta\alpha(s)$:

$$\kappa = \frac{1}{R} = \frac{\Delta\alpha(s)}{\Delta s}.$$

So, for a circle the curvature is equal to the change of direction of the tangent per unit arc length. For an arbitrary smooth curve, this change of direction is not constant. However, it is possible to consider this quantity as the *average curvature* between P and P_1. Hence,

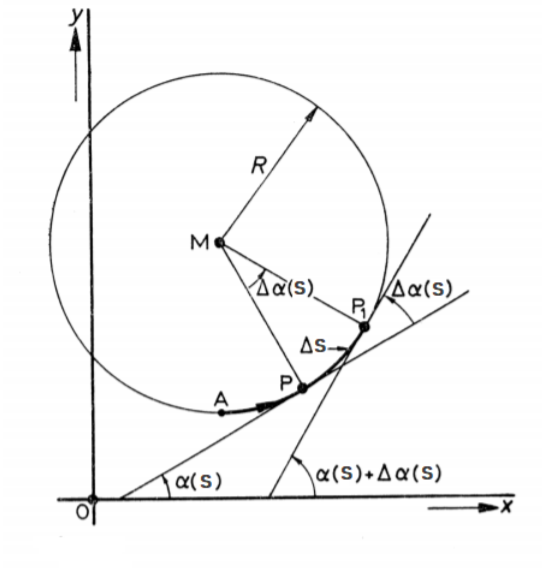


FIGURE 5: The curvature of a circle

in Figure 6, the quotient $\frac{\Delta\alpha(s)}{\Delta s}$ is the average curvature of the arc between P and P_1. If the distance between P_1 and P decreases and approaches zero, the average curvature will hardly change and if it approaches a limit, this limit is called the curvature in P.

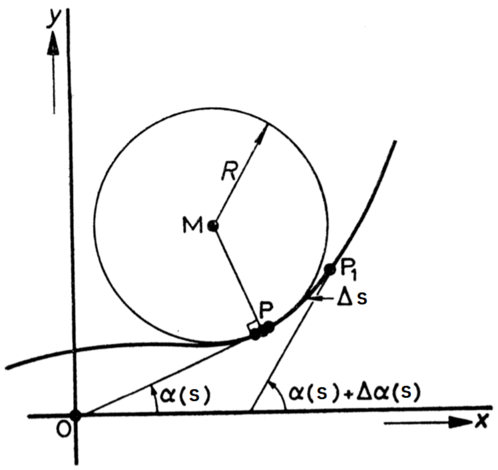


FIGURE 6: The curvature of a smooth curve

Definition 2.5 (Curvature smooth curve)**.** The curvature $\kappa(s)$ in P on a smooth curve with arc length s is the limit of the average curvature between P and P_1 on the curve if P_1 approaches P:

$$\kappa(s) = \lim_{\Delta s \to 0} \frac{\Delta\alpha(s)}{\Delta s} = \frac{d\alpha}{ds},$$

where α denotes the angle between the tangent of the curve at P and the horizontal axis.

□

For the definition above, it is important for the curve to be smooth in order for α to be differentiable to s. If α is an increasing function of s, then $\kappa(s)$ is positive. Similarly, if α is a decreasing function of s, then $\kappa(s)$ is negative.

For a circle, the inverse of the curvature is the radius (Def. 2.4). Similarly, for any random smooth curve the absolute value of the inverse of the curvature is the radius of an imaginary circle. The radius of this circle is the *radius of curvature* $R(s)$:

$$R(s) = \frac{1}{|\kappa(s)|}. \tag{5}$$

2.3 Model Predictive Control

Model predictive control (MPC) is a method of process control which is based on iterative, finite-horizon optimization of a plant model. At a specific time, the current state is evaluated and a cost minimizing strategy is computed for a relatively short time horizon in the future. This is then repeated for the next moment in time close to the last moment. The prediction horizon is shifted forwards and therefore MPC is also called receding horizon control. MPC uses an internal dynamic model of the process, a cost function J over the receding horizon N and an optimization algorithm minimizing the cost function J using the control input u. More specifically, the structure of MPC is as follows:

1. At time k, compute the optimal control $u(k)$ for $k, k+1, k+2, ..., k+N$ by solving the optimization problem J for the prediction horizon N;
2. Apply the first value of the computed control sequence;
3. At time $k+1$, retrieve the system-state and re-compute.

For clarity, regard the following abstract example of an MPC problem:

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & J = \sum_{t=k+1}^{k+N} (x(t))^2 + (u(t))^2 \\ \text{subject to} \quad & x(t+1) = Ax(t) + Bu(t), \\ & u(t) \leq u_0, \end{aligned} \tag{6}$$

where $x(t)$ denotes the state of the system at step t, $u(t)$ denotes the input to the plant at step t and A and B are used to predict the next state of the plant using the current state and input. The upper limit of the input is u_0, creating a constraint. The cost function J looks N steps into the future and collects at each step a value $(x(t))^2 + (u(t))^2$.

MPC is well suited for vehicle control, since it takes vehicle dynamics into account and performs trajectory planning and execution at each step, so that sudden developments such as the position of other vehicles can be taken into account. By using MPC, optimal control inputs to the model can be determined based on minimzation of a certain objective function. This objective function is subjected to constraints set by physical limits of the vehicle or the track environment.

When nonlinear system models are used in MPC, the control problems on a finite prediction horizon need not be convex anymore. This challenges real-time feasibility. Therefore, it is often beneficial to linearize and discretize the dynamic equations of a system to allow the problem to be formulated as a convex optimization problem.

3 Problem Formulation

This chapter focuses on formulating the problem. First, the road model will be presented, followed by the vehicle model. Using the model, the state representation for the system is given and then the differential equations are derived to describe the vehicle dynamics.

3.1 Road Model

The road is modelled in two dimensions from a top-view, see Figure 7.

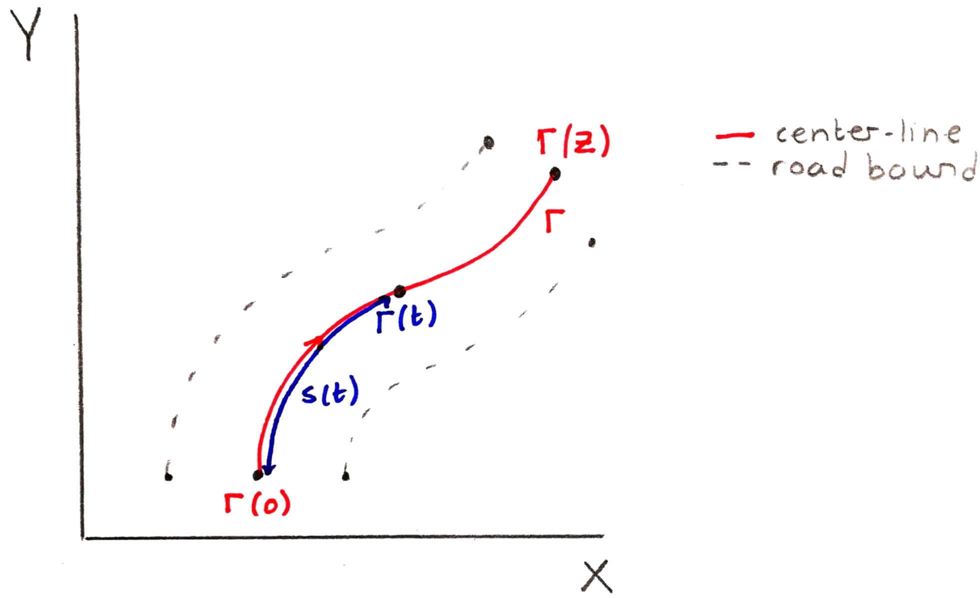


FIGURE 7: Sketch of the road parametrized by time

In the figure above, the road bounds and the center-line (the path Γ) are indicated. The center-line is a smooth curve (Section 2.2.1) and it is directed, since there is a specific driving direction on the road. The path can be denoted as a vector parametrized by time (t) as follows (see Section 2.2.1) :

$$\boldsymbol{\Gamma}(t) = x(t)\mathbf{i} + y(t)\mathbf{j}, \quad 0 \leq t \leq Z. \tag{7}$$

Each point $\Gamma(t) = (x(t), y(t))$ has arc length parameter $s(t)$ (see Section 2.2.1):

$$s(t) = \int_0^t \sqrt{\left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2} d\tau. \tag{8}$$

The curve $\boldsymbol{\Gamma}$ is parametrized with regard to parameter t, but it can be reparametrized in terms of s by substituting for t: $\boldsymbol{\Gamma} = \boldsymbol{\Gamma}(t(s))$. This is under the assumption that the vehicle always has a forward (positive) velocity. This is reasonable, considering the vehicle is in a racing environment. Then, with the arc length function $s(t)$, it is possible to solve for t as function of s: $t = t(s)$.

This reparametrization introduces a second coordinate system: the *curvilinear system*. Any point P on the road can be projected onto a point $\bar{P}$ at the center-line Γ, with $\bar{P}$

being the point on Γ closest to P (see Figure 8). The curvilinear abscissa is the arc length parameter s of $\bar{P}$ and the curvilinear ordinate is the *"directed distance"* d between P and $\bar{P}$. The absolute value of d is the distance between P and $\bar{P}$. The ordinate d is positive if P is on the left side of the road and negative if it is on the right side of the road. This is equivalent to P being to the left or right, respectively, of $\bar{P}$, seen from the road heading in its direction. More formally, the curvilinear ordinate d is defined by the Cartesian coordinates of $P = (x_P, y_P)$ and $\bar{P} = (x_{\bar{P}}, y_{\bar{P}})$:

$$d = \begin{cases} \sqrt{(x_P - x_{\bar{P}})^2 + (y_P - y_{\bar{P}})^2}, & \text{if } P \text{ to the left of } \bar{P} \\ -\sqrt{(x_P - x_{\bar{P}})^2 + (y_P - y_{\bar{P}})^2}, & \text{otherwise.} \end{cases} \tag{9}$$

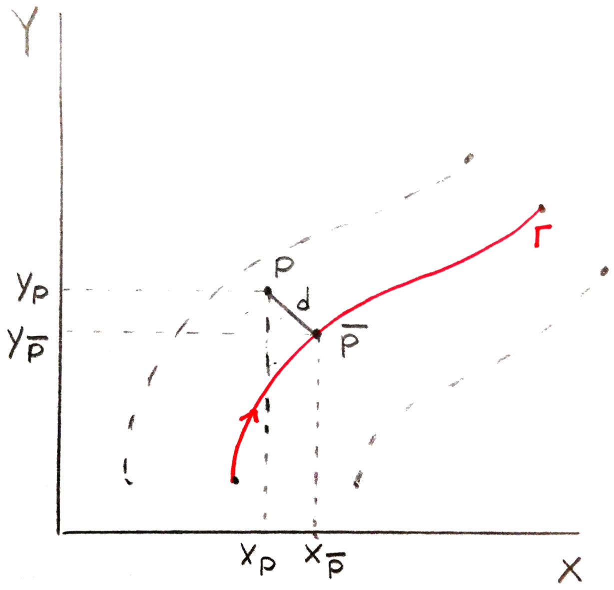


FIGURE 8: Sketch of the road with point P and its projection $\bar{P}$ on center-line Γ

3.2 Vehicle Model

The vehicle (the go-kart) is modelled as a bicycle with a kinematic model (see Section 2.1). Consider Figure 9, which illustrates the model with its variables (an overview of all variables with their unit and range can be found in Appendix A). The figure is a representation of the system frozen at a specific time t or place s. So, even though the parameters seem to have a distinct value in the figure, this only holds for t or s and they are actually time- or space-dependent.

The bicycle has a front wheel F and rear wheel R at a distance L. The bicycle is placed on a road with center-line Γ. The point on Γ closest to R is $\bar{P}$ and the directed distance (see Section 3.1) between R and $\bar{P}$ is d. At point $\bar{P}$, the center-line has angle θ_Γ with respect to the global X-axis. At point $\bar{P}$ the center-line has a circular motion and the center of that motion is indicated by point M. The radius of curvature is ρ and the angular velocity is $\dot{\theta}_\Gamma$. The vehicle is driving forwards with *velocity* v and the *projection of* v *in the direction of the path* is denoted v_Γ. The *angle between the current position of the vehicle and the global X-axis* is θ_G and the *angle between the vehicle and the direction of the path at* P is θ.

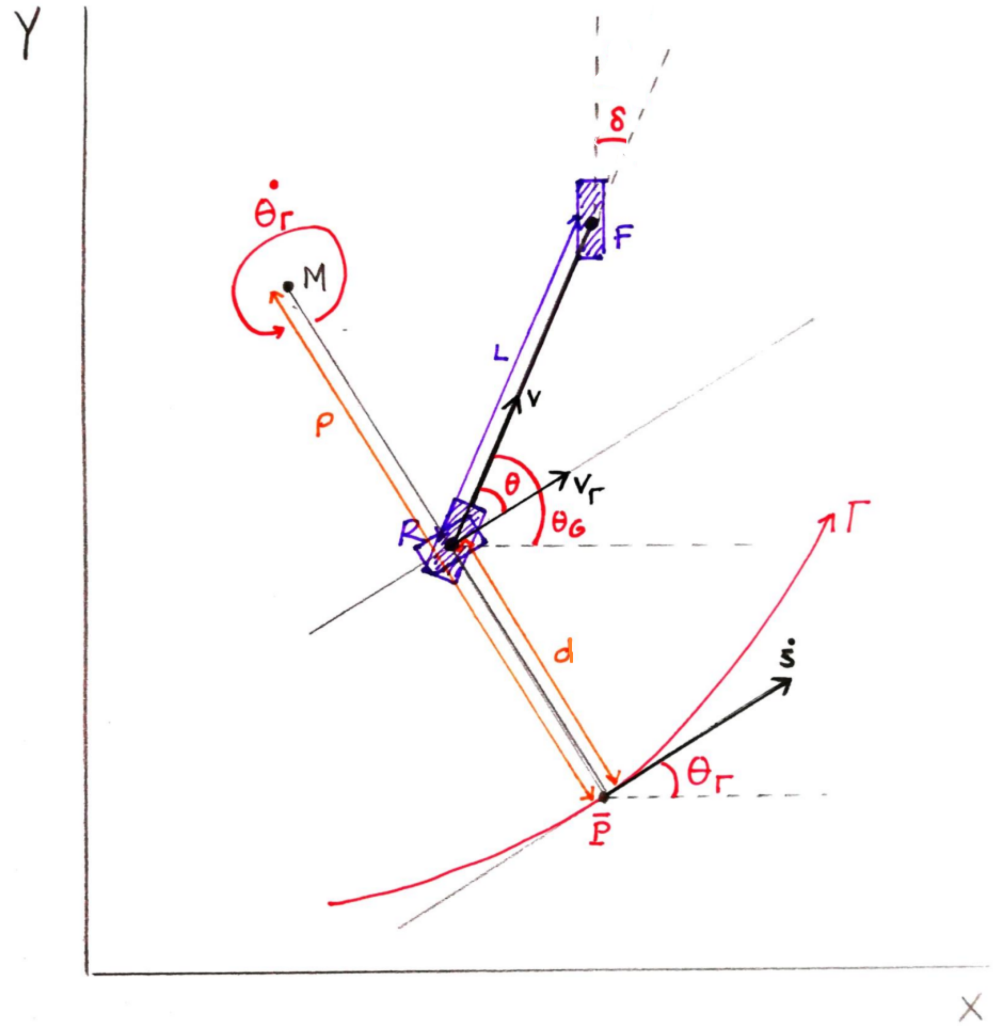


FIGURE 9: Extended kinematic model of the vehicle as a bicycle

The vector $\dot{s}$ indicates the progression along the center-line of the track. The front wheel of the vehicle is turned δ degrees.

Furthermore, the curvature of the path the vehicle is travelling is denoted κ_G and the curvature of the center-line at $\bar{P}$ is κ_Γ (with $\kappa_\Gamma = \frac{1}{\rho}$, see Section 2.2.2).

3.3 State Representation

The *state vector* $\zeta = [d, \theta, v, \kappa_G]^T$ is defined as a 4-dimensional vector with a meaning as described in Table 1. These four quantities can, together, uniquely describe the system. Intuitively, they describe where the vehicle is, how it is placed on the road, how fast it is currently going and how the wheels are turned. Along with these quantities, the time t or place s are also known.

The *input vector* $u = [a, c_G]^T$ is defined as a 2-dimensional vector with a meaning as described in Table 2. The longitudinal acceleration a and vehicle steering c_G are the input quantities of the system, since the vehicle can be controlled by accelerating/decelerating and steering the wheel.

State	Unit	Description
d	m	Deviation from center-line
θ	rad	Yaw angle relative to path
v	m/s	Longitudinal velocity
κ_G	m^{-1}	Vehicle curvature

TABLE 1: Spatial states of the vehicle model

Control	Unit	Description
a	m/s^2	Longitudinal acceleration
c_G	rad$/ms$	Vehicle steering

TABLE 2: Control inputs of the vehicle model

3.4 Dynamics

For any variable x, its time derivative (with respect to t) will be denoted by $\dot{x}$ and its spatial derivative (with respect to s) will be denoted by x'. All quantities in the state vector ζ (recall: d, θ, v, κ_G) will first be derived with respect to t and then with respect to s.

3.4.1 Time Dynamics

First, the time derivative of the longitudinal velocity v is the *longitudinal acceleration* a after taking friction into account with the *longitudinal friction coefficient* μ as follows:

$$\dot{v} = a - \mu v. \tag{10}$$

The longitudinal acceleration is one of the input variables and the longitudinal friction coefficient is a parameter depending on the type and conditions of the tires and the road [15].

Secondly, the time derivative of the vehicle curvature is the vehicle steering as follows:

$$\dot{\kappa}_G = c_G. \tag{11}$$

For verification, both units of $\dot{\kappa_G}$ and c_G are rad$/ms$. That is, how many radians the wheels are turned per meter, per second.

Thirdly, regard $\dot{d}$ (the time derivative of d) as the vector $v - v_\Gamma$. Then:

$$\dot{d} = v \sin \theta. \tag{12}$$

Finally, regard the yaw angle θ of the vehicle with respect to the path. This is the difference between the yaw angle of the vehicle with respect to the global X-axis (θ_G) and the yaw angle of the center-line of the road at $\bar{P}$ with respect to the global X-axis (θ_Γ):

$$\theta = \theta_G - \theta_\Gamma. \tag{13}$$

Hence (by the sum rule in differentiation),

$$\dot{\theta} = \dot{\theta}_G - \dot{\theta}_\Gamma. \tag{14}$$

Both terms ($\dot{\theta}_G$ and $\dot{\theta}_\Gamma$) will be derived in the following two paragraphs.

Derivation Yaw Angle of Vehicle to Global X-axis
This paragraph focuses on deriving $\dot{\theta}_G$.

Consider Figure 10, which is a simplification of Figure 9. It contains the vehicle as a bicycle with R and F. It has current velocity v and the wheels are pointed δ rad.

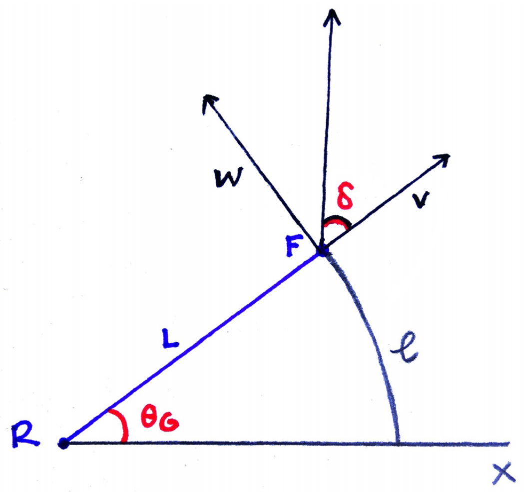


FIGURE 10: Bicycle model of vehicle with length L, current velocity v and angle δ of the front wheels

The arc-length from the global X-axis around the circle to F (the front wheel) is $l = L\theta_G$. Furthermore, the tangential velocity $\frac{dl}{dt} = w = L\dot{\theta}_G$. Namely, the tangential velocity at F is the angular velocity proportional to its distance to the axis of rotation.

So,

$$\dot{\theta}_G = \frac{w}{L} = \frac{v \tan(\delta)}{L}. \tag{15}$$

This equation is already valid, but it can be expressed shorter when relating the vehicle curvature κ_G to the steering angle δ. Therefore we consider Figure 11.

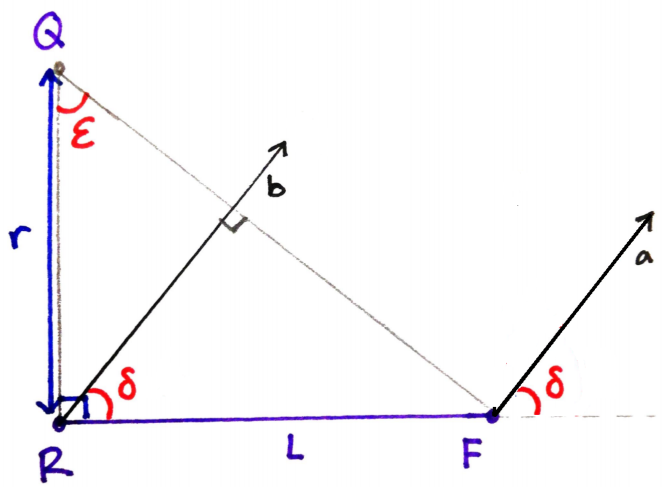


FIGURE 11: Sketch of bicycle model for derivation vehicle curvature

Here, the front and rear wheel are presented and the front wheel is turned at angle δ. Consequently, the vehicle will drive in direction a which is under the angle δ with regard to the current position. The translated vector b of a on the rear wheel also has angle δ with regard to the current position. The angle ϵ is the angle between R and F at the center of the (imaginary) circle the rear wheel will drive. This angle is equal to δ (since $RF \perp QR$). Hence,

$$\tan(\epsilon) = \tan(\delta) = \frac{L}{r} \tag{16}$$

and therefore

$$\kappa_G = \frac{1}{r} = \frac{\tan(\delta)}{L}. \tag{17}$$

Finally, substituting Equation (17) into Equation (15) gives:

$$\dot{\theta}_G = \kappa_G v. \tag{18}$$

Derivation Yaw Angle of Road to global X-axis
This paragraph focuses on deriving $\dot{\theta}_\Gamma$.

Notice that the angular velocity can be used to derive the tangential velocities v_Γ and $\dot{s}$ as follows (see Figure 9):

$$\dot{s} = \rho\dot{\theta}_\Gamma \tag{19a}$$

$$v_\Gamma = (\rho - d)\dot{\theta}_\Gamma. \tag{19b}$$

First, consider Equation (19b). The vehicle velocity in the direction of the path can also be expressed geometrically using yaw angle θ as follows:

$$v_\Gamma = v\cos(\theta). \tag{20}$$

Combining these two (Equation (19b) and (20)) gives:

$$\dot{\theta}_\Gamma = \frac{v\cos\theta}{\rho - d}. \tag{21}$$

Rewriting this equation by multiplying the nominator and denominator with κ_Γ (remember, $\kappa_\Gamma = \frac{1}{\rho}$) gives:

$$\dot{\theta}_\Gamma = \frac{\kappa_\Gamma v\cos\theta}{1 - d\kappa_\Gamma}. \tag{22}$$

Finally, substituting Equation (18) and (22) into Equation (14) gives:

$$\dot{\theta} = \dot{\theta}_G - \dot{\theta}_\Gamma = \kappa_G v - \frac{\kappa_\Gamma v\cos\theta}{1 - y\kappa_\Gamma}. \tag{23}$$

The vehicle dynamics of the states are now derived as functions of the control inputs and other known variables. They are given one more time in the overview below.

$$\dot{d} = v\sin\theta \tag{24a}$$

$$\dot{\theta} = \kappa_G v - \frac{\kappa_\Gamma v\cos\theta}{1 - d\kappa_\Gamma} \tag{24b}$$

$$\dot{v} = a - \mu v \tag{24c}$$

$$\dot{\kappa}_G = c_G. \tag{24d}$$

3.4.2 Spatial Dynamics

Rewriting the time dynamics to spatial dynamics can be achieved by the following derivation rule (chain rule):

$$\zeta'(s) := \frac{d\zeta}{ds} = \frac{d\zeta}{dt}\frac{dt}{ds}. \tag{25}$$

Considering the racing conditions, it is reasonable to assume positive velocity and therefore $\dot{s} \neq 0$. This gives $\frac{dt}{ds} = \frac{1}{\dot{s}}$. Therefore, the spatial states can be derived using the time states and $\dot{s}$ as follows:

$$\zeta'(s) = \dot{\zeta}\frac{1}{\dot{s}}. \tag{26}$$

Revise Equation (19a) and substitute Equation (21) to obtain:

$$\dot{s} = \rho\dot{\theta}_\Gamma = \rho\frac{v\cos\theta}{\rho - d} = \frac{\rho v\cos\theta}{\rho - d} = \frac{v\cos\theta}{1 - d/\rho} = \frac{v\cos\theta}{1 - d\kappa_\Gamma}. \tag{27}$$

Substituting Equation (27) in Equation (26) gives the following set of spatial vehicle dynamics:

$$d' = \frac{\dot{d}}{\dot{s}} = v\sin\theta \cdot \frac{1 - d\kappa_\Gamma}{v\cos\theta} = (1 - d\kappa_\Gamma)\tan\theta \tag{28a}$$

$$\theta' = \frac{\dot{\theta}}{\dot{s}} = \left(\kappa_G v - \frac{\kappa_\Gamma v\cos\theta}{1 - d\kappa_\Gamma}\right)\cdot\frac{1 - d\kappa_\Gamma}{v\cos\theta} = \kappa_G\frac{1 - d\kappa_\Gamma}{\cos\theta} - \kappa_\Gamma \tag{28b}$$

$$v' = \frac{\dot{v}}{\dot{s}} = (a - \mu v)\cdot\frac{1 - d\kappa_\Gamma}{v\cos\theta} = a\frac{1 - d\kappa_\Gamma}{v\cos\theta} - \mu\frac{1 - d\kappa_\Gamma}{\cos\theta} \tag{28c}$$

$$\kappa_G' = \frac{\dot{\kappa}_G}{\dot{s}} = c_G\frac{1 - d\kappa_\Gamma}{v\cos\theta}. \tag{28d}$$

4 Controller Design

An overview of the control structure can be found in Figure 12. Here, the computing unit that performs Model Predictive Control (MPC) receives the coordinates of a reference path. It also receives information of the vehicle (its current position on the road) and determines the optimal vehicle steering and longitudinal acceleration/deceleration taking the objective function and constraints into account. These inputs are given to the vehicle model that will causes it to reach a new state.

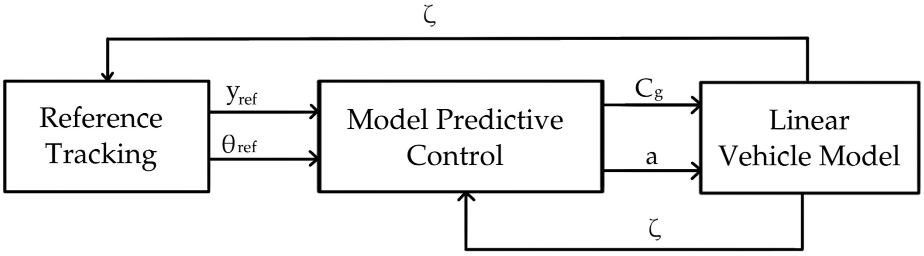


FIGURE 12: Overview control structure

The horizon will be cut into N points and at each point k, the control structure of above is executed. The vehicle model will be simplified for this end, by linearizing and discretizing, to guarantee feasibility at each discrete sampling time.

More specifically, the controller should solve the following generic optimal control problem over a finite horizon:

$$\begin{aligned} \max \quad & \text{Progress along a given path} \\ \text{s.t.} \quad & \text{Vehicle model,} \\ & \text{Actuation limits,} \\ & \text{Handling limits,} \\ & \text{Driving corridor.} \end{aligned} \tag{29}$$

Each of the components will be specified in the coming sections.

4.1 Progress Maximization

The objective of the controller is to get the vehicle to cross the finish line as fast as possible. In other words, the progression along the center-line of the track ($\dot{s}$) must be maximized. However, since minimization problems are more common in practice, the problem will be denoted as minimization of $\frac{1}{\dot{s}}$ (which is valid under the assumption $\dot{s} \neq 0$). That gives, considering Equation (27), the following objective function f:

$$f(y, \theta, v) := \frac{1}{\dot{s}} = \frac{1 - y\kappa_\Gamma}{v \cos\theta}, \tag{30}$$

where κ_Γ is determined for every s.

As explained in Section 2.3, it is important for the objective function to be convex to have a real-time implementation. Therefore, f is approximated by a second order Taylor series as follows (with $\mathbf{x} = \langle d, \theta, v \rangle$ and $\mathbf{a} = \langle d_{ref}, \theta_{ref}, v_{ref} \rangle$):

$$f(\mathbf{x}) \approx f(\mathbf{a}) + [(\mathbf{x} - \mathbf{a}) \cdot \nabla f(\mathbf{a})] + [(\mathbf{x} - \mathbf{a})^T \cdot (H(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{a}))], \tag{31}$$

where $\nabla f(\mathbf{a})$ denotes the gradient of f at $\mathbf{a}$ and H is the Hessian matrix (the matrix of second derivatives). More specifically:

$$\nabla f(\mathbf{a}) = \left[\begin{array}{ccc} \frac{\partial f}{\partial d}(\mathbf{a}) & \frac{\partial f}{\partial \theta}(\mathbf{a}) & \frac{\partial f}{\partial v}(\mathbf{a}) \end{array}\right]^T \tag{32}$$

and

$$H(d,\theta,v) = \begin{bmatrix} f_{dd}(d,\theta,v) & f_{d\theta}(d,\theta,v) & f_{dv}(d,\theta,v) \\ f_{\theta d}(d,\theta,v) & f_{\theta\theta}(d,\theta,v) & f_{\theta v}(d,\theta,v) \\ f_{vd}(d,\theta,v) & f_{v\theta}(d,\theta,v) & f_{vv}(d,\theta,v) \end{bmatrix}. \tag{33}$$

The first order derivatives in the gradient of f are:

$$f_d(d,\theta,v) = -\frac{\kappa_\Gamma}{v\cos\theta} \tag{34a}$$

$$f_\theta(d,\theta,v) = \frac{1-d\kappa_\Gamma}{v}\frac{\sin\theta}{(\cos\theta)^2} = \frac{1-d\kappa_\Gamma}{v}\tan\theta\sec\theta \tag{34b}$$

$$f_v(d,\theta,v) = -\frac{1-d\kappa_\Gamma}{v^2\cos\theta}. \tag{34c}$$

The second order derivatives in the Hessian matrix are:

$$f_{dd}(d,\theta,v) = 0 \tag{35a}$$

$$f_{\theta\theta}(d,\theta,v) = \frac{(1-d\kappa_\Gamma)\sec\theta((\tan\theta)^2+(\sec\theta)^2)}{v} \tag{35b}$$

$$f_{vv}(d,\theta,v) = \frac{2(1-d\kappa_\Gamma)}{v^3\cos\theta} \tag{35c}$$

$$f_{d\theta}(d,\theta,v) = f_{\theta d}(d,\theta,v) = -\frac{\kappa_\Gamma}{v}\tan\theta\sec\theta \tag{35d}$$

$$f_{dv}(d,\theta,v) = f_{vd}(d,\theta,v) = \frac{\kappa_\Gamma}{v^2\cos\theta} \tag{35e}$$

$$f_{\theta v}(d,\theta,v) = f_{v\theta}(d,\theta,v) = -\frac{1-d\kappa_\Gamma}{v^2}\tan\theta\sec\theta. \tag{35f}$$

The Taylor approximation will be done with respect to $\theta_{ref} = 0$ and $v_{ref} \neq 0$. Namely, it is assumed throughout the thesis that $v \neq 0$, since the vehicle is in a racing environment. Furthermore, $\theta_{ref} = 0$ minimizes the angle between the heading of the vehicle and the heading of the path. It encourages the vehicle to travel in the same direction as the path and therefore maximise progress.

Filling in these linearization points in the derivatives in Equations (34) and (35) cancels $f_\theta, f_{d\theta}, f_{\theta d}, f_{\theta v}$ and $f_{v\theta}$, since they are equal to zero (as $\tan(0) = 0$).

Then, this leaves Equation (31) as follows:

$$\begin{aligned} f(d,\theta,v) \approx\, & f(d_{ref},\theta_{ref},v_{ref}) + f_d(d_{ref},\theta_{ref},v_{ref})(d-d_{ref}) + \\ & f_v(d_{ref},\theta_{ref},v_{ref})(v-v_{ref}) + f_{dv}(d_{ref},\theta_{ref},v_{ref})(d-d_{ref})(v-v_{ref}) + \\ & \frac{1}{2}\left[f_{\theta\theta}(d_{ref},\theta_{ref},v_{ref})(\theta-\theta_{ref})^2 + f_{vv}(d_{ref},\theta_{ref},v_{ref})(v-v_{ref})^2\right]. \end{aligned} \tag{36}$$

After filling in the linearization points in the remaining derivatives, Equation (36) becomes:

$$\begin{aligned} f(d,\theta,v) \approx & f(d_{ref},\theta_{ref},v_{ref}) + \frac{-\kappa_\Gamma}{v_{ref}}(d-d_{ref}) + \\ & \frac{-(1-d_{ref}\kappa_\Gamma)}{v_{ref}^2}(v-v_{ref}) + \frac{\kappa_\Gamma}{v_{ref}^2}(d-d_{ref})(v-v_{ref}) + \\ & \frac{1}{2}\left[\frac{1-d_{ref}\kappa_\Gamma}{v_{ref}}(\theta-\theta_{ref})^2 + \frac{2(1-d_{ref}\kappa_\Gamma)}{v_{ref}^2}(v-v_{ref})^2\right]. \end{aligned} \tag{37}$$

Now, minimizing $\frac{1}{\dot{s}}$ (or $f(d,\theta,v)$) is equal to the optimization problem:

$$\min \gamma_1 d + \gamma_2 v + \gamma_3\theta^2 + \gamma_4 vd + \gamma_5 v^2, \tag{38}$$

where the terms that do not depend on model states or inputs are excluded. The values for $\gamma_1, \gamma_2, \gamma_3, \gamma_4$ and γ_5 can be retrieved by rewriting Equation (37) and are defined as follows:

$$\gamma_1 = -\frac{2\kappa_\Gamma}{v_{ref}} \tag{39a}$$

$$\gamma_2 = -\frac{1}{v_{ref}^2} \tag{39b}$$

$$\gamma_3 = \frac{1-d_{ref}\kappa_\Gamma}{2v_{ref}} \tag{39c}$$

$$\gamma_4 = \frac{\kappa_\Gamma}{v_{ref}^2} \tag{39d}$$

$$\gamma_5 = \frac{1-d_{ref}\kappa_\Gamma}{v_{ref}^3} \tag{39e}$$

with the additional requirement $d_{ref}\kappa_\Gamma \neq 1$.

Namely, when $d_{ref}\kappa_\Gamma = 1$, then $d_{ref} = 1/\kappa_\Gamma$, so the vehicle is exactly at the center of the imaginary circle from the curvature of the road. This is a unique point, comparable to the center of a roundabout, on which the vehicle is not permitted to drive.

The objective function of the MPC controller is a simplification of the optimization problem in Eq. (38), where the final two terms ($\gamma_4 vd$ and $\gamma_5 v^2$) are omitted. During simulations, the optimization problem in Eq. (38) turned out to be non-convex and therefore unsolvable. Furthermore, the remaining terms still contain all 3 variables (d, θ and v).

Revise that the objective function of a MPC controller solves an optimization problem for each point k prediction horizon (see Section 2.3). Hence, the objective function can be denoted as follows:

$$J(d,\theta,v) = \sum_{k=0}^{N-1} \gamma_{(1,k)} d_k + \gamma_2 v_k + \gamma_{(3,k)}\theta_k^2, \tag{40}$$

where k is the discrete step length in space, N is the prediction horizon and γ_1, γ_2 and γ_3 are from Equations (39). Note that the variables d, θ and v depend on k, as well as γ_1 and γ_3, since κ_Γ depends on the place on the road.

This objective function encourages progress maximization, since it:

1. maximises the lateral deviation between the vehicle and the path with the same sign as the road curvature. That is, when the road turns to the left $\kappa_\Gamma > 0$ and, therefore, $d > 0$, since that minimizes their product $\kappa_\Gamma d$. Vice-versa when the road turns to the right.
2. maximises the longitudinal speed. Namely, γ_2 will always have a negative value, so it is beneficial for v to be as large as possible.
3. minimizes the angle between the heading of the vehicle and the heading of the path. This also makes the approximation $\theta_{ref} = 0$ valid.

4.2 Vehicle Model

From the previous chapter, we know:

$$\begin{pmatrix} d' \\ \theta' \\ v' \\ \kappa_G' \end{pmatrix} = \zeta' = f(\zeta, u) = \begin{pmatrix} f_1(\zeta,u) \\ f_2(\zeta,u) \\ f_3(\zeta,u) \\ f_4(\zeta,u) \end{pmatrix} = \begin{pmatrix} (1 - d\kappa_\Gamma)\tan\theta \\ \kappa_G \frac{1-d\kappa_\Gamma}{\cos\theta} - \kappa_\Gamma \\ a\frac{1-d\kappa_\Gamma}{v\cos\theta} - \mu\frac{1-d\kappa_\Gamma}{\cos\theta} \\ c_G\frac{1-d\kappa_\Gamma}{v\cos\theta}. \end{pmatrix} \tag{41}$$

To solve the optimization problem at each discrete sampling point, this model will be linearized and discretized (to construct a *linear space varying model*). This is a variation on linear time varying models where the vehicle dynamics are expressed in spatial coordinates instead of time. A commonly used linearization point in vehicle control is the center-line of the road, where the goal is to steer the vehicle to the center-line (i.e. $(d, \theta) = (0,0)$). However, racing lines are often different than the center-line. Often, the entire width of the track is used to increase the radius of a turn. For this reason, solutions of previous iterations from the optimization problem are used instead as reference $(\bar{\zeta}, \bar{u})$ for linearization.

For this reason, the initial state ζ_0 should be estimated by proper guess (a so-called warm start):

$$\zeta_0 = \zeta(s). \tag{42}$$

Estimating $f(\zeta, u)$ with a Taylor series around the estimated optimal solutions (i.e. the solutions of the previous iteration) given by $(\bar{\zeta}, \bar{u})$ is in general:

$$\begin{aligned} &f(\zeta,u) \approx f(\bar{\zeta},\bar{u}) + \left.\frac{\partial f(\zeta,u)}{\partial \zeta}\right|_{\bar{\zeta},\bar{u}} (\zeta - \bar{\zeta}) + \left.\frac{\partial f(\zeta,u)}{\partial u}\right|_{\bar{\xi},\bar{u}} (u - \bar{u}) \Leftrightarrow \\ &\Leftrightarrow f(\zeta,u) \approx f(\bar{\zeta},\bar{u}) + A_c(\zeta - \bar{\zeta}) + B_c(u - \bar{u}). \end{aligned} \tag{43}$$

This gives:

$$\dot{\zeta} = \begin{bmatrix} \frac{\partial f_1}{\partial d} & \frac{\partial f_1}{\partial \theta} & \frac{\partial f_1}{\partial v} & \frac{\partial f_1}{\partial \kappa_G} \\ \frac{\partial f_2}{\partial d} & \frac{\partial f_2}{\partial \theta} & \frac{\partial f_2}{\partial v} & \frac{\partial f_2}{\partial \kappa_G} \\ \frac{\partial f_3}{\partial d} & \frac{\partial f_3}{\partial \theta} & \frac{\partial f_3}{\partial v} & \frac{\partial f_3}{\partial \kappa_G} \\ \frac{\partial f_4}{\partial d} & \frac{\partial f_4}{\partial \theta} & \frac{\partial f_4}{\partial v} & \frac{\partial f_4}{\partial \kappa_G} \end{bmatrix} (\zeta - \bar{\zeta}) + \begin{bmatrix} \frac{\partial f_1}{\partial a} & \frac{\partial f_1}{\partial C} \\ \frac{\partial f_2}{\partial a} & \frac{\partial f_2}{\partial C} \\ \frac{\partial f_3}{\partial a} & \frac{\partial f_3}{\partial C} \\ \frac{\partial f_4}{\partial a} & \frac{\partial f_4}{\partial C} \end{bmatrix} (u - \bar{u}) + f(\bar{\zeta}, \bar{u}). \tag{44}$$

And this can be rewritten as follows:

$$
\begin{aligned}
\dot{\zeta} = & \begin{bmatrix} \frac{\partial f_1}{\partial d} & \frac{\partial f_1}{\partial \theta} & \frac{\partial f_1}{\partial v} & \frac{\partial f_1}{\partial \kappa_G} \\ \frac{\partial f_2}{\partial d} & \frac{\partial f_2}{\partial \theta} & \frac{\partial f_2}{\partial v} & \frac{\partial f_2}{\partial \kappa_G} \\ \frac{\partial f_3}{\partial d} & \frac{\partial f_3}{\partial \theta} & \frac{\partial f_3}{\partial v} & \frac{\partial f_3}{\partial \kappa_G} \\ \frac{\partial f_4}{\partial d} & \frac{\partial f_4}{\partial \theta} & \frac{\partial f_4}{\partial v} & \frac{\partial f_4}{\partial \kappa_G} \end{bmatrix} \zeta + \begin{bmatrix} \frac{\partial f_1}{\partial a} & \frac{\partial f_1}{\partial C} \\ \frac{\partial f_2}{\partial a} & \frac{\partial f_2}{\partial C} \\ \frac{\partial f_3}{\partial a} & \frac{\partial f_3}{\partial C} \\ \frac{\partial f_4}{\partial a} & \frac{\partial f_4}{\partial C} \end{bmatrix} u + \\
& f\left(\bar{\zeta}, \bar{u}\right) - \begin{bmatrix} \frac{\partial f_1}{\partial d} & \frac{\partial f_1}{\partial \theta} & \frac{\partial f_1}{\partial v} & \frac{\partial f_1}{\partial \kappa_G} \\ \frac{\partial f_2}{\partial d} & \frac{\partial f_2}{\partial \theta} & \frac{\partial f_2}{\partial v} & \frac{\partial f_2}{\partial \kappa_G} \\ \frac{\partial f_3}{\partial d} & \frac{\partial f_3}{\partial \theta} & \frac{\partial f_3}{\partial v} & \frac{\partial f_3}{\partial \kappa_G} \\ \frac{\partial f_4}{\partial d} & \frac{\partial f_4}{\partial \theta} & \frac{\partial f_4}{\partial v} & \frac{\partial f_4}{\partial \kappa_G} \end{bmatrix} \bar{\zeta} - \begin{bmatrix} \frac{\partial f_1}{\partial a} & \frac{\partial f_1}{\partial C} \\ \frac{\partial f_2}{\partial a} & \frac{\partial f_2}{\partial C} \\ \frac{\partial f_3}{\partial a} & \frac{\partial f_3}{\partial C} \\ \frac{\partial f_4}{\partial a} & \frac{\partial f_4}{\partial C} \end{bmatrix} \bar{u} \\
= & A\zeta + Bu + H.
\end{aligned}
\tag{45}
$$

To perform discretization in a continuous system, it is common to hold each sample value for one sample interval Δ_s and assume constant control signals in between the samples (*Zero-Order Hold*, ZOH, see [16]). Doing so, the model can be written in the following way:

$$\zeta(k+1) = A^{\mathrm{ZOH}}(k)\zeta(k) + B^{\mathrm{ZOH}}(k)u(k) + H^{\mathrm{ZOH}}(k), \quad k \geq 0, \tag{46}$$

where

$$A^{\mathrm{ZOH}}(k) = \exp(A\Delta_s), \tag{47a}$$

$$B^{\mathrm{ZOH}}(k) = \left(\int_0^{\Delta_s} \exp(A\tau)d\tau\right) B, \tag{47b}$$

$$H^{\mathrm{ZOH}}(k) = H. \tag{47c}$$

Equations (47) can also be written as follows, which is easier to solve numerically [16]:

$$\begin{bmatrix} A^{\mathrm{ZOH}} & B^{\mathrm{ZOH}} \\ 0 & I \end{bmatrix} = \exp\left(\begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix} \Delta_s\right). \tag{48}$$

Note that the matrices are not written down specifically due to their size.

4.3 Constraints

The spatial states and control inputs are limited by constant bounds that act as constraints.

Namely, considering a racing environment, only forward motion of the vehicle is assumed. This is defined by spatially dependent bounds in the curvilinear system as follows:

$$
\begin{aligned}
&\theta_{min} \leq \theta \leq \theta_{max}, \\
&v_{min} \leq v \leq v_{max},
\end{aligned}
\tag{49}
$$

where $\theta_{min} = \left(-\frac{\pi}{4}, 0\right]$, $\theta_{max} = \left[0, \frac{\pi}{4}\right)$ and $v_{min} \geq 0$.

Furthermore, the other states and control inputs are limited by constant bounds:

$$
\begin{aligned}
a_{min} &\leq a \leq a_{max}, \\
\kappa_{min} &\leq \kappa \leq \kappa_{max}, \\
c_{G,min} &\leq C \leq c_{G,max}.
\end{aligned}
\tag{50}
$$

Finally, the lane boundaries spatially determine state bounds:

$$d_{min} \leq d \leq d_{max}. \tag{51}$$

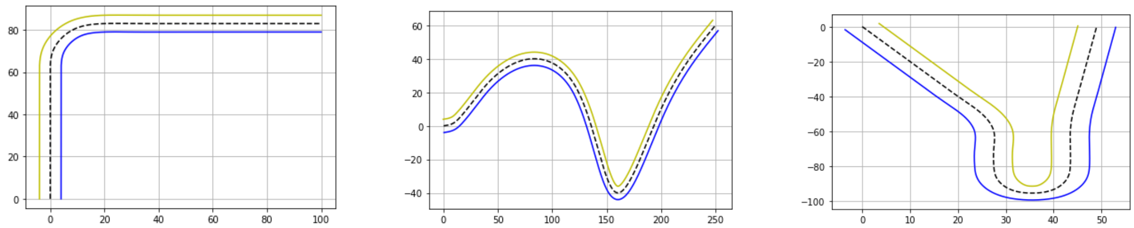
4.4 Final Controller

Now, the optimal control problem can be put together. At each discretization step length Δ_s, the spatial-based MPC controller must solve the following convex optimization problem:

$$\begin{aligned}
\underset{a, c_G}{\text{minimize}} \quad & J(d, \theta, v) \\
\text{subject to} \quad & \zeta_{k+1} = A_k \zeta_k + B_k u_k + H_k, \\
& \zeta_0 = \zeta(s), \\
& \theta_{min} \leq \theta \leq \theta_{max}, \\
& v_{min} \leq v \leq v_{max}, \\
& a_{min} \leq a \leq a_{max}, \\
& \kappa_{min} \leq \kappa \leq \kappa_{max}, \\
& c_{G,min} \leq c_G \leq c_{G,max}, \\
& k = 0, ..., N-1.
\end{aligned} \tag{52}$$

5 Simulation Setup

To test the proposed MPC controller and analyse its performance, a simulation environment is programmed in Python. Three racetracks are manually created and imported into the environment, using a cubic-spline planner where the spline is representing the center-line of the track (from GitHub, [17]). See Figure 13 for the visualization of the three tracks, which will be referenced to by the S-curve, the 90-degrees curve and the hairpin curve, from left to right, from now on. The left and right boundary of the road are shown with yellow and blue line, respectively. The center-line is indicated by the dotted line.



(A) The 90-degrees curve (B) The S-curve (C) The hairpin curve

FIGURE 13: Representation of the track segments, with the global X- and Y- coordinate on the horizontal and vertical axis, respectively

The optimization problem (Eq. (52)) is modeled as a convex optimisation problem in *CVXPY*, a Python-embedded modeling language [18]. First, the problem is initialized and subsequently, the convex optimization problem is solved with the solver *ECOS* at each sampling point k, over the prediction horizon N with step size Δ_s. ECOS is an efficient embedded conic solver, which performs well for small and medium-sized optimization problems such as (Eq. (52)) [19]. The following pseudo-code shows the structure of the algorithm :

Algorithm 1 Pseudo-code spatial-based MPC

```
Warm start ζ_0 = ζ(s) to compute optimal input ū
while prediction horizon has not crossed finish line do
    for k = 0, ..., N − 1 do
        Linearize around (ζ̄, ū) (Eq. 44) and discretize (Eq. 46)
        Solve MPC problem with ECOS in CVXPY (Eq. 52)
        if Solution is optimal then
            ζ̄ = ζ, ū = u
        else
            MPC unsolvable
    Recover Cartesian coordinates
```

All figures are generated with the Python module *Matplotlib*.

6 Results

This chapter is devoted to the results from the simulations and their interpretations.

Unless stated otherwise, the following initial values of the state parameters were used in the simulation:

Parameter	Unit	Value
d	m	0
θ	rad	0
v	m/s	40
κ_G	rad/m	0

Table 3: Initial values state parameters

Similarly, the following constraints were used:

Constraint	Unit	Value
θ_{min}	rad	$-\frac{\pi}{4}$
θ_{max}	rad	$\frac{\pi}{4}$
v_{min}	m/s	0
v_{max}	m/s	$\frac{150}{3.6}$
a_{min}	m/s^2	-5
a_{max}	m/s^2	5
$\kappa_{G,min}$	rad/m	$-\frac{\tan(\delta_{\max})}{L}$
$\kappa_{G,max}$	rad/m	$\frac{\tan(\delta_{\max})}{L}$
$c_{G,min}$	rad/ms	-0.2
$c_{G,max}$	rad/ms	0.2

Table 4: Values constraints

where the maximum steering angle $\delta_{max} = \frac{\pi}{4}$ and the length of the vehicle $L = 3$ m. Lastly, $N = 15$, $\Delta_s = 4$ and $\mu = 0.8$ (unless stated otherwise).

6.1 Main Results

The main question of this thesis concerns the fastest path and the corresponding speed of an autonomous vehicle on a racing track. Therefore, these quantities are plotted for each of the simulation scenarios and can be found in Appendix B.1 (see Figure 17, 18, 19).

Note that the road and the fastest path are plotted on the global X-axis, while the speed is plotted against the curvilinear abscissa s.

The optimal path, denoted by the pink line in the upper picture of the figures, seems to be realistic. Namely, the vehicle takes the inner corner and does not switch sides on the lane more often than necessary. One can also notice that the pink line stops before reaching the end of the road. After all, the algorithm is written such that it stops once it cannot predict into the future for the full prediction horizon.

The corresponding speed of the vehicle belonging to these paths seem somewhat optimal. In all three figures, it can be seen that the speed decreases when driving towards a corner and increases when leaving the corner. This is realistic, since it might otherwise not be possible to take the corner. However, consider Figure 17, where the vehicle lowers it speed after $s = 100$, even though there is no corner coming up. This behaviour could either be the result of a mistake in the controller, or a bug in the simulation program.

6.2 Validation Cost Function

In order to validate whether the proposed model functions as formulated, each term in the cost function is taken into consideration, separately. The cost function (see Eq. (40)) is split in three parts, namely:

$$J_1(y, \theta, v) = \sum_{k=0}^{N-1} \gamma_{1,k} d_k \tag{53}$$

$$J_2(y, \theta, v) = \sum_{k=0}^{N-1} \gamma_{2,k} v_k \tag{54}$$

$$J_3(y, \theta, v) = \sum_{k=0}^{N-1} \gamma_{3,k} \theta_k^2. \tag{55}$$

From studying the results of the controller for each of the cost functions J_1, J_2 and J_3, the contribution from each of the terms can be examined and cost function J can be validated. However, unfortunately, cost function J_1 and J_2 lead to non-convex optimisation problems. So, only cost function J_3 can be evaluated, see Figure 14. The optimal path follows the center-line of the road closely, since minimizing J_3 results in minimizing θ_k^2. So, in practice the heading angle with respect to the center-line is minimized and the vehicle follows it closely.

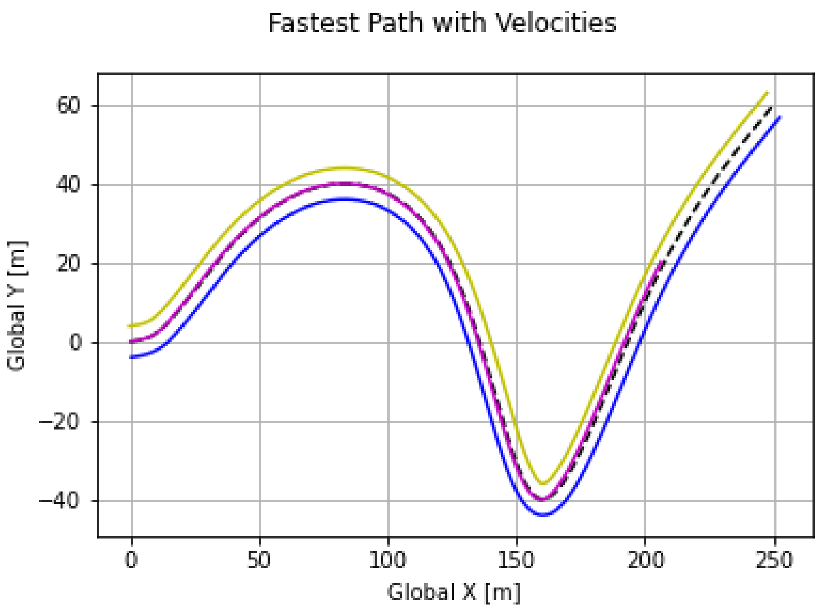


FIGURE 14: Optimal path on the road (pink line) for cost function J_3

To investigate the effect of the terms in the cost function J, consider Figure 15. Here, the value for γ_1 and γ_2 are presented at every value for s on the S-Curve (when driven as in Figure 18).

First, consider the values of γ_1. Whenever the road turns to the right, γ_1 is negative. Therefore, it is beneficial to have a positive value of d_k (considering the goal of minimiza-

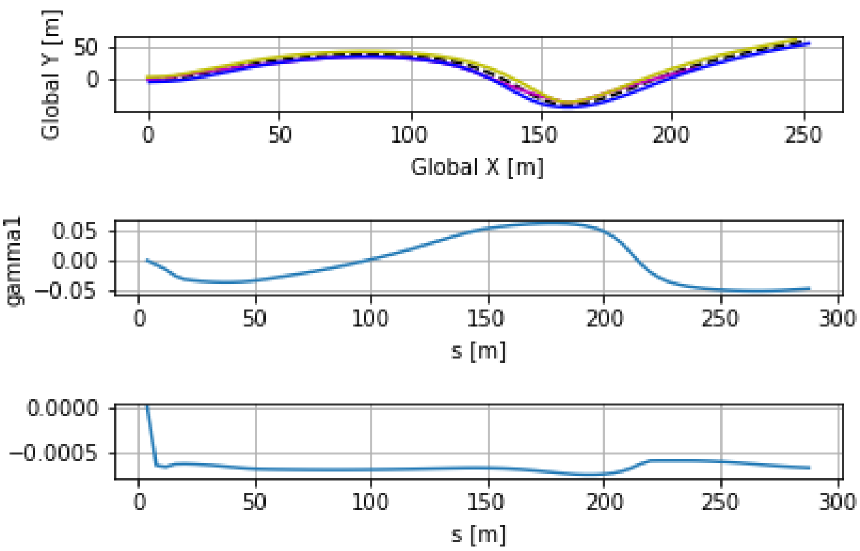


FIGURE 15: Values of γ_1, γ_2 and γ_3 on the S-curve (from top to bottom)

tion) and so to be on the right side of the road. Similarly, when the road turns to the left, γ_1 is positive and it is beneficial to have a negative value of d_k (and so to be on the left side of the road). This is in line with the rationale as explained in Section 4.1: the term γ_1 will maximize the lateral deviation along with the curvature of the track. Secondly, consider γ_2. Its value is always negative and significantly smaller than γ_2 for the entire track. This implies that the addition of γ_2 in the cost function has a small effect and could explain why the speed decreases after $s = 100$ in Figure 17, since there is not much regulation on the vehicle's speed.

So, even though the vehicle's speed might not be optimal by the controller, it does behave as expected.

6.3 Robustness MPC

This section is meant to analyze the robustness of the MPC controller by varying the parameters in the controller. This gives insight in the effect of the successive linearization and the receding horizon strategy and enables us to make general conclusions on the MPC controller.

6.3.1 Prediction Horizon and Step-Size

The prediction horizon N and step size Δ_s are important parameters for the MPC controller, since they determine how far in the future predictions are done.

Consider Table 5 which shows the average time per iteration (t_{solve}) and the total number of iterations (# Iterations) needed for different values for the planning horizon N and the step size Δ_s for the S-curve. The total number of iterations needed is the number of times at which the optimization problem is solved. The average time per iteration denotes the average time needed to solve the optimization problem once, that is to find the optimal input parameters to minimize the cost function J. Note that the simulation also needs time to linearize the vehicle model, which is not taken into account in t_{solve}.

N	**5**			**10**		
$\mathbf{\Delta_s}$ $[m]$	**2**	**4**	**6**	**2**	**4**	**6**
# **Iterations**	-	81	-	-	76	47
$\mathbf{t}_{\text{solve}}$ [sec]	-	0.0049	-	-	0.0076	0.0074
N	**15**			**20**		
$\mathbf{\Delta_s}$ $[m]$	**2**	**4**	**6**	**2**	**4**	**6**
# **Iterations**	-	71	42	-	-	37
$\mathbf{t}_{\text{solve}}$ [sec]	-	0.0095	0.0099	-	-	0.0136

Table 5: Computational effects of varying planning horizon and step length

Not all values for the planning horizon and step size are valid for the controller, these are denoted by a dash (-) in Table 5. This happens for different values of N and for Δ_s. One reason is the limited memory of the computer which performed the simulation, as it reached the maximum number of iterations. However, this is not the case for all of the scenarios. In all of the test scenarios, the optimization problem is the same, except for the linearizations of the vehicle model (because of the different step size). Therefore, it can be concluded that the linearization of the vehicle is not stable. It should actually perform for different values of the planning horizon and step size.

It can also be seen from Table 5 that higher horizon lengths N lead to a more time-consuming optimization problem. A longer horizon length and greater step size, both enable to plan further along on the track. As example, Figure 24 in Appendix B.3 shows the effect of the step size. The figure shows the path driven at the same iteration number at a constant horizon length $N = 5$ and varying step size $\Delta_s = 2, 4, 6$. The red dots indicate the predictions in the horizon, at which the vehicle dynamics are linearized in the next iterations. It can be seen that the algorithm is able to plan further ahead for greater step sizes. Also note that in Figure 24a the vehicle turns later to the inner bound of the track than in Figure 24c, as it recognizes the turn later. After all, in Figure 24a the vehicle is able to see 20 meters in the future, whereas the vehicle can see 30 meters ahead in Figure 24c.

Similar results are depicted in Figure 25, where the step size is kept constant, but the horizon length is increased. The vehicle has not traveled further along the track, but it does plan further ahead and recognizes earlier when a turn is coming up. Consequently, it drives to the inner bound of the track earlier.

Even though Figure 24 and 25 show successful simulations with results as expected, Table 5 shows that the MPC controller is not robust with respect to a change in planning horizon N and step length Δ_s.

6.3.2 Initial Conditions

This section is meant to show the effect of different values for the initial conditions of the problem, which concern the initial position, heading, velocity and car curvature of the simulated vehicle. After all, varying these initial conditions show the performance of the controller well. The figures which are referenced in this section can all be found in Appendix B.4.

First, consider Figure 26, which shows the optimal path for a vehicle starting at the left side of the road of the 90 deg curve ($d = 4$). The figure shows the vehicle steers towards the curve as soon as it recognises it, which is similar to previous results.

Secondly, consider Figure 27, which shows the optimal path and the corresponding speed for a vehicle starting with a speed of 20 m/s on the hairpin curve. It can be seen that the vehicle still takes the same path, although its speed profile looks differently, since it does not need not to slow down to make the corner.

Thirdly, consider Figure 28, which shows the optimal path for a vehicle starting at $\theta = \frac{\pi}{4}$ rad on the 90 deg curve. Hence, it enters the figure already turned $\frac{\pi}{4}$ rad. It does take the corner as soon as it recognises it, although it approaches it with a lot of little turns.

Finally, consider Figure 29 which shows the optimal path for a vehicle on the 90 deg curve with a vehicle curvature of 0.1 rad/m. Hence, it does start on the center-line, although its wheels are already turned 0.1 rad/m. Therefore, it has a small deviation to the left side of the right, but steers back to the center-line and eventually the corner, as expected.

So, the controller performs reasonably well for different initial conditions. In all test cases, the vehicle still steers to the inside of the corners. However, when initializing the vehicle at an angle with respect to the center-line, the vehicle makes a lot of little of turns. This is far from the optimal solution in the real world. Hence, it should be penalized as well when the path is not smooth, i.e. has a lot of little turns. Such a penalty is not present in the controller.

6.3.3 Constraints

This section is meant to show the effect of different values in the constraints of the problem, which concern the maximal acceleration/deceleration and maximal curvature rate. The figures which are referenced in this section can all be found in Appendix B.5.

Consider Figure 30, which shows the optimal path and corresponding speed for a vehicle on the hairpin curve, where the maximal acceleration/deceleration is set to 1 m/s^2. Here, the lowest speed of approximately 30 m/s is reached at $s = 100$. Compare this to Figure 19, where the same is depicted, but for a vehicle with a maximal acceleration/deceleration of 5 m/s^2. There, the lowest speed is also reached at $s = 100$ m and its value is approximately 28 m/s. Both figures also show an approximately linear relationship of the speed over place and a similar optimal path. Hence, a different value for the acceleration/deceleration of 1 m/s^2 does not seem to change the performance of the controller.

Consider Figure 31, which shows the optimal path for a vehicle on the hairpin curve, where the maximal vehicle steering is set to 0.5 rad/ms. The vehicle steers to the inside of the corner, once it sees the curve is coming up. This is similar to the behaviour of the controller when the maximal vehicle steering is set to 0.2 rad/ms (see Figure 19).

So, the controller performs well for different values in the constraints of the problem.

6.4 Friction Coefficient

This section is meant to show the effect of changing the friction coefficient μ on the generated trajectory. Note that the initial value of v was set to 20 m/s and $a_{min} = -2$ m/s^2 and $a_{max} = 2$ m/s^2.

Figure 16 shows the effect of enlarging the value for μ for the spatial state v on the S-curve. The result of the lateral dynamics are given in Appendix B.2. From Figure 16 is visible that setting μ to zero gives an approximately linear increase in velocity, until it hits the maximum value v_{max}. This is as expected, since there is no longitudinal resistance when $\mu = 0$ and therefore the vehicle can freely accelerate. As the value of μ increases, the acceleration of the vehicle decreases. This is as expected, since there is more longitudinal resistance for higher values of μ.

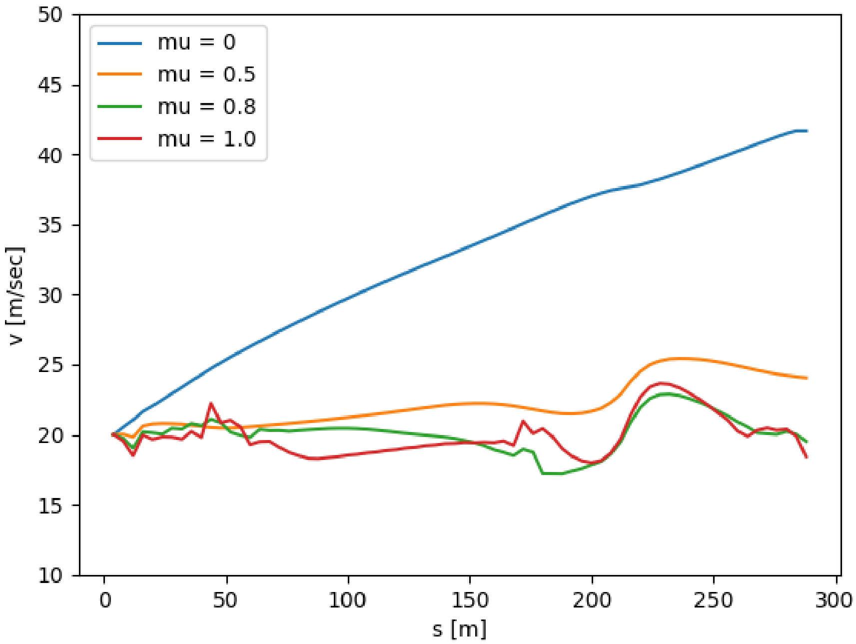


FIGURE 16: Velocity profile S-curve different μ-values

7 Discussion

In general, the results of the MPC controller are in line with the expectations. However, there are a few conflicts between the results and the expectations, namely:

1. Even though there is no corner coming up, the vehicle's speed decreases in some situations. This is not as expected, since this is not the fastest way to travel a straight track. However, the objective of the controller is to maximise the progression along the center-line of the road and not to have the highest possible velocity. Due to the approximation of the cost function, there is not much regulation on the vehicle's speed. This could be the explanation for this conflict.

2. The MPC controller is not robust with respect to a change in planning horizon N and step size Δ_s. This conflict could be due to the successive linearizing and discretizing of the vehicle dynamics. Namely, for different step sizes, the vehicle model is linearized and discretized at different locations on the road and therefore also different for each step size.

3. The proposed path is not smooth in all situations, in the sense that it contains a lot of little turns. This could never be the optimal way to cover a track. However, this behaviour is expected from the proposed MPC controller, since there is no criterium on the smoothness of the proposed path. The controller does not aim to keep the curvature at the next step similar to the current one.

Furthermore, it should be noted that the results are reflected upon as approximations of the optimal path and speed. The following simplifications have been made:

1. The kinematic bicycle model does not include slip, which cannot be justified in racing environments. See Appendix C for an elaboration on slip.

2. Both the cost function and the vehicle dynamics are subsequently approximated in the controller to enable convexity and real-time feasibility.

3. The vehicle is modelled by a bicycle model. Therefore, the width of the vehicle is neglected. This can be overcome by adjusting the bounds of the road or adjusting the dimensions of the vehicle in the simulation.

8 Conclusion

In this thesis, a MPC algorithm is proposed for path and trajectory planning in autonomous racing. The problem is formulated as a convex optimization problem by linearizing the vehicle dynamics successively around previously predicted optimal solutions. The aim of the controller is to maximise the progress along the center-line of a predefined track, which is approximated by a second-order Taylor series. The vehicle is modeled kinematically and the time dynamics are reformulated into spatial dynamics, which enables a natural formulation of the track.

The MPC algorithm can be solved with efficient computation times and it is able to find the fastest path and the corresponding speed of an autonomous vehicle reasonably well, which answers the main question of this thesis.

9 Recommendations

If more time was provided, it would be interesting to perfect the current MPC controller. I would change the cost function, to check what would give the most satisfactory results. Currently, the cost function is a second-order Taylor approximation of the progress maximization along the center-line of the track. I would add another value to this cost function, which regulates the vehicle's speed better. The ideal value for this value could be obtained by testing. Furthermore, I suspect the robustness of the MPC controller could also be improved. I would research why certain combinations for prediction horizon N and step size Δ_s fail. I suspect this could also be the effect of a wrong, or inefficient implementation of the algorithm in Python code. But it could also be the result of the successive linearizing and discretizing of the vehicle dynamics, for which it is harder to come up with an alternative. In theory, the linearizing and discretizing should guarantee convexity, but it would be good to verify this by testing with the code. Lastly, I would add another criterium to the cost function in the objective function that takes smoothness into account, to make the proposed paths smoother.

For future research, it would also be interesting to determine the fastest path and corresponding velocities of autonomous vehicles on a given track with other methods than model predictive control. I would suggest determining the racing lines with reinforcement learning in artificial intelligence, where an intelligent agent is developed that perceives its environment and takes actions that minimise the total time needed to cover the track. The agent would start at point i and decide the points $i + 1, i + 2, ...$ until the finish line using some intelligent algorithm with probability distribution. This would be run many times and the best racing line is chosen. Considering all possible racing lines would be infeasible, therefore a form of branch-trimming should be chosen such that options can be discarded without calculating them. This could be done, for example, by disregarding all points $i+1$ and $i + 2$, where the curve from i to $i + 2$ is not smooth enough. This approach saves the formulation of an exact mathematical model for the problem and could obtain better results.

A List of Variables

The variables used in the vehicle model are presented with their description and range in Table 6. It must be noted that the ranges of the variables do not take any specific characteristics of the system into account, only the physical boundaries.

List of Variables			
Variable	Unit	Range	Description
s	m	[0, ∞]	Curvilinear abscissa
G	-	-	Center of mass of the go-kart
F	-	-	Front wheel of the go-kart
R	-	-	Rear wheel of the go-kart
ζ	-	-	State vector (d, θ, v, κ_G)
u	-	-	Input vector (a, c_G)
Γ	-	-	The center-line of the road
P	-	-	The point on Γ closest to R, assumed to be unique
L	m	-	The length of the go-kart from the center of the front wheel (F) to the rear wheel (R)
θ_G	rad	[-π, π]	Angle between the vehicle and the global X-axis
θ_Γ	rad	[-π,π]	Angle between the path Γ and the global X-axis
θ	rad	[-π,π]	Angle between the vehicle and the path Γ, equivalent to $\theta_G - \theta_\Gamma$
v	m/s	[0, ∞)	Vehicle velocity
d	m	(-∞, ∞)	Directed distance between R and P (positive means R is to the left of P, negative means to the right)
μ	-	-	The longitudinal tire-road friction coefficient
a	m/s^2	[0, ∞)	The longitudinal acceleration
c_G	rad/ms	[0, ∞)	The vehicle steering
ρ	m	(-∞, ∞)	Radius of the circle motion of the path at P, sign indicates orientation (positive means the midpoint is to the left of P, negative means to the right)
v_Γ	m/s	[0, ∞)	Vehicle velocity projected in direction of Γ
κ_Γ	rad/m	[0, ∞)	Curvature of the path Γ at M
κ_G	rad/m	[0, ∞)	Curvature of the go-kart
δ	rad	[-π,π]	The steering angle of the vehicle with respect to its current direction
α	rad	[-π,π]	The slip angle
F_c	N	[0, ∞)	The centripetal force on the go-kart
F_w	N	[0, ∞)	The force from tire-road friction on the go-kart

Table 6: List of Variables

B Figures

B.1 Main Results

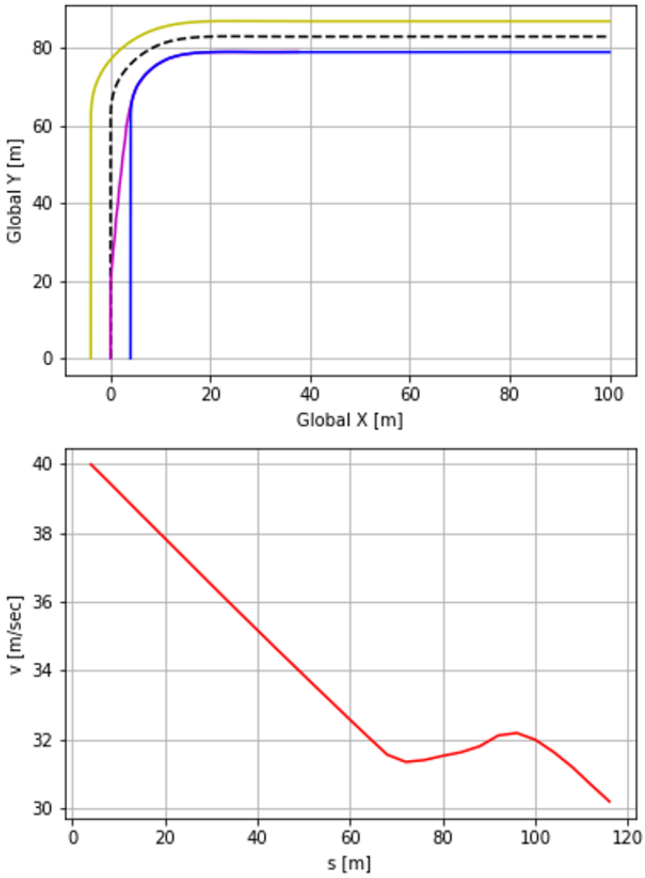


FIGURE 17: The 90-degrees curve (above: optimal path on the road (pink line), below: corresponding speeds)

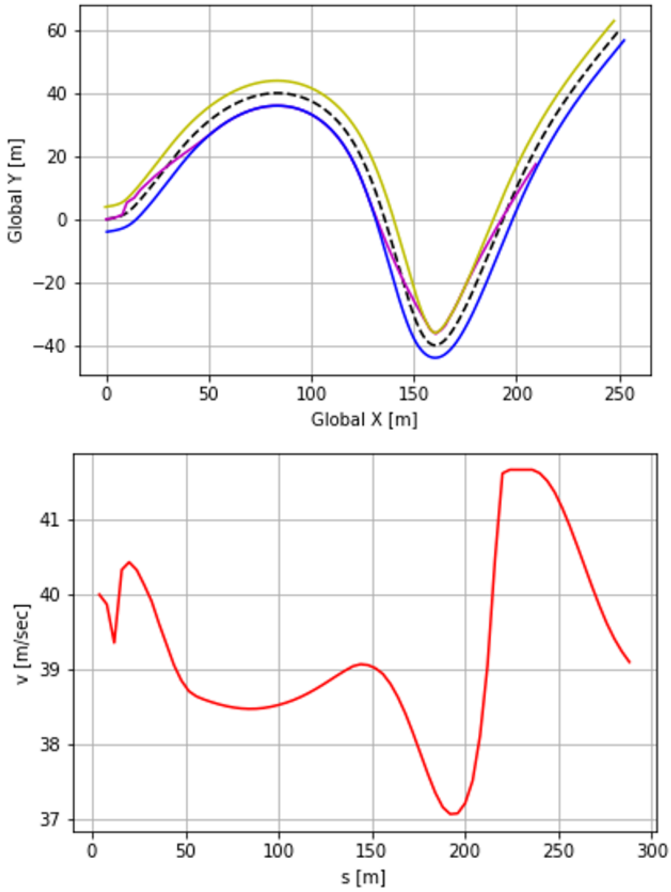
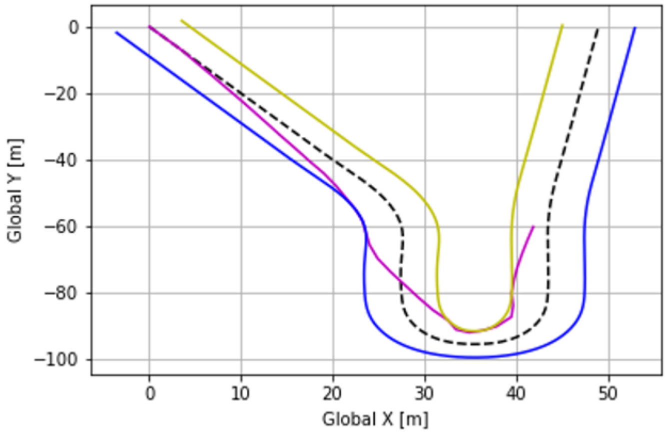


FIGURE 18: The S-curve (above: optimal path on the road (pink line), below: corresponding speeds)



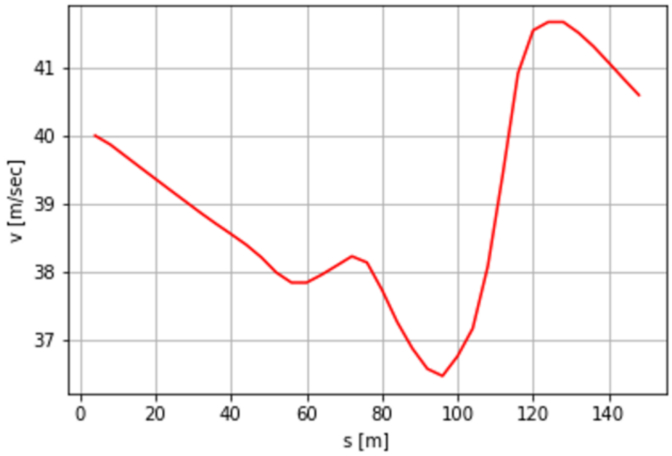


FIGURE 19: The hairspin curve (above: optimal path on the road (pink line), below: corresponding speeds)

B.2 Optimal Path for different Longitudinal Coefficients

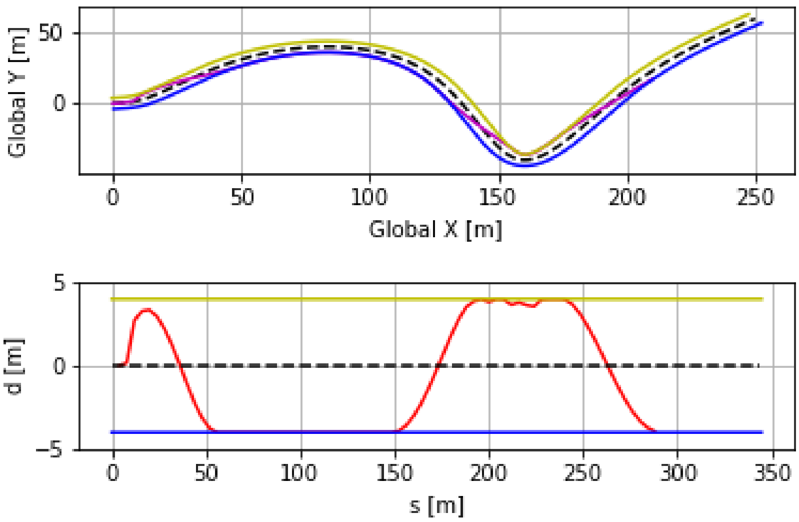


FIGURE 20: Optimal path for $\mu = 0$

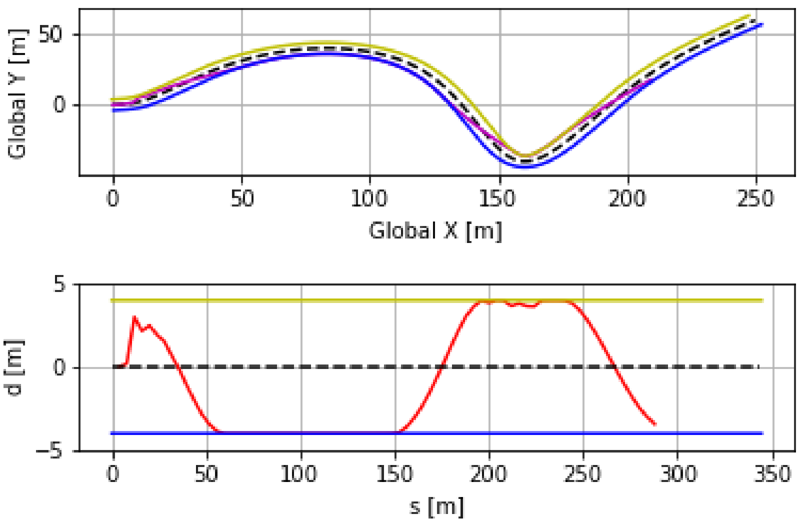


FIGURE 21: Optimal path for $\mu = 0.5$

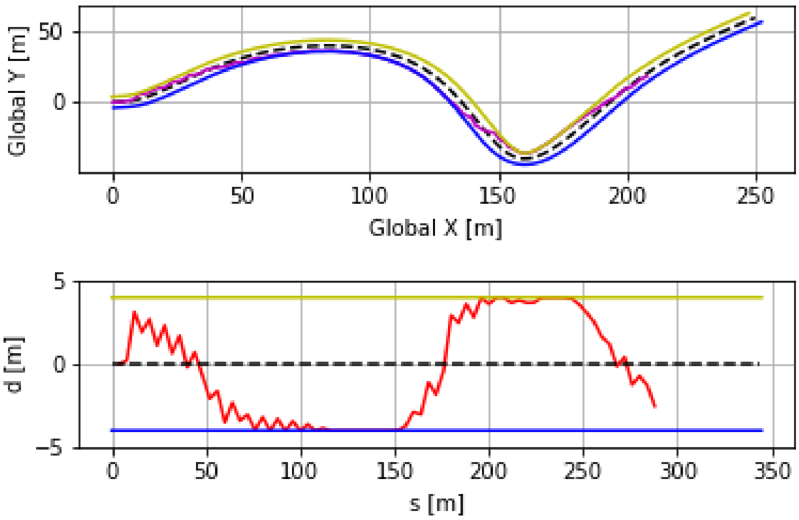


FIGURE 22: Optimal path for $\mu = 0.8$

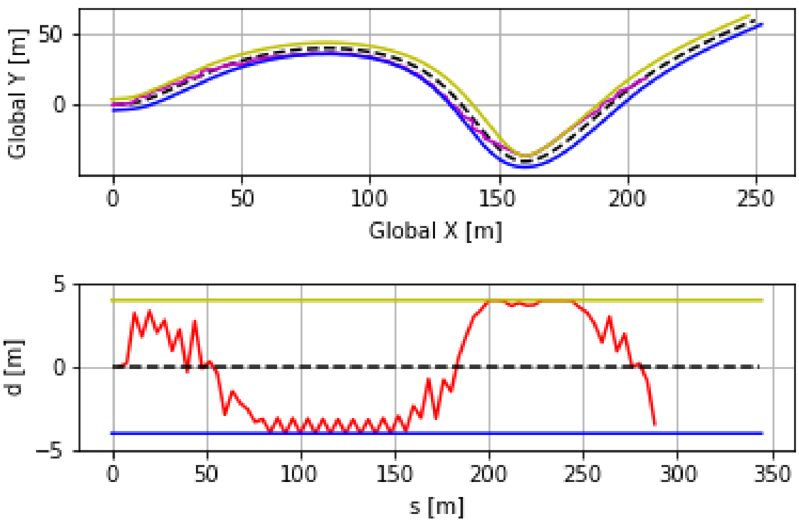
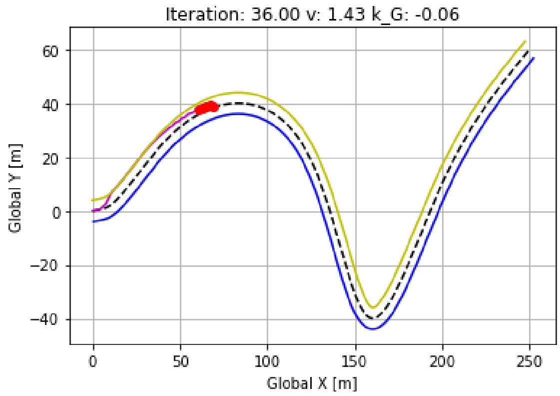
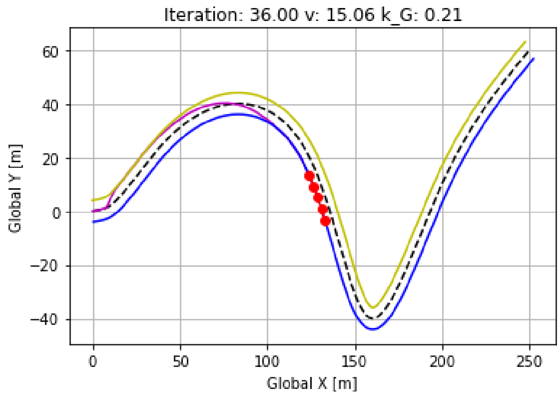


FIGURE 23: Optimal path for $\mu = 1.0$

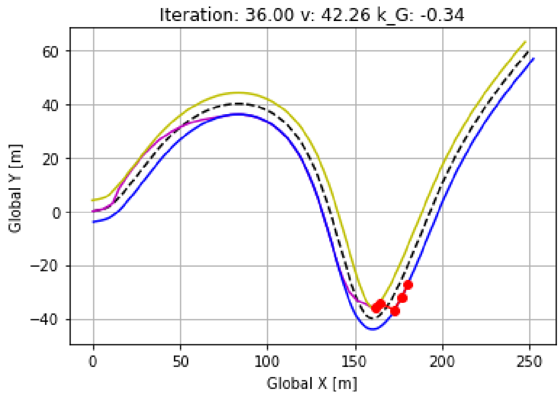
B.3 Robustness Analysis



(A) $N = 5, \Delta_s = 2$

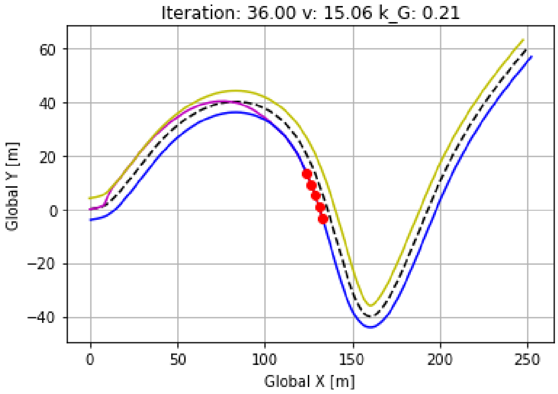


(B) $N = 5, \Delta_s = 4$

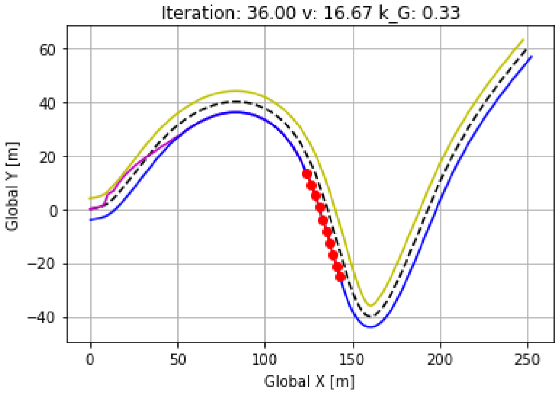


(C) $N = 5, \Delta_s = 6$

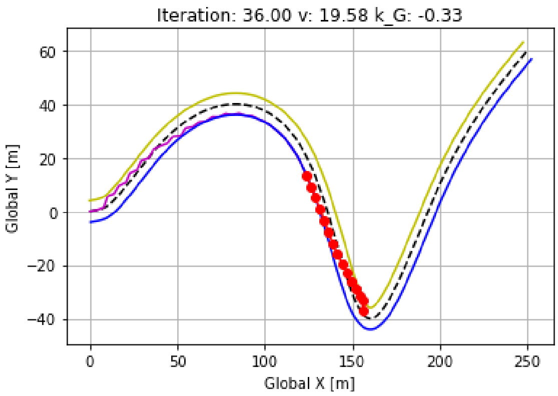
FIGURE 24: Optimal path for varying $N = 5$ and constant Δ_s



(A) $N = 5, \Delta_s = 4$



(B) $N = 105, \Delta_s = 4$



(C) $N = 15, \Delta_s = 4$

FIGURE 25: Optimal path for varying N and constant Δ_s

B.4 Initial Conditions

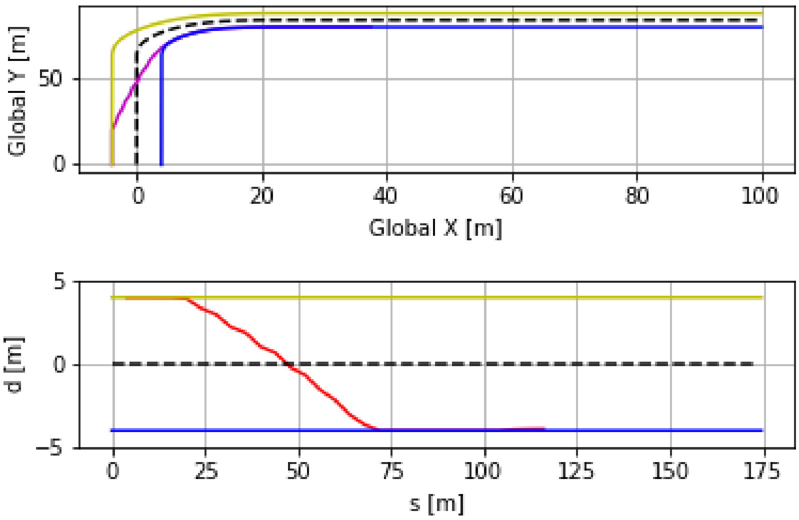


FIGURE 26: The optimal path for the 90 deg curve, starting at $d = 4\ m$

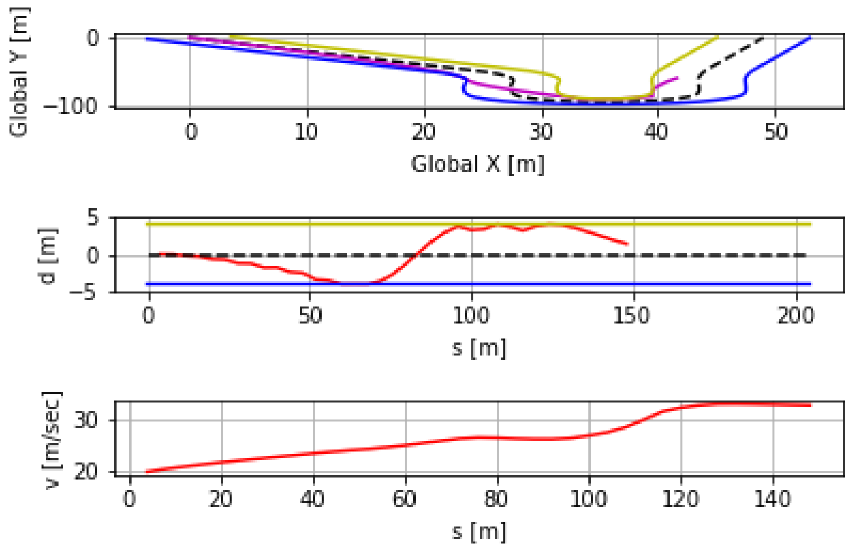


FIGURE 27: The optimal path and corresponding speed for the hairpin curve, starting at $v = 20\ m/s$

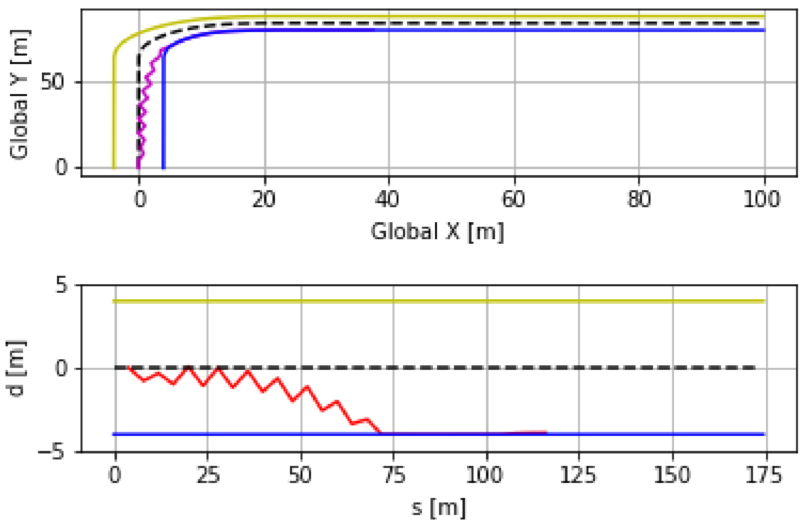


FIGURE 28: The optimal path for the 90 deg curve, starting at $\theta = \frac{\pi}{4}$ rad

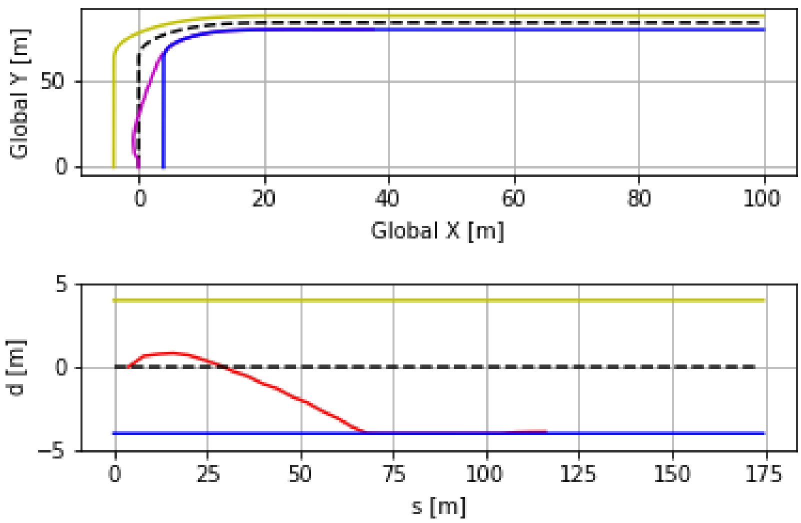


FIGURE 29: The optimal path for the 90 deg curve, starting at $c_G = 0.1$ rad$/m$

B.5 Constraints

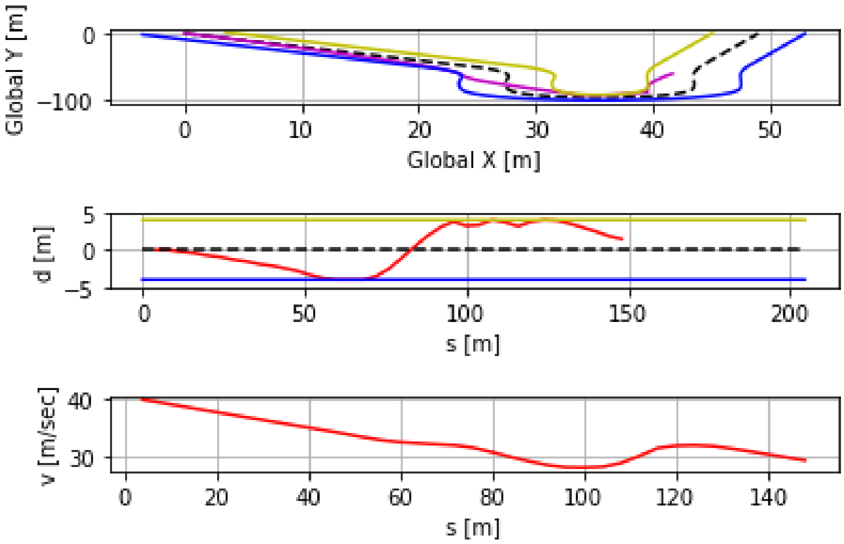


FIGURE 30: The optimal path and corresponding speed for the hairpin curve, with $a_{\max} = 1\ m/s^2$

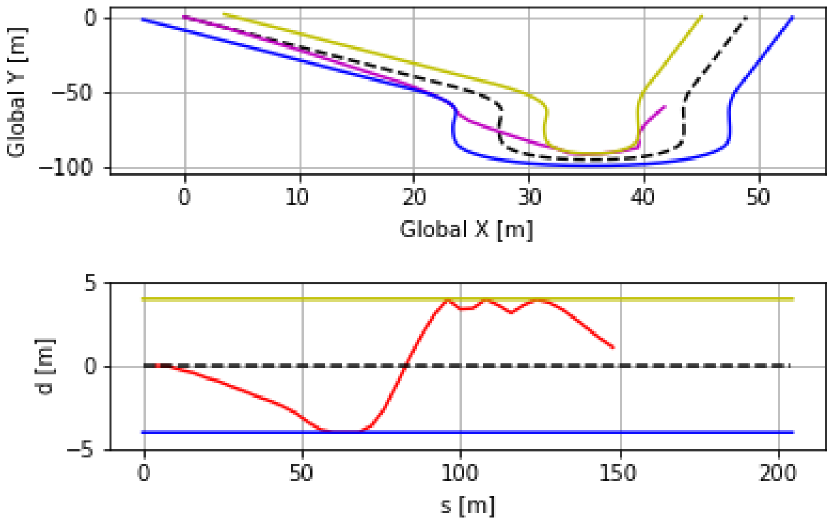


FIGURE 31: The optimal path and corresponding speed for the hairpin curve, with $c_{G,max} = 0.5$ rad$/ms$

C Elaboration on Slip

In this thesis, an attempt was done to account for slip in the vehicle model, to make it more complying to racing conditions. This Appendix presents the findings and work done with regards to this topic, which has not been used in the thesis. First, the relevant mechanics and vehicle dynamics are presented. Then, the side slip angle is introduced. Finally, a reflection upon this work is given and a recommendation for future research.

The objective of introducing the side slip angle is to formulate an extended kinematic bicycle model as described in [20]. This adds complexity to the regular bicycle model, but does make the extended kinematic bicycle model more suitable for vehicles operating on slippery surfaces and high speed range [21]. Then, a similar controller to the current one could be set up to solve the problem of path and trajectory planning.

C.1 Relevant Mechanics

According to the first law of Newton, a body will not change its motion unless a force acts upon it. More specifically, a driving vehicle must experience a force to change its driving direction. This force is realized by turning the wheels to experience *friction*. The friction is caused by interaction from the tires with the road and depends on the type of tires and the conditions of the road. The friction is perpendicular to the front wheels of the vehicle. One component of the friction is perpendicular to the motion of the vehicle and the other is longitudinal friction.

Consider Figure 32 where three vehicles are presented with each a forward velocity v. Their front wheels are turned to the right and therefore, they experience a friction force F_w perpendicular to the motion of the vehicle. This force changes the direction in which the vehicles travel and enables them to take the corner.

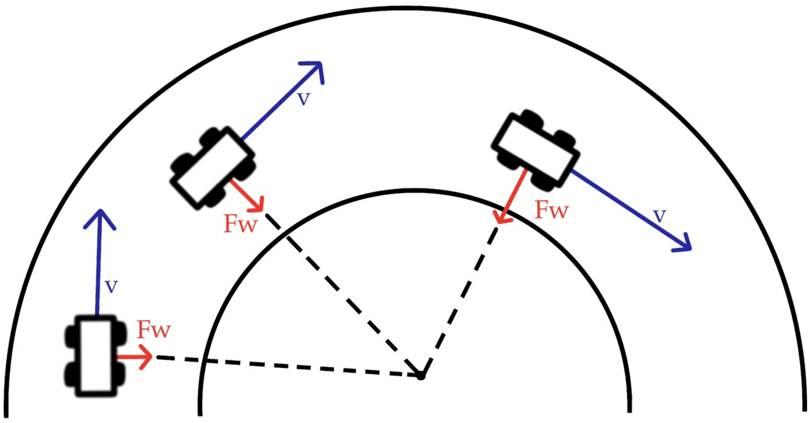


Figure 32: Model of 3 vehicles taking a corner

It actually follows from Figure 32 that the friction acts as a centripetal force: a force that makes a body follow a curved path. The direction of a centripetal force is always orthogonal to the motion of the body and towards the center of curvature of the path. The magnitude of the centripetal force of an object of mass m with tangential velocity v along a path with radius of curvature r is:

$$F_c = \frac{mv^2}{r}. \tag{56}$$

The centripetal force is illustrated in Figure 33. Here a vehicle with center of mass G has a forward velocity v. Its front wheel is turned, experiencing a friction perpendicular to the velocity that acts as a centripetal force and is directed at point M. This point acts as center of the circle that the centripetal force introduces. This circle is followed if G keeps experiencing the centripetal force in the direction of M.

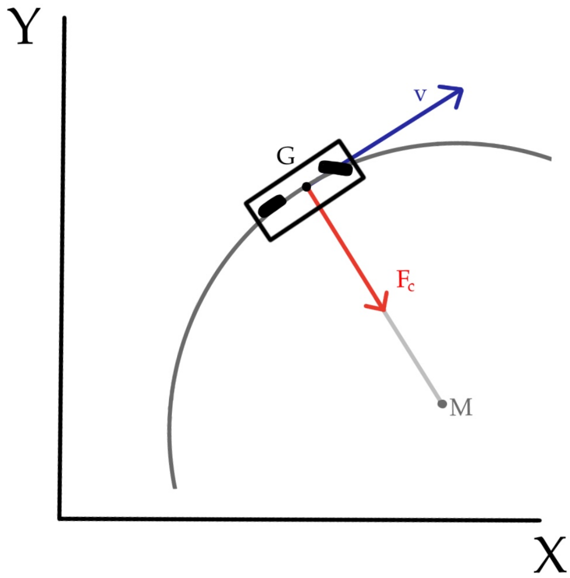


FIGURE 33: Sketch of vehicle and the path with the centripetal force

C.2 Relevant Vehicle Dynamics

The force that enables a vehicle on rubber wheels to turn is produced by the elasticity of the tires. This force is discussed in this section with the development of slip angle.

First, consider Figure 34 that illustrates a tire from the side. It is rotating clockwise and so moving to the right of the figure. The part of the wheel that touches the ground is called the *contact patch* and it starts at point B and ends at E. We will follow an element of rubber starting at point A, where it does not yet touch the ground. Once the tire advances, the element will reach B and enter the contact patch.

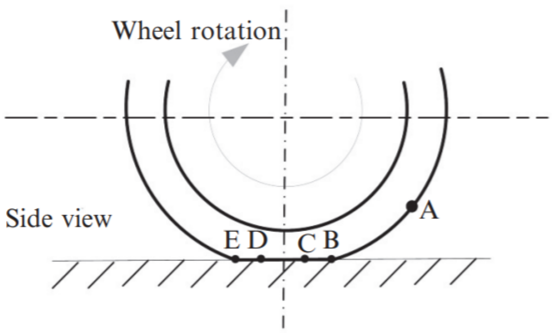


FIGURE 34: Side view tire [22]

The contact patch experiences vertical load from the vehicle weight and balances this with the elastic deformation of the tire (the contact patch is flat) and the inflation pressure of the gas inside [22]. As the rubber element moves to C, the vertical load increases until around the midpoint. Then, the vertical load will decrease until it reaches zero at point E where the element leaves the contact patch. In this process, heat will be generated and energy lost leading to rolling resistance.

This process can also be seen in Figure 35 that illustrates the tire from above.

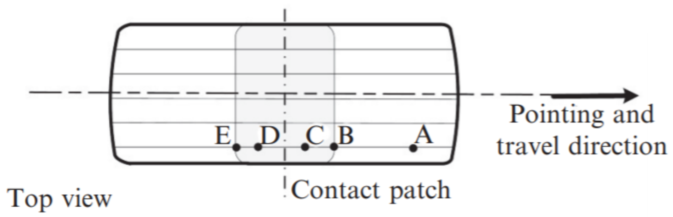


FIGURE 35: Top view tire [22]

In Figure 36, the tire is also viewed from above, but the tire has been steered. It is important to notice the tire travels to the right, even though it is not pointing in this direction. Namely, once the tire steers, the rubber that is in contact with the ground (the contact patch) cannot easily move due to the vertical load of the vehicle's weight.

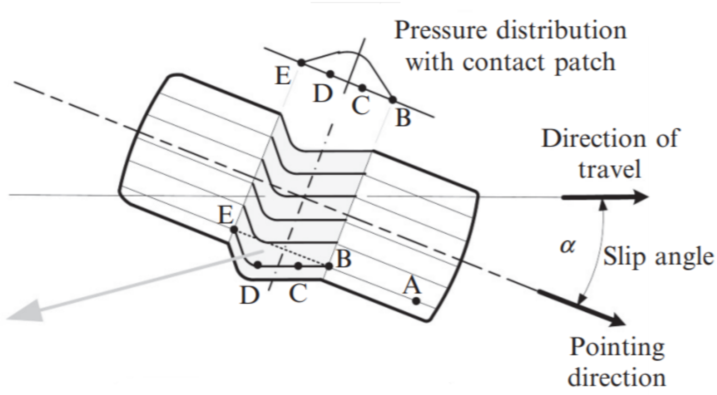


FIGURE 36: Top view steered tire [22]

Reconsider the element of rubber as before, starting at point A. As it reaches point B and enters the contact patch, it makes contact with the road and bonds to it. Therefore, it is constrained to stay at rest with respect to that particular point on the road. As the tire moves to the right, it must move to C. In other words, it is displaced laterally and creates an elastic force laterally. As the element moves rearwards, both the displacement and the vertical load from the vehicle's weight increase, balancing each other. However, around position D the vertical load on the element starts to decrease and the element is urged to move back in its rest position. By the time the element exits the contact patch at E, the process is over and the element is back in rest position.

The angle between the direction of travel and the direction in which the wheels are pointing is the *slip angle*, as indicated in Figure 36. The arrow in the Figure denotes the lateral force on the contact patch. This force enables the vehicle to turn and change travel direction eventually.

C.3 Derivation Side Slip Angle

This section focuses on the attempt that was made to derive the side slip angle with the mechanics as presented in Section C.1.

Consider Figure 37 which contains a bicycle model viewed in a rotating frame of reference. The front and rear wheel experience friction, F_f and F_r respectively. The front wheel is turned δ degrees from the current orientation of the vehicle and the slip angle α is indicated. The center of gravity of the car (C_G) has a distance r to the center of the circular motion the vehicle is making. The distance from the front and rear wheel to the center of gravity are a and b, respectively. The friction experienced by the front wheel (F_f) can now be expressed by the slip angle (α) and the cornering stiffness (C_α). The latter depends on characteristics of the vehicle and the tires. The friction varies linearly with the slip angle [23]:

$$F_f = C_\alpha \alpha. \tag{57}$$

An attempt was made to express α by known parameters in the vehicle model by expressing F_f in Eq. (57) by known parameters. However, this attempt requires the assumption that $\alpha \approx \delta$ and δ is small, which cannot be supported in all scenarios. Nevertheless, the attempt is still presented below.

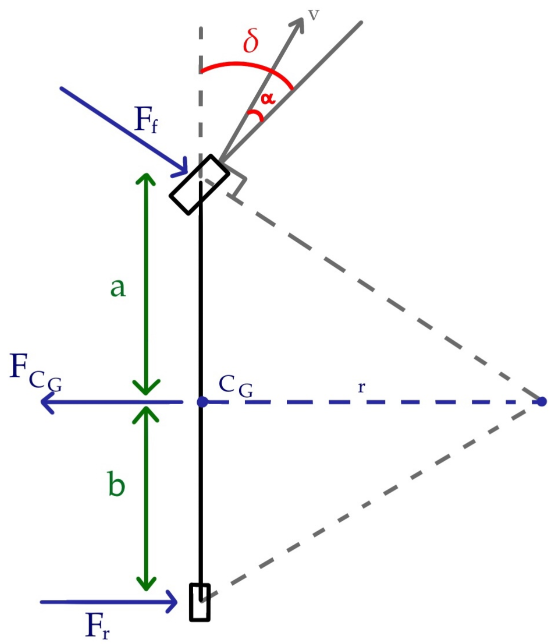


FIGURE 37: Bicycle model with rotating reference frame

In Figure 37 the center of gravity (C_G) experiences a centrifugal force (F_{C_G}) which is the reaction force of the centripetal force. It is a pseudo force appearing to act on the center of gravity that exists only in this reference frame. Its value is equal to the centripetal force:

$$F_{C_G} = \frac{mv^2}{r}. \tag{58}$$

Summing the forces in Figure 37 yields

$$F_{C_G} = F_r + F_f \cos(\delta - \alpha). \tag{59}$$

For of $\delta \approx \alpha$, $\cos(\delta - \alpha) \approx 1$, so combining Equation (58) and (59) gives:

$$F_r + F_f = \frac{mv^2}{r}. \tag{60}$$

Because the vehicle itself is in equilibrium in the turn, the sum of moments about the center of gravity must be equal to zero (assuming a small slip angle again):

$$\sum M_{C_G} = 0 = F_r b - F_f a \tag{61}$$

or

$$F_f = \frac{F_r b}{a}. \tag{62}$$

Substituting Equation (62) in (60) and noting that $a + b = L$ gives:

$$F_r\left(1 + \frac{b}{a}\right) = F_r\left(\frac{L}{a}\right) = \frac{mv^2}{r} \tag{63}$$

or

$$F_r = \left(\frac{ma}{L}\right)\frac{v^2}{r}. \tag{64}$$

Next, the vehicle is viewed in the lateral plane (side-view), as shown in Figure 38. The center of gravity experiences a downward force (F_g) due to gravity and the front and rear wheel experience upward forces from the ground (their weight, W_f and W_r respectively).

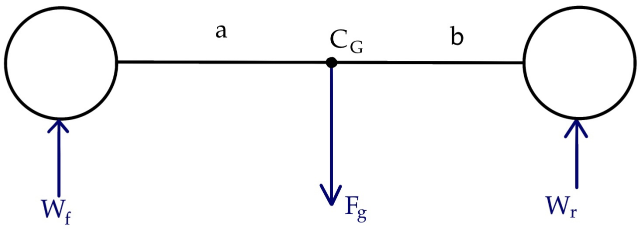


FIGURE 38: Bicycle model in lateral plane

These forces are due to gravity. The gravitational force that the center of gravity experiences can be expressed as:

$$F_g = mg, \tag{65}$$

where m denotes the mass of the vehicle and g denotes the gravitational acceleration ($g = 9.80665\ m/s^2$).

The sum of the moments about the front wheel (M_f) must equal zero:

$$\sum M_f = 0 = W_r(a + b) - F_g a = W_r L - F_g a. \tag{66}$$

Rewriting Equation (66) and using $F_g = mg$:

$$\frac{W_r}{g} = \frac{ma}{L}. \tag{67}$$

Substituting Equation (67) into (64) gives:

$$F_r = \frac{W_r}{g}\frac{v^2}{r}. \tag{68}$$

Finally, substituting Equation (68) into (57) gives:

$$\alpha = \frac{W_r v^2}{grC_\alpha} = \frac{W_r \kappa_\Gamma v^2}{gC_\alpha}. \tag{69}$$

Finally, an expression is derived for the side slip angles in known parameters.

C.4 Reflection and Recommendations

Mathematically deriving the side slip angle is difficult, as it requires a very elaborate dynamical model to precisely define it. For example, the model would require accurate tire models and their parameters such as the axle cornering stiffness. However, for derivation it is necessary to include the slip angle in a dynamic model, instead of integrating it into a kinematic model. The attempt of this thesis failed, because the side slip angle could not be placed within the context of the full dynamics and therefore not derived from any known forces.

An extended kinematic vehicle model could work properly, when the side slip angle is part of the state description and therefore also estimated by differential equations. An approach would be to use extended Kalman filtering as presented in [24]. This would be the recommendation for future research, as such a Kalman algorithm could be integrated with the MPC algorithm as presented in this thesis.

References

[1] EU. “Communication from the commission to the European Parliament, the Council, the European economic and social committee, the committee of the regions”. In: *On the road to automated mobility: An EU strategy for mobility of the future*. 2018. COM/2018/283 final.

[2] E. Moravčík and M. Jaśkiewicz. “Boosting car safety in the EU”. In: *2018 XI International Science-Technical Conference Automotive Safety*. 2018, pp. 1–5. DOI: 10.1109/AUTOSAFE.2018.8373307.

[3] Tesla. *Autopilot - Future of Driving*. URL: https://www.tesla.com/autopilot. (accessed on 13.11.2020).

[4] D. González et al. “A Review of Motion Planning Techniques for Automated Vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 17.4 (2016), pp. 1135–1145. DOI: 10.1109/TITS.2015.2498841.

[5] RDW (2018). *The Mission, Vision and Strategy of the RDW - Safe and Reliable on the Road.* URL: https://www.rdw.nl/-/media/rdw/rdw/pdf/sitecollectiondocuments/over-rdw/brochures-en-folders/rdw_strategy_en_def.pdf. (accessed on 13.11.2020).

[6] NVIDIA Bosch. *Bosch Announces AI Self-Driving Computer with NVIDIA*. URL: https://nvidianews.nvidia.com/news/bosch-announces-ai-self-driving-computer-with-nvidia. (accessed on 16-11-2020).

[7] C. Katrakazas et al. “Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions”. In: *Transportation Research Part C: Emerging Technologies* 60 (2015), pp. 416–442. ISSN: 0968-090X. DOI: 10.1016/j.trc.2015.09.011.

[8] M. Kissai et al. “A. Adaptive Robust Vehicle Motion Control for Future Over-Actuated Vehicles”. In: *Machines* 7.2 (2019), p. 26. DOI: 10.3390/machines7020026.

[9] N.H. Amer et al. “Modelling and Control Strategies in Path Tracking Control for Autonomous Ground Vehicles: A Review of State of the Art and Challenges”. In: *Journal of Intelligent and Robotic Systems* 86.2 (2017), pp. 225–254. DOI: 10.1007/s10846-016-0442-0.

[10] R. Rajamani. *Vehicle Dynamics and Control.* Second. Mechanical Engineering Series. Clarendon Press, 2012. ISBN: 978-1-4614-1432-2. DOI: DOI10.1007/978-1-4614-1433-9.

[11] W. Schwarting, J. Alonso-Mora, and D. Rus. “Planning and Decision-Making for Autonomous Vehicles”. In: *Annual Review of Control, Robotics, and Autonomous Systems* 1.1 (2018), pp. 187–210. DOI: 10.1146/annurev-control-060117-105157.

[12] J. Wang, J. Steiber, and B. Surampudi. “Autonomous ground vehicle control system for high-speed and safe operation”. In: *2008 American Control Conference*. 2008, pp. 218–223. DOI: 10.1109/ACC.2008.4586494.

[13] J. Kong et al. “Kinematic and dynamic vehicle models for autonomous driving control design”. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. 2015, pp. 1094–1099. DOI: 10.1109/IVS.2015.7225830.

[14] J.R. Hass G.B. Thomas Jr. M.D. Weir. *Thomas' Calculus Early Transcendentals.* Twelfth. Pearson Education, 2010. ISBN: 978-1-292-02123-3.

[15] Juan Cabrera et al. “A Procedure for Determining Tire-Road Friction Characteristics Using a Modification of the Magic Formula Based on Experimental Results”. In: *Sensors* 18 (Mar. 2018), p. 896. DOI: 10.3390/s18030896.

[16] K.J. Åström and Bjorn Wittenmark. *Computer–Controlled Systems Theory and Design.* Oct. 2011. ISBN: 0-486-48613-3.

[17] Atsushi Sakai. *cubic_spline_planner.py*. https://github.com/AtsushiSakai/PythonRobotics/blob/master/PathPlanning/CubicSpline/cubic_spline_planner.py. 2020, commit 77f6277.

[18] Steven Diamond and Stephen Boyd. "CVXPY: A Python-Embedded Modeling Language for Convex Optimization". In: (2016). arXiv: 1603.00943 [math.OC].

[19] A. Domahidi, E. Chu, and S. Boyd. "ECOS: An SOCP solver for embedded systems". In: *2013 European Control Conference (ECC)*. 2013, pp. 3071–3076. DOI: 10.23919/ECC.2013.6669541.

[20] E. Lucet, R. Lenain, and C. Grand. "Dynamic path tracking control of a vehicle on slippery terrain". In: *Control Engineering Practice* 42 (2015), pp. 60–73. ISSN: 0967-0661. DOI: 10.1016/j.conengprac.2015.05.008.

[21] G. Bayar et al. "Improving the trajectory tracking performance of autonomous orchard vehicles using wheel slip compensation". In: *Biosystems Engineering* 146 (2016), pp. 149–164. ISSN: 1537-5110. DOI: 10.1016/j.biosystemseng.2015.12.019.

[22] James Balkwill. "Chapter 2 - Tyres". In: *Performance Vehicle Dynamics*. Ed. by James Balkwill. Butterworth-Heinemann, 2018, pp. 11–52. ISBN: 978-0-12-812693-6. DOI: https://doi.org/10.1016/B978-0-12-812693-6.00002-X.

[23] G.S. Vorotovic et al. "Determination of cornering stiffness through integration of a mathematical model and real vehicle exploitation parameters". In: *FME Transactions* 41 (Jan. 2013), pp. 66–71.

[24] Jakob Bechtoff, Lars Koenig, and Rolf Isermann. "Cornering Stiffness and Sideslip Angle Estimation for Integrated Vehicle Dynamics Control". In: *IFAC-PapersOnLine* 49.11 (2016). 8th IFAC Symposium on Advances in Automotive Control AAC 2016, pp. 297–304. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2016.08.045. URL: http://www.sciencedirect.com/science/article/pii/S2405896316313672.